Recursive data clustering through finding vague solutions

Domenico Monaco[1]

monaco.d@gmail.com
http://domenicomonaco.it

May 7, 2019

Abstract

This work is developed over the question "How to automatically create a good clustering on spatial dataset with hight different local densities?" opened by previus work of Berzi.

To answer the main question, this work describe a approach of recursive clustering process based on a technique of finding "vague-solution", where the output is an hierarchical clustering of initial dataset. In particularly the the approach is developed and tested on DBSCAN [8] algorithm with large dataset gathered by Google Place in Metropolitan Area of Milan.

The core solutions developed in this algorithm are condensed in the capacity of generation a Hierarchical Clustering with a recursive select the best solutions in according to the our goals, previously dfined by some sets of rules.

The algorithm described here, and developed in my Master Thesis, rosolve two problem:

- When we use an algorithm of clustering that can create a set of differents clustering, but equally valid and don't know exctly what we must have as good solution, we are led to ask ourselves: "What are the better?" This obviously depends by our goals, so now the question is: "What are our goals?"
- The second problem is condesned in the sentence "Not all clusters can be found in one-shot clustering process, more often we must reapply the process to some part of datataset", so there we have a second question that this paper answering: "How to create clustering of data with ad-hoc processing for each different part of input dataset?"

These questions are resolved by the approaches namend in this work as: Space of Solutions, Vague-Solution, Vague-Solution finding Method and finaly Recursive Clustering. All of these approach was drafetd and testes algoruthm in mine Master Thesis titled "Geospatial data analysis for Urban informatics applications: the case of the Google Place of the City of Milan" [22].

Keywords: clustering, vague neighbourhoods, geospatial data, hierarchical, recursive

Contents

1 Introduction

Clustering can be understood as a synonym for unsupervised learning. It is generally used to discover classes (also called clusters) of belonging within datasets. [12]

Clustering is an essential component for Data Mining and Machine Learning. There are different definitions of Clustering, but in general it is an automatic unsupervised process that allows the grouping (or partitioning) of the items of a dataset through the similarity of the characteristics, measured through specific similarity formulas. [12] [11] [9] [6]

In the literature several Clustering techniques are presented which can often overlap each other, for this reason it is difficult to provide a single taxonomy that characterizes them in their entirety. [12].

One way to classify Clustering methods is to divide them according to the type of reorganization of the dataset that they are able to carry out (figure 1), in this sense the Clustering methods are divided into two main categories, Hard Clustering and Fuzzy Clustering (or Soft Clustering).

In the case of hard clustering, an item belongs to only one cluster, while in the case of fuzzy clustering it can belong to more than one cluster. [11].

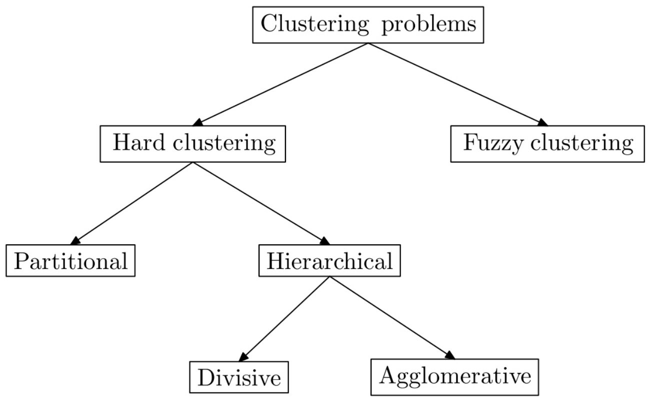

Figure 1: Clustering classfication provided by Gan et al. [11]

Among the most studied types of Clustering are Partitive Clustering and Hierarchical Clustering. In turn, Hierarchical Clustering can be subdivided into Agglomerative (bottom-up) and Divisive (top-down). [11]

Another ways of categorizing Clustering algorithms is the one proposed in the work [19] which divides them according to the type of approach used to reorganize the dataset, dividing them into:

- Probabilistic models
- Generative models
    - Density based
    - Grid based
    - Distance based
        - Flat

    - Hierarchical
        - Agglomerