Hierarchical-based Clustering using Local Density Information for Overlapping Distributions *

Damaris Pascual[†], Filiberto Pla[‡], J. Salvador Sánchez[‡]

[†] Dept de Ciencia de la Computación, Universidad de Oriente,
Av. Patricio Lumunba s/n, Santiago de Cuba, CP 90100, Cuba
dpascual@csd.uo.edu.cu
[‡] Dept. Llenguatges i Sistemes Informàtics, Universitat Jaume I,
12071 Castelló, Spain,
{pla, sanchez}@lsi.uji.es

Abstract

Clustering techniques are widely used in many application fields like image analysis, data mining, and knowledge discovery, among others. In this work, we present a new clustering algorithm to find clusters of different sizes, shapes and densities, able to deal with overlapping cluster distributions and background noise. The algorithm is divided in two stages, in a first step; local density is estimated at each data point. This local density is used to initialize the clustering grouping the objects around the object of local maximum density (core point). In a second stage, a hierarchical approach is used by merging clusters according to the introduced cluster distance, also based on local density in-formation. Experimental results on synthetic and real databases show the validity of the proposed method.

Keywords: density based clustering, overlapped distributions.

1 Introduction

Clustering algorithms are techniques widely used to discover relevant distributions and relationships in databases. The problem of clustering can be defined as: Given n points belonging to a d-dimensional space, provided some measurement of similarity or dissimilarity, the aim is to divide these points into a set of clusters so that the simi-larity between

* This work has been partially supported by projects IST-2001-37306 from EU, TIC2003-08496 from the Spanish CICYT, and GV04A/705 from Generalitat Valenciana.

patterns belonging to the same cluster is maximized whereas the similarity between patterns of different clusters is minimized.

Basically, there are two approaches in clustering techniques: the partitional approach and the hierarchical approach [7]. The partitioning methods build a partition from the database of n objects in k clusters. These algorithms assume a priori knowledge about the number of classes in which the database must be divided. The K-means is the best known partitional algorithm.

Hierarchical methods consist of a sequence of nested data partitions in a hierarchical structure, which can be represented as a dendogram. There exist two hierarchical approaches: agglomerative and divisive. The first one can be described in the follow-ing way: initially each point of the database form a single cluster, and in each level, the two most similar clusters are joined, until either a single cluster is reached with all the data points, or some stopping condition is defined, for instance, when the distance between the clusters is smaller than certain threshold. In the divisive approach, the process is the other way around.

The Single Link (SL) and the Complete Link (CL) methods are the most well known hierarchical strategies [3]. Some hierarchical algorithms are based on proto-types selection, as CURE [4]. On the other hand, in density–based algorithms, the clusters are defined as dense regions, where clusters are separated by low density areas [5]. Some of the most representative ones of this approach are DBSCAN [1], KNNCLUST [7] and SSN [2] algorithms.

Some of the problems these algorithms fail to tackle are the fact that clusters are not completely separable, due to the overlapping of cluster distributions in usual real situations, and the presence of noisy samples. In this work we present an algorithm based on a hybrid strategy between the hierarchical and density-based approaches, with the aim of dealing with overlapped clusters and noisy samples, in order to discover the most signifcant density based distributions in the database.

2 Hierarchical Clustering using Local Probability Density

The objective of the algorithm here presented is to detect clusters of different shapes, sizes and densities even in the presence of noise and overlapping cluster distributions. The algorithm is a mixture of a density-based and a hierarchical-based approach, and it is divided in two stages. In the first stage, the initial clusters are constructed using a density-based approach. In a second stage, a hierarchical approach is used, based on a cluster similarity function defined in terms of cluster density measures and distances, joining clusters until either arriving to a pre-defined number or reaching a given stop-ping criterion.

2.1 Estimating Local Density

Let *X* be a set of patterns provided with a similarity measure between patterns *d*. Let *x* be an arbitrary element in the dataset and *R>0*. The neighbourhood *VR* of radius *R* of *x* is defined as the set:

$$V_R(x) = \{y / d(x, y) \le R\}$$

and the local density p(x) of the non-normalized probability distribution at point x as:

$$p(x) = \sum_{i=1}^{N_x} \exp\left(- \frac{d^2(x, x_i)}{R^2} \right) \tag{1}$$

where x_i are the points that belong to the neighbourhood of radius *R* of *x*, *VR*.

In the algorithm we are going to differentiate between two concepts: core cluster and cluster. We will call core clusters to the sets that are obtained after applying the first stage of the algorithm, and we will call cluster to the sets of core clusters that will be grouped into clusters in a further stage.

2.2 Defining Cluster Similarities

As the objective of the second stage is to perform a hierarchical algorithm between the classes obtained in the first stage we need to define the di-similarity between two clusters.

Given two core clusters Ci and Cj , let us define the distance between them as:

$$d'(C_i, C_j) = \min \{d(x_i, x_j)\}; \ \forall\, x_i, x_j \,/\, x_i \in C_i \text{ and } x_j \in C_j$$

Given two clusters Ki and Kj , let us define distance between them as:

$$\bar{d}(K_i, K_j) = \frac{P_c - P_m}{P_c} \ (1 + d''(K_i, K_j)) \tag{2}$$

where

$$d''(K_i, K_j) = \min \{d'(C', C'')\}, \ \forall\, C', C'' / C' \in K_i \text{ and } C'' \in K_j$$

Given a core cluster *C* , let us define the centre of the core cluster to the point whose density is maximal within the core cluster. Let *x'* and *x''* be the centres of C_i and C_j respectively. Therefore, let us define Pc as the minimum density of the core cluster centres *x'* and *x''*, that is,

$$P_c = \min(p(x'), p(x''))$$

In equation (2), P_m is the density of the midpoint between the two core clusters, that is, it is the midpoint of the border between both core clusters, which is defined as the midpoint between the nearest points xb_i and xb_j, one from each core cluster. To estimate the density of such a midpoint, it is interpolated from the density values of the mentioned points belonging to each cluster. To calculate the interpolated value, two different cases are taken into account when comparing the two neighbouring core clusters:

1. If $d'(C_i, C_j)>R$, the midpoint do not have points in its neighbourhood of radius R, then we take $P_m=0$ and the distance between the clusters becomes:

   $$\bar{d}(K_i, K_j) = 1 + d''(K_i, K_j) \tag{3}$$

   That is, in the cases that clusters are well separated, the di-similarity measurement is given by the distance between the nearest core clusters.

2. If $d'(C_i, C_j)\leq R$, the midpoint has got points in its neighbourhood from both core clusters. In this case, the midpoint x is defined as either the border point xb_i from core cluster C_i or the border point xb_j from core cluster C_j. In order to avoid negative values in expression (2), the midpoint is chosen as the border point such as,

   if $P_c=p(x')$, then $x=xb_i$, and $P_m=p(xb_i)$

   else $x=xb_j$, and $P_m=p(xb_j)$

2.2 Grouping Core Clusters into Clusters

The clustering algorithm here presented consists of a hierarchical agglomerative strategy based on a Single Link approach, using the di-similarity measures defined in the previous Section. The use of such dissimilarity measures defines the behaviour of the clustering process and the response to the different local distributions of the patterns in the data set.

In a few words, the dissimilarity measure defined in (2) is aimed at considering that clusters are more similar when they probability distributions are either nearer in the feature space by means of a Single Link concept, or when their probability distributions are more overlapped. In the last case, when probability distributions are over-lapped ($d'=0$), the measure of similarity becomes the probability density term that appear in equation (2), which is a local estimate of the mixed probability distributions at the clusters border.

Therefore, the proposed algorithm can be summarized in two stages as follows:

```
First stage:
Input:   radius $R$, data points and density noise threshold
Output:  $N$ core clusters
1. Initially, each point of the database is assigned to a single core cluster.
2. For each point $x$, calculate its neighbourhood of radius $R$, $VR(x)$
3. For each point $x$ in the database, estimate its probability density $p(x)$ according to
   expression (1).
4. Assign each point x to the core cluster of the point $x_c$ in its neighbourhood, being
   $x_c$ the point with maximal density in the neighbourhood.
5. Mark all core clusters with density less than the density noise threshold as noise
   core clusters. The rest are the resulting $N$ core clusters.
```

```
Second stage
Input:   $N$ core clusters
Output:  $K$ clusters
1. Initially, assign each core cluster from the first stage to a cluster. Therefore, there
   are $N$ clusters with one core cluster.
2. Repeat until obtaining $K$ clusters,
    2.1 Calculate the distance between each pair of clusters using expression (2)
    2.2 Join the clusters that their distance is minimum
3  Assign the noise core clusters to a nearest.
```

