Clustering by density in any metric space

Alain Guénoche

IML, 163 Av. de Luminy, 13009 Marseille (France)
guenoche@iml.univ-mrs.fr

RÉSUMÉ. Nous présentons une nouvelle méthode de classification à partir d'une distance pour construire des classes et des partitions d'un ensemble X. La construction repose sur une valeur de densité calculée en chaque sommet d'un graphe $G = (X, E)$ construit à partir de la matrice de distance. Les classes sont des parties connexes de G de forte densité.

MOTS-CLÉS : Classes, partitions, densité, espace métrique

1. Graph, classes and partitions

Given a distance matrix on X, establishing partitions (all the elements are clustered in disjoint classes) is generally made optimizing a criterion over the set of all the partitions with a number of classes that must be indicated. In this paper, we investigate a different approach. From the distance matrix, we first build a valued graph $\Gamma = (X, E)$, for each vertex a density function De is evaluated and we perform clustering from Γ and De. Our algorithm differs from similar approaches (Fraissex and Kuntz 1991, Matsuda et al. 1999, Rives and Galitsky 2003, Rougemont and Hingamp 2003) in many ways ; the graph is not a threshold graph, we use the valuation of the edges to measure a density in each vertex and to perform progressive clustering.

Starting from a distance matrix, a threshold graph is generally used, keeping only edges corresponding to distance values lower than a threshold σ. The nested family of threshold graphs is obtained from a distance matrix making σ vary from 0 to $Dmax$, the largest distance value. But for many metrics, as the Sczekanowski-Dice distance on graphs, or Manhattan distance on boolean array, choosing a threshold value is a delicate problem. On sparse graph, most of the distance values are equal to the maximum, and a small threshold gives many singletons. So we try to keep the same number of edges from any vertex, selecting a graph which generally contains a single connected component.

1.1. *Graph*

Given a distance matrix, $D : X \times X \to \mathbb{R}$, the first operation is to select a degree δ which works as a threshold. From any element x, the distance values $D(x, y)$ are ranked in decreasing order and the δ-th value gives the σ_x threshold. Then, we take as edges all the pairs (x, y) such that $D(x, y) \leq \sigma_x$. Let $n = |X|$, $m = |E|$ and $\Gamma_\delta = (X, E)$ the corresponding graph. It is not a classical threshold graph on D, since the threshold values are not the same for all the vertices. And it is not really a regular graph with degree δ, because the edge selection process is not symmetrical ; y can be one of the δ closest element to x but it can be different for x to y. This directed relation is symmetrized in Γ_δ ; consequently, there can be more than δ vertices incident to x.

When there is no ambiguity on the δ value, the graph will simply be denoted Γ. For any part Y of X, let $\Gamma(Y)$ be the set of vertices not in Y that are adjacent to Y. Thus, the neighborhood of x is denoted $\Gamma(x)$, the degree of a vertex x is $Dg(x) = |\Gamma(x)|$.

1.2. *Density function*

For each vertex x, we evaluate a density function denoted De which would be high when the elements of $\Gamma(x)$ are close to x. We propose a density function computed from the average length of the edges from x.

$$De(x) = \frac{Dmax - \frac{1}{Dg(x)} \sum_{y \in \Gamma(x)} D(x,y)}{Dmax}$$

The dense classes are by definition connected parts in Γ sharing high density values. Our initial idea was to search for a density threshold and to consider the partial subgraph whose vertices have a density greater than this threshold. Classes would have been its connected components. This strategy does not give the expected results. Enumerating all the possible threshold values, we have observed that often none was satisfying. By decreasing the threshold, we often obtain only a single growing class, and many singletons. Since there is no straightforward way to determine a threshold, the *local maximum values* of the density function are considered.

1.3. *Classes at three levels*

We construct classes in three steps :

Kernels

A kernel, denoted K, is a connected part of Γ, obtained by the following algorithm : we first search for the local maximum values of the density function and we consider the partial subgraph of Γ reduced to these vertices.

$$\forall x \in K, \forall y \in \Gamma(x) \text{ we have } De(x) \geq De(y).$$

The initial kernels are the connected components of this graph. More precisely, if several vertices with maximum value are in the same kernel, they necessary have the same density value ; otherwise the initial kernels are singletons. Then, we assign recursively to each kernel K the vertices (i) having a density greater than or equal to the average density value over X and (ii) that are adjacent to only one kernel. Doing so, we avoid any ambiguity in the assignment, postponing the decision when several are possible.

The number of kernels is the number of classes and it remains unchanged in the following. So it is not necessary to indicate first this number, as for all the alternative methods optimizing a criterion. We shall see that it performs pretty well, when there is a small number of classes, having from 30 to 50 elements.

Extended classes

At the second level, we just assign elements that are connected to a unique kernel, whatever is their density. If an element not in a kernel is connected to several ones, the decision is again postponed.

Complete classes

Finally, to get partitions, we assign the remaining elements to one class. For x and any extended class C to which it is connected, we compute the number of edges between x and C, and also its average distance value to C. Finally there are two candidates, the majority connected class C_m and the closest one C_d. If they are identical, x is connected to it. And if they are different we apply the empiric following rule : if $\frac{|C_m|}{|C_d|} > 1.5$, class C_m is retained, because the number of links to C_m is clearly larger than to C_d ; otherwise C_d is retained. When classes are not necessary disjoint, x can be assigned to both classes.

1.4. *Complexity*

To establish graph Γ, it is necessary to order the distance values from any x. The computation of σ_x is in $O(n \log n)$ and the selection of the edges is in $O(n)$. Finally of the graph construction is in $O(n^2 \log n)$. To evaluate $De(x)$ it is sufficient to test the edges in the neighborhood of x which contains at most n vertices. The computation of the density function is thus in $O(n^2)$.

Kernel computation is in $O(n^2)$ to find the local maximum vertices, and in $O(m) = O(n^2)$ to determine the kernel elements. During the following steps, for any x we count its connections to the kernels, and then to the extended classes. Both are also in $O(n^2)$. Finally, the complexity of the clustering method is $O(n^2 \log n)$.

2. Monte Carlo simulation on binary data

In order to show that this method permits to recover existing classes, we have tested it on euclidian, boolean and graph distances. For all these metric spaces, we have generated distance arrays containing initial classes. The two main points to assess are the ability to recover the correct number of classes and their quality. Here, we only detail the results on the symmetrical difference distance on binary tables.

First, we have developed a generator of binary tables (n rows, m columns) in which there are p classes. Each class is indicated by a specific attribute taking value 1 only for its elements. For the $m - p$ other attributes, value 0 or 1 is selected at random, with .5 probability. The attributes are weighted : 1.0 for those characterizing the classes and a random number between 0 and 1 for the others. At each trial, the symmetrical difference distance between rows is computed.

2.1. *Quality of the classes compared to the initial partition*

For the three levels of classes, we would like to estimate the quality of the obtained sets of classes, and so the efficiency of the clustering process. The initial partition is denoted $P = \{C_1, ..C_p\}$. Let n'_c be the number of classified vertices at each level. They are distributed in p' classes denoted $C'_1, ..C'_{p'}$ realizing a partition P' over a subset of X for the kernels and the extended classes.

We first aim to map the classes of P' onto those of P evaluating $n_{i,j} = |C_i \bigcap C'_j|$. We define the *corresponding* class of C'_j, denoted $\Theta(C'_j)$, as the one in P, that contains the greatest number of elements of C'_j. $\Theta(C'_j) = C_k$ if and only if $n_{k,j} \geq n_{i,j}$ for all i from 1 to p.

