On the Problem of Clustering Spatial Big Data

Gabriella Schoier and Giuseppe Borruso(✉)

DEAMS – Department of Economic, Business, Mathematic and Statistical Sciences
"Bruno de Finetti", University of Trieste, Via A. Valerio, 4/1 34127, Trieste, Italy
{gabriella.schoier,giuseppe.borruso}@econ.units.it

Abstract. Different motivation are related with the analysis of Spatial Big Data (SBD). Google Earth, Google Maps, Navigation, location-based service allow to obtain a great amount of geo-referenced data. Often spatial datasets exceed the capacity of current computing systems to manage, process, or analyze the data with reasonable effort. Considering SBD history methodology as Data-intensive Computing and Data Mining techniques have been useful. In this context the problem regards the analysis of of high frequency spatial data. In this paper we present an approach to clustering of high dimensional data which allows a flexible approach to the statistical modeling of phenomena characterized by unobserved heterogeneity. We consider the MDBSCAN and compare it with the classical k-means approach. The applications concern a synthetic data set and a data set of satellite images.

Keywords: Spatial data mining · Clustering algorithms · Arbitrary shape of clusters · Efficiency on large spatial databases · Handling noise · Lagrange-Chebychev metrics · Image analysis

1 Introduction

The rapid developments in the availability and access to spatially referenced information in a variety of areas, has induced the need for better analysis techniques to understand different phenomena. In particular spatial clustering algorithms, which groups similar spatial objects into classes, can be used for the identification of areas sharing common characteristics.

Clustering is an unsupervised classification of patterns - observations, data items, or feature vectors - into groups or clusters [6]. Cluster analysis can be defined as the organization of a collection of patterns - usually represented as a vector of measurements, or a point in a multidimensional space - into clusters based on similarity.

The clustering problem has been considered in many contexts and by researchers in different disciplines. It is useful in several exploratory pattern-analysis, grouping, decision-making and machine-learning situations, including data mining (see e.g. [5]), spatial data mining (see e.g. [1], [2], [8]), document retrieval, image segmentation, and pattern classification.

Clustering techniques have been recognized as primary Data Mining methods for knowledge discovery in spatial databases, i.e. databases managing 2D or 3D points, polygons etc. or points in some d-dimensional feature space (see e.g. [8], [13, [14]]).

O. Gervasi et al. (Eds.): ICCSA 2015, Part III, LNCS 9157, pp. 688–697, 2015.
DOI: 10.1007/978-3-319-21470-2_50

The aim of this paper is to compare a density based algorithm (MDBSCAN) for the discovery of clusters of units in large spatial data sets with the classical k-means method. This algorithm is a modification of the DBSCAN algorithm (Ester et. al.(1996)). The modifications of this algorithm with respect to the original regard the consideration of spatial and non-spatial variables and the use of a Lagrange-Chebychev metrics instead of the usual Euclidean one. The applications concern a synthetic data set and a data set of satellite images.

2 Spatial Big Data

Spatial data mining can be used for browsing spatial databases, understanding spatial data, discovering spatial relationships, optimizing spatial queries.

Spatial data, as other kinds of data, are becoming bigger and bigger, although since the introduction of GIS and desktop GIS in particular, GIS users and experts have become facing with the issue of managing big amount of data, even though often data were much more difficult to retrieve than today. In geographical terms, the nature of data is such that an increase of dimension of the dataset is always very possible, both in terms of the number of the records to be considered, as well as in terms of the attribute of the geographical data. Both the vector and raster data formats used in GIS analysis tend to be multidimensional, i.e., containing a quantity of elements to be considered in any form of grouping and aggregation. In any case at least two fields (if not three) are needed to store the spatial information while all the attribute data contribute to increasing the dimension of the dataset. Satellite imagery in particular represents another case, in which redundant information is also considered, as very close pixels present very little differences although weighting in the processing, storage and visualization time of the data. So compression algorithms on one side and proficient clustering tools are needed in order to extract the more precise and complete set of geographic information ([15], [16], [17]).

3 The Methodology

3.1 K-means Algorithm

The K-means algorithm is very well known (see e.g. [7]). The algorithm allocates the data points (objects) into clusters, so as to minimize the sum of the squared distances between the data points and the center of the clusters.

The centers of the clusters are initialized by randomly selecting from the data or by fixing particular data points. Then the data set is clustered in the process of assigning each point to the nearest center .When the data set has been identified, the average position of the data points within each cluster is calculated and the cluster center then moved to the average position. This process is repeated until a condition of stopping is reached, in other words the algorithm has these steps:

Step 1

Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.