3 Experimental Results

In this section, some experimental results are presented aimed at evaluating the proposed algorithm, hereafter named H-density, and to compare it with some other similar algorithms referred in the introduction, DBSCAN, CURE and K-means. In order to test the algorithm, three groups of experiments are performed. The first one uses synthetic databases based on overlapped Gaussian distributions, in order to see the response of the proposed algorithm in these conditions. The second experiment uses two synthetic databases from [6], for comparison purposes, and to test the problem of the presence of noise, overlapping, and clusters of different sizes and shapes. Finally, some experiments are performed on two real databases.

3.1 Gaussian Databases

Four databases using Gaussian distribution were generated with different number of classes (Gaussian distributions), sizes and overlapping degrees. The number of samples and classes in each database is shown in Table 1.

Table 1. Gaussian databases generated used in the experiments.

Database	No samples	No classes
G1	4000	4
G2	6000	6
G3	6000	3
G4	8000	4

The results obtained with the proposed algorithm are shown in Figure 1, where we can notice how the algorithm has been able to correctly detect each one of the existing classes, even in the presence of significant overlapping (Figure 1 right).

The DBSCAN algorithm does not correctly detect all the classes in different databases because it is not able to separate the overlapped classes. For example, in data-base G2 it detects 3 classes for radius 5 and some noise points (Figure 2 left). If the radius is increased, it obtains three or less classes. If the radius decreases, the objects of the edge of the three classes are separated because they stay as noise points. This can be noticed in Figure 2 right, where the number of noise points increases. The others databases have a similar behaviour.

The CURE algorithm detects all the classes in databases G1, G3 and G4, but in database G2 it correctly detects three classes. However, in the case of six classes, it cannot detect the 4 classes that are highly overlapped. The same happens with the K-means algorithm, it detects the classes in databases G1, G3 and G4, but in the case of the database G2 the results depend on the initial centres (see Figure 3).

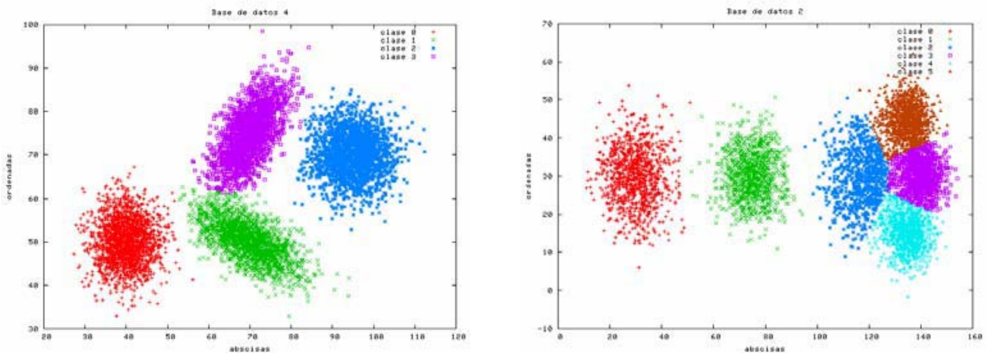

Figure 1. Results of the H-density algorithm on G4 (left) and G2 (right) databases.