In order to measure the accuracy of the classes, we evaluate three criteria.
– τ_c : the percentage of clustered elements in P' ($\tau_c = \frac{n'_c}{n}$).
– τ_e : the percentage of elements in one of the p' classes which belong to its corresponding class in P.

$$\tau_e = \frac{\sum_i |\Theta(C'_i) \bigcap C'_i|}{n'_c}$$

– τ_p : the percentage of pairs in the same class in P' which are also joined together in P.
The first criterion measures the efficiency of the clustering process at each level ; if very few elements are clustered, the method is inefficient. For the second criterion, we must compute, for each class in P', the distribution of the elements of any initial class to define its corresponding class in P. Thus it can be interpreted as a percentage of ”well classified” elements. The third one estimates the probability for a pair in one class of P' to come from a single class in P.

Remark : The two last criteria may reach their maximum value (1.0) even when partitions P and P' are not identical. When a class in P is subdivided in two parts, they will have the same corresponding class ; all their elements will be considered as well classified, and the rate of pairs will also be equal to 1.

2.2. *Results*

We have generated 200 binary tables with 200 elements distributed in 5 classes, setting first $m = 10$ and then $m = 20$. Table 1 indicates the percentage of trials giving each computed number of classes when δ varies.

	Nb. of classes	2	3	4	5	6	7	8	9
	$\delta = 18$	0.0	.01	.04	.41	.36	.12	.04	.01
m=10	$\delta = 20$	0.0	.01	.16	.63	.15	.05	.01	0.0
	$\delta = 22$	0.1	.05	.32	.52	.09	.01	.01	0.0
	$\delta = 18$	0.0	.04	.14	.29	.25	.19	.09	.01
m=20	$\delta = 20$	0.0	.05	.27	.32	.23	.12	.01	0.0
	$\delta = 22$	.02	.11	.30	.33	.19	0.4	.01	0.0

Table 1 : Distribution of the number of classes according to the number of attributes m and the degree δ.

One can see that for $m = 10$ and $\delta = 20$ the correct number of classes is the most frequently recovered. It differs at most of one unity in 94% of the trials. It is less promising when $m = 20$ but, in that case the 5 partitioning attributes are hidden by 15 random ones. Now we evaluate the quality of the classes, using the above criteria with $\delta = 20$, because problems giving a number of classes greater than 5 are compensated by those giving a lower one.

	$m = 10$			$m = 20$		
	τ_c	τ_e	τ_p	τ_c	τ_e	τ_p
Kernels	.46	1.0	1.0	.28	.90	.83
Classes	.80	.96	.93	.47	.80	.68
Partitions	1.0	.92	.86	1.0	.77	.63

Table 2 - Average results of the quality criteria.

These two tables prove that this clustering method is able to recover classes in binary tables when some attributes possess the partitioning information. And the number of classes can be correctly predicted.

The density clustering method has many advantages over classical partitioning ones.
– It allows both to extract partial classes (that do not cover the complete set of elements) and to built a partition.
– It provides the number of classes, and the correct number can be recovered if the classes have a few ten of elements.
Finally it is a one parameter method (the initial degree) that can be used for large clustering problems.

3. Bibliography

A. GUÉNOCHE Clustering by graph density, *Proceedings of IFCS'04 conference* (2004) to appear.

H. DE FRAISSEX AND P. KUNTZ (1992) Pagination of large scale networks ; embedding a graph in $\mathbb{R}^n$ for effective partitioning, *Algorithmic review*, 2(3), 105-112.

H. MATSUDA, T. ISHIHARA, A. HASHIMOTO (1999) Classifying molecular sequences using a linkage graph with their pairwise similarities, *Theoretical Computer Science*, 210, 305-325.

A.W. RIVES AND T. GALITSKI (2003) Modular organization of cellular networks, *P.N.A.S.*, 100, 1128-1133.

J. ROUGEMONT AND P. HINGAMP (2003) DNA microarray data and contextual analysis of correlation graphs, *BMC Bioinformatics*, 4 :15, 11p.

Acknowledgements

This work is supported by the "Origine de l'Homme, des Langages et des Langues" (OHLL) CNRS program.