Step 2

Assign each object to the group that has the closest centroid.

Step 3

When all objects have been assigned, recalculate the positions of the K centroids.

Step 4

Repeat Steps 2 and 3 until the centroids no longer move or another stopping rule is achieved. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

The K-means algorithm requires three user-specified parameters: number of clusters K, cluster initialization, and distance metric.The most critical choice is K.

K-means is typically used with the Euclidean metric for computing the distance between points and cluster centers. As a result, K-means finds spherical or ball-shaped clusters in data. K-means with Mahalanobis distance metric has been used to detect hyperellipsoidal clusters (see [9]), but this comes at the expense of higher computational cost

3.2 DBSCAN and MDBSCAN Algorithms

In this section we will consider clustering methods based on the notion of density.These regard clusters as dense regions of units which are separated by regions of low density (representing noise); moreover they may be used to discover clusters of arbitrary shape (see e.g. [3], [4], [10], [11, [12]).

Among these the *DBSCAN* algorithm (Ester (1996)) is a locality-based algorithm relying on a density based notion of clustering. Density based methods can be used to filter out noise (outliers) and discover clusters of arbitrary shape. This algorithm judges the density around the neighborhood on an unit to be sufficiently dense if the number of points within a distance *EpsCoord* of an unit is greater than *MinPts* , in this case the unit is a *core point* otherwise is a *border point*. This algorithm has been generalized in different papers (e.g. Sander (1998)).

The key idea is that for each point of a cluster the neighbourhood of a given radius has to contain at least a minimum number of points, i.e. the density in the *neighborhood* has to exceed some threshold. The shape of a *neighborhood* is determined by the choice of a distance function for two points p and q, denoted by *dist(p,q)* The *Epscoord* neighbourhood of a point q is defined by

$$N_{\varepsilon}(q) = \{q \in D | dist(p,q) \leq \varepsilon\},$$

where D is a data set of points.

A naive approach could require for each point in a cluster that there are at least a minimum number (*MinPts*) of points in an *Eps-neighborhood* of that point. However, this *ap*proach fails because there are two kinds of points in a cluster, points inside of the cluster (*core points*) and points on the border of the cluster (*border points*).

In general, an *Eps-neighborhood* of a border point contains significantly less points than an *Eps-neighborhood* of a core point. Therefore, one would have to set the minimum number of points to a relatively low value in order to include all points belonging to the same cluster. This value, however, will not be characteristic for the respective cluster particularly in the presence of noise. Therefore, one has to require that for every point p in a cluster C there is a point q in C so that p is inside of the *Eps-neighborhood* of q and $N_\varepsilon(q)$ contains at least *MinPts* points.

This definition is elaborated in the following: a point p is *density reachable* from a point q if there is a chain of points $p_1,p_2,\ldots\ldots,p_{n-1},p_n$ where $p_1=p$ and $p_n=q$ such that p_i is *direct density reachable* from p_{i+1}.

Moreover a point p is *directly density reachable* from a point q if p belongs to the neighborhood of q and q is a core point.

The clustering formed from DBSCAN follows the rules below:

1. A point can only belong to a cluster if and only if it lies within the *Epscoord-* neighborhood of some core point in the cluster.
2. A *core point o* within the *Epscoord-* neighborhood of another core point p must belong to the same cluster as p.
3. A *border point r* within the *Epscoord-* neighborhood of some core point must belong to the same cluster to at least one of the core points.
4. A border point which does not lie within the *Epscoord-* neighborhood of any core point is considered to be noise.

There are some problems regarding both the considerations of spatial and non-spatial variables and the use of different distances.

Our generalization *the Modified Density-Based Spatial Clustering of Applications with Noise* MDBSCAN ([]) considers an approach density based that takes into account at the same time spatial and non spatial variables. It has a similar structure of the DBSCAN but introduces a notion of proximity not only for spatial characteristics but also for non spatial characteristics.

The key idea is that the cardinality of the neighborhood of an unit is given not only by counting the number of units that have distance from it less than a radius *EpsCoord* (the limiting distance value for the spatial variables) but by the points that have distance less than *EpsCoord* and that are "sufficiently" similar as regards non spatial attributes. In order to have a sufficient homogeneity for the non-spatial attributes another radius *Eps*, that represent the threshold for the distances calculated on the bases of the non-spatial variables is evaluated. In so doing we want to find clusters of elements which are spatially close to each other and homogeneous as regards other observed variables. The elements of such clusters may be interpreted as elements which are similar as regards some variables and belong to the same spatial area.