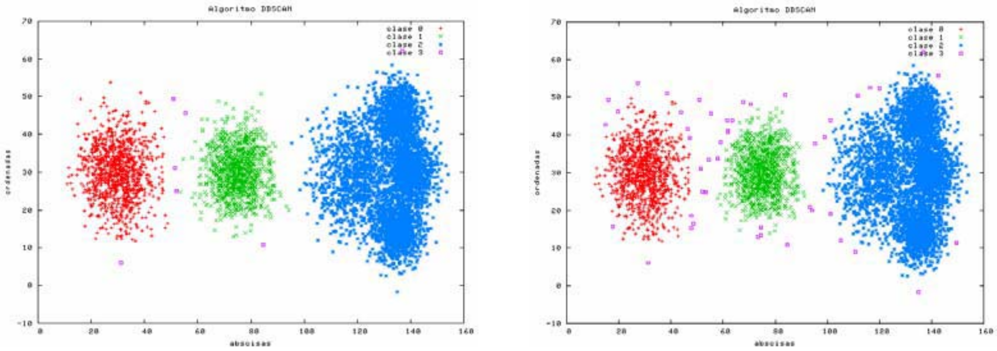

Figure 2. Results of the DBSCAN algorithm on G2 using radius=5, MinPts=4 (left), and radius=3, MinPts=4 (right).

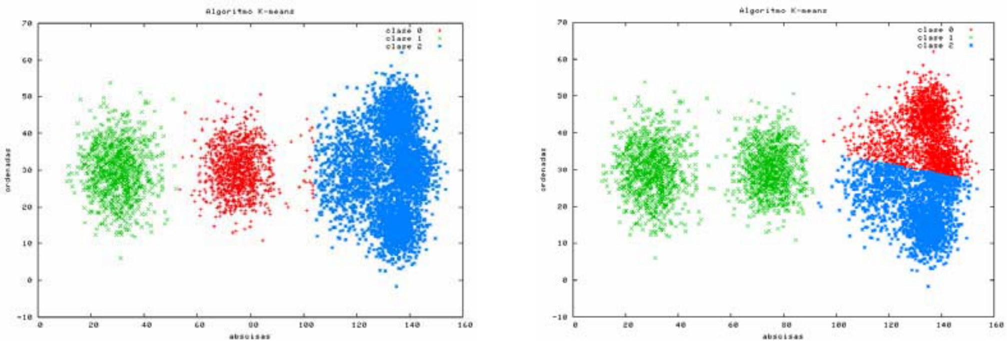

Figure 3. Results of the K-means algorithm on G2 with two different initializations.

3.2 Synthetic Databases

In [6], some experiments were presented for the DBSCAN and CURE algorithms using the databases of Figure 4 (see [6] for comparison results with those algorithms). Notice the presence of clusters of different shape, size, noise and overlapping. Figure 4 shows the result of applying the proposed H-density algorithm on these databases. Note how the algorithm has correctly grouped the main clusters present in the data set. Figure 5 shows the result of the K-means algorithm for 6 clusters (left) and 9 clusters (right) of the corresponding databases. The errors in the grouping are noticeable.

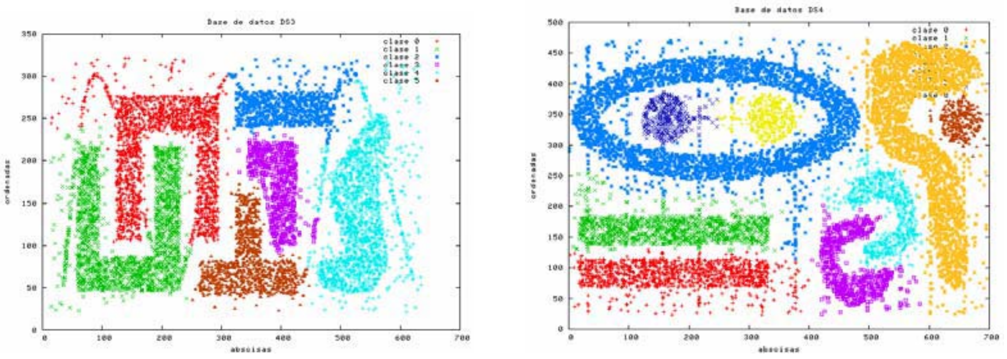

Figure 4. Results of the H-density algorithm on databases from [6].

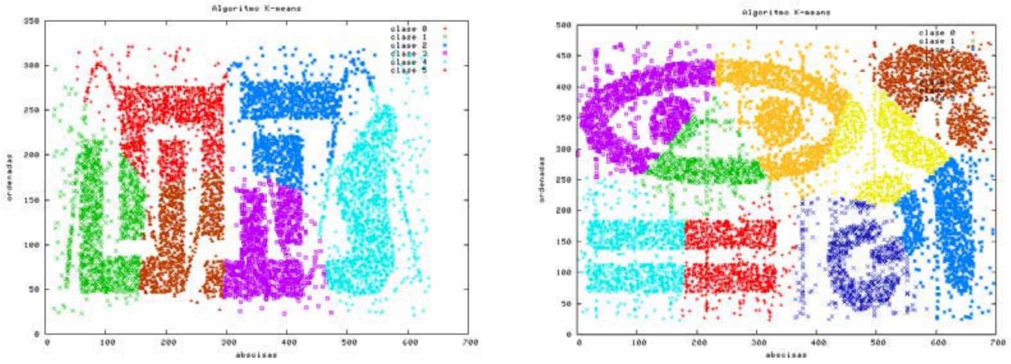

Figure 5. Results of the K-means algorithm on databases from [6]. Left: for 6 clusters. Right for 9 clusters.

3.3 Real Databases

Two real databases were used in this experiment, Iris and Cancer. The first one is a database of Iris plants containing 3 classes, with a total of 150 elements, 50 each of the three classes: Iris Setosa, Iris Versicolour, Iris Virginica. The number of attributes is 4, all numeric. The first class, Iris Setosa, is linearly separated from the other two.

In the first experiment, all the algorithms were run to obtain two classes, and all of them obtained 100% of correct grouping or classification, that is, all the tested algorithms were able to correctly separate the Setosa class from the other ones.

In a second experiment, the algorithms were run to find three clusters. The results are shown in Table 2. Notice how, due to the overlapping between Versicolour and Virginica classes, the proposed H-density algorithm outperforms the other ones reaching a 94% correct classification. In the case of the Cancer database, it has 2 classes. The proposed H-density algorithm obtained a 95.461% of correct classification, the same as CURE (Table 3).

Table 2. Classification rate of the clustering algorithm on Iris database.

Algorithm	% in two classes	% in three classes
DBSCAN	100	71.33
CURE	100	83.33
K-means	100	88.33
H-Density	100	94.00

Table 3. Classification rate of the clustering algorithms in Cancer database (two classes).

Database	DBSCAN	CURE	K-means	H-Density
Cancer	94.28	95.461	95.04	95.461

5 Conclusions and Further Work.

A hierarchical algorithm based on local probability density information has been presented. The way the density of the probability distribution is estimated, and the use of this information in the introduced dissimilarity measure between clusters, provides to the algorithm a mechanism to deal with overlapping distributions and the presence of noise in the data set. The experiments carried out show satisfactory and promising results to tackle these problems usually present in real databases. The experiments also show the proposed algorithm outperforms some existing algorithms. Future work is directed to unify the treatment of noise and overlapping in the process, and to introduce a measure to assess the right number of clusters in the hierarchy.

References

[1] Ester, M.; Kriegel, H. P.; Sander, J. and Xu, X.; A density-based algorithm for discovering clusters in large spatial databases with noise. In Proc. of the second International Conference on Knowledge Discovery and Data Mining, Portland, (1996) 226-231.

[2] Ertöz, L.; Steinbach, M. and Kumar V.: Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data. In Proceedings of Third SIAM International Conference on Data Mining, (2003).

[3] Fred A. L. and Leitao J.: A New Cluster Isolation Criterion Based on Dissimilarity Increments. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol 25, No 8, (2003) 944-958.

[4] Guha, S.; Rastogi, R. and Shim, K.; CURE: An Efficient Clustering Algorithm for Large Databases. In Proceedings of ACM SIGMOD International Conference on Management of Data, . ACM, New York, (1998) 73-84.

[5] Hinneburg A. and Keim D.A.: An efficient Approach to Clustering in Large Multimedia Databases with Noise. In Proc. of the ACM SIGKDD, (1998).

[6] Karypis, G.; Han, E.H. and Kumar, V.; Chameleon: A Hierarchical Clustering Algorithm Using Dynamic Modeling. In the IEEE Computer Society. Vol 32, No 8 (1999) 68-75.

[7] Tran T. N., Wehrens R. and Buydens L.M.C.: Knn Density-Based Clustering for High Dimensional Multispectral Images. Analytica Chimica Acta 490 (2003) 303–312.