The distance function determines the shape of the neighbourhood. *MinPts* is the minimum number of points that must be contained in the neighbourhood of that point in the cluster. In the following we present the main steps of our algorithm the *Modified DBSCAN* (*MDBSCAN*)

Step 1.

Insert the values of the parameters:

SetOfPoints representing the matrix with the values of the non-spatial variables, *Coordinates* representing the spatial variables ,

Eps the limiting distance value for the non-spatial variables ,

EpsCoord the limiting distance value for the spatial variables,

MinPts the minimum number of points to consider a point as a *core point*.

Step 2.

Chose an arbitrary point *i* in the database, if *i* is a *core point* built the cluster by choosing all the points which are density-reachable from i else if it is a *border point* pass to the point *i+1*.

Step 3.

Classify the points which are not *density-reachable* as *noise*.

The value of the parameter *MinPts* is fixed as in Ester (1996). In order to determine the parameter *EpsCoord*, regarding the spatial variables, and the parameter *Eps*, regarding the non spatial variables we consider the algorithms *SorteKdist* and respectively *SorteKdist2*. The former *SorteKdist*, used for the spatial variables, is implemented following Ester (1996), it evaluates, for every point of the data set represented by the matrix of coordinates, the distances from the nearest *k*-points (belonging to the same data set),orders them in decreasing way and gives a graphical representation of these distances (for the choice of *k* see Sander (1996) and see fig:1 for an example.

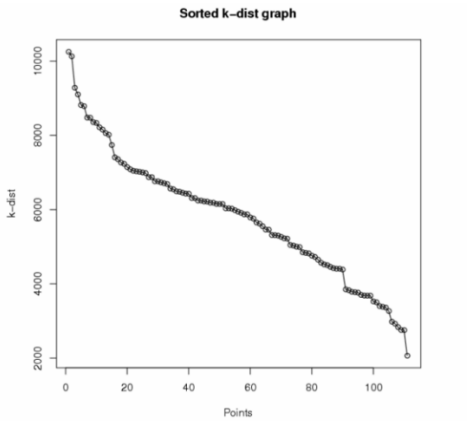


Fig. 1. Sorted k-distances graph for a sample data set

The latter *SorteKdist2* used for non spatial variables is similar, it evaluates, for every point of the data set represented by the matrix *SetOfPoints*, the distances from the nearest *k-points* (belonging to the same data set) and gives the mean value.

As written before the form of the neighborhood is determined by the choice of the distance function between two points $\underline{x}_i$ and $\underline{x}_j$. The euclidean distance generates a sphere around the point of circular shape (MDBSCAN (euc))

$$_2d_{ij} = \left[\sum_{s=1}^{p}\left|x_{is} - x_{js}\right|^2\right]^{1/2}$$

The Lagrange-Tchebychev distance generates a sphere around the point of square shape (MDBSCAN (lag))

$$_\infty d_{ij} = \max_s\left|x_{is} - x_{js}\right|$$

We have verified the validity of the MDBSCAN based on the Lagrange distance and that based on Euclidean on the base of the CPU times, as one can see from Fig. 2 the time (in seconds) with the increase of the number of points remain similar

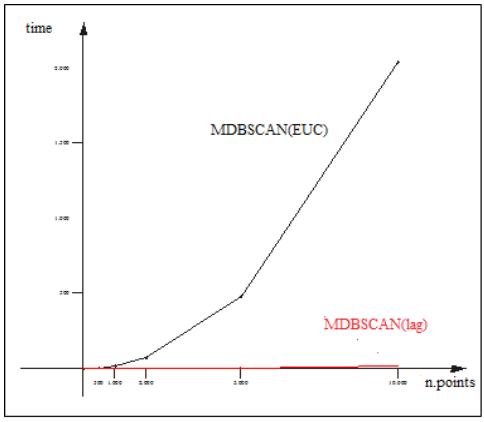


Fig. 2. MDBSCAN(lag) and MDBSCAN(euc)

4 The Application Results and Discussion

4.1 MDBSCAN versus K-means Clustering

In order to test the MDBSCAN(lag) and the K-means algorithm we consider some applications to a synthetic dataset and to a satellite images data set.

The analysis on bit-map images are a real example of raster spatial analysis, the image is formed by a regular grid of pixel and to every cell a colour has 24 bit (16 million of colours are possible).We have considered images that have 300 pixel by side (for a total of 90000 pixel) in a space of RGB colours to 24 bpp (bit by pixel, about 16 millions of colours) this colour format is known as true-colour.

Every pixel is a statistical unit, a point in the space of five dimensions: two relatively of the spatial attributes, the other to the non-spatial attributes. In order to apply the algorithm a standardization of the variables has been performed. The implemented MDBSCAN(lag) algorithm involved choosing different thresholds for the

coordinates and for the other indices. As regards the spatial variables the algorithm *SorteKdist* gives the first point in the first valley. As regards the non-spatial variables the algorithm *SorteKdist2* has been implemented.

After the standardization of the variables the *R* language and a visualization language have been used for the analysis.

In the data set presented in Fig. 1 we consider a simple synthetic data set with two spherical clusters without noise.

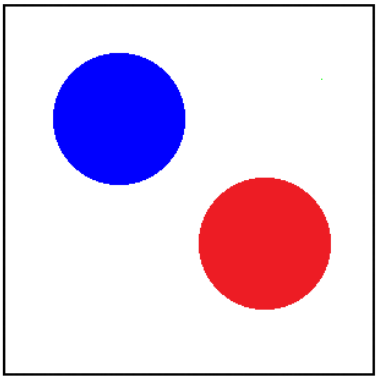
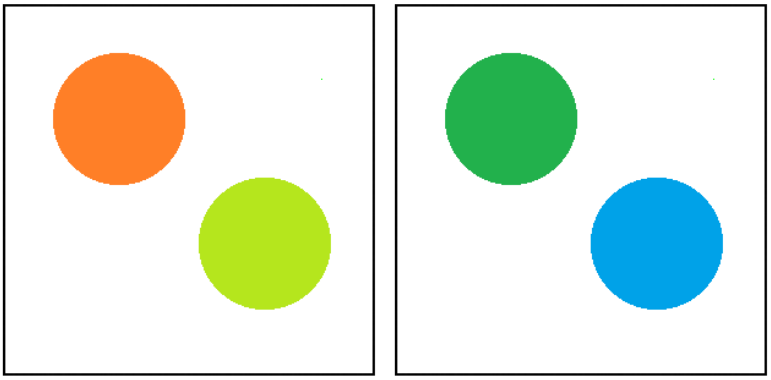


Fig. 1. (a) A synthetic data set

As one can see both algorithms find the clusters.



(a) (b)

Fig. 2. (a) K-means and (b)clustering MDBSCAN results

The difficulty for K-means algorithm is handling clusters of arbitrary shape and noise. This is not the case of the MDBSCAN.

Next applications of MDBSCAN regards data set of satellite images

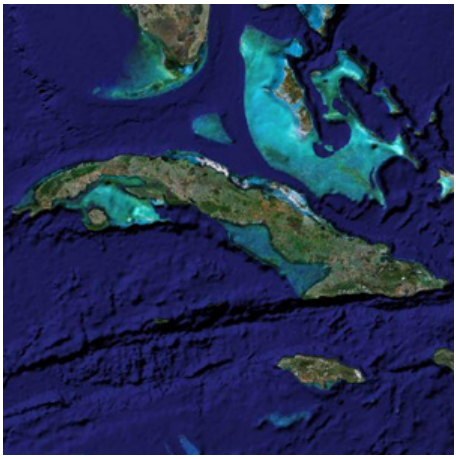


Fig. 3. (a) A satellite images data set

The algorithm was used on a satellite image of the Island of Cuba in the Gulf of Mexico in order to understand the performance of the algorithm over areas in which the land and sea borders are not so easy to detect. In Figure 3 the starting dataset is portrayed, while Figure 4 reports in false colors the different groups or clusters identified by the procedures.

The more easily detectable parts of the pictures have been assigned a common color. From Figure 4 it is therefore possible to observe in red the water component of he sea, while other colours as green and brown highlights the island and the other components as part of the Florida peninsula.

The less easily detectable parts of the pictures have been assigned a white colour representing a level of 'noise'.

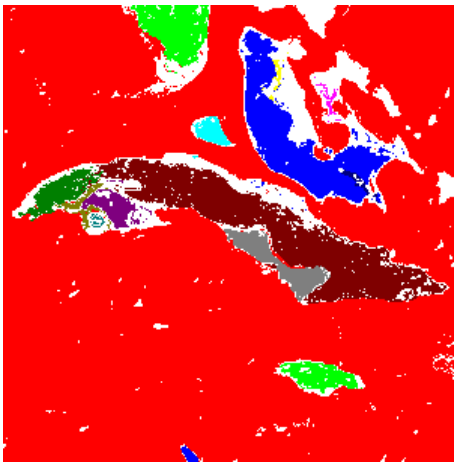


Fig. 4. MDBSCAN clustering results

5 Conclusions

In this paper two way of clustering methods applied to identify homogeneous areas are compared: K-means, and MDBSCAN.

MDBSCAN is an algorithm which is a modification of the original DBSCAN algorithm. Our algorithm takes into consideration both spatial and non spatial variables relevant for the phenomena that has to be analyzed. To improve computational aspects we proposed to use a Lagrange-Chebychev metrics instead of the Euclidean one.

The applications regards both synthetic and real datasets. The spatial clustering analysis allowed to obtained good bit-map images and good representation of satellite images.

The main advantage of K-means algorithm is its simplicity and speed which allows it to run on large datasets. One problem of the application of the K-means is the necessity of knowing *a prori* the number of clusters. Other problem regard the identification of noise (outliers) and the discovering of clusters of arbitrary shape

MDBSCAN is robust enough to identify clusters in noisy data, requires just a few parameters and is mostly insensitive to the ordering of the points in the database. This algorithm is efficient even for very large spatial databases, discovers clusters of arbitrary shape and does not need to know the number of clusters in the data a priori, as opposed to K-means.

References

1. Bailey, T.C., Gatrell, A.C.: Interactive Spatial Data Analysis. Addison Wesley Longman, Edinburgh (1996)
2. Cressie, N.A.C.: Statistics for spatial data. John Wiley & Sons, London (1993)
3. El-Sonbaty, Y., Ismail, M.A., Farouk, M.: An efficient density-based clustering algorithm for large databases In: Proceedings of the 16th IEEE International Conference on Tods with Artificial Intelligence (ICTAI) (2004)
4. Ester, M., Kriegel, H.P., Sander, J., Xiaowei, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceeding of the 2nd International Confererence on Knowledge Discovery and Data Mining, pp. 94–99 (1996)
5. Fayyad, U., Piatesky-Shapiro, G., Smyth, P.: From Data Mining to Knowledge Discovery in Databases (1996). http://www.kdnuggets.com/gpspubs/aimag-kdd-overview-1996-Fayyad.pdf
6. Han, J., Kamber, M., Tung, A.K.H.: Spatial Clutering Methods in Data Mining: A Survey (2001). ftp://fas.sfu.ca/pub/cs/han/pdf/gkdbk01.pdf
7. Jan, A.K.: Data Clustering. 50 years beyond K-means. Pattern Recognition Letters, 651-666 (2010)
8. Koperski, K., Han, J., Adhikary, J.: Mining Knowledge in Geographical Data (1998). ftp://fas.sfu.ca/pubcs/han/pdf/geo_survey98.pdf
9. Mao, J., Jain, A.K.: A self-organizing network for hyper-ellipsoidal clustering (HEC). IEEE Trans. Neural Networks, 16–29 (1996)
10. Sander, J., Ester, M., Kriegel, H.P., Xiaowei, X.: Density-Based Clustering. from in Spatial Databases: The Algorithm GDBSCAN and its applications (1999). http://www.dbs.informatik.uni-muenchen.de/Publikationen/

11. Schoier, G., Borruso, G.: A clustering method for large spatial databases. In: Laganá, A., Gavrilova, M.L., Kumar, V., Mun, Y., Tan, C., Gervasi, O. (eds.) ICCSA 2004. LNCS, vol. 3044, pp. 1089–1095. Springer, Heidelberg (2004)
12. Schoier, G., Bato, B.: A modification of the DBSCAN Algorithm in a Spatial Data Mining Approach. In: Meeting of the Classification and Data Analysis Group of the SIS: CLADAG 2007, pp. 395-398 . Macerata: EUM, Macerata, settembre, 12-14, 2007
13. Steinbach, M., Ertöz, L., Kumar, V.: The Challenges of Clustering High Dimensional Data (2003). http://www-users.cs.umn.edu/~kumar/papers/high_dim_clustering_19.pdf
14. Xu, R., Wunsch II., D.: Survey of Clustering Algorithms (2005). http://ieeexplore.ieee.org/iel5/72/30822/01427769.pdf
15. Bedard, Y.: Beyond GIS: Spatial On-Line Analytical Processing and Big Data, Univ. of Maine (2014). http://umaine.edu/scis/files//09/Beyond-GIS-Spatial-On-Line-Analytical-Processing-and-Big-Data-Yvan-Bedard.pdf
16. Chen, Y., Suel, T., Markowetz, A.: Efficient query processing in geographic web search engines. In: SIGMOD 2006, Chicago, Illinois, USA, June 27–29, 2006. http://cis.poly.edu/suel/papers/geoquery.pdf
17. Worboys, M., Duckham, M.: GIS.: A Computing Perspective. CRC Press, Boca Raton (2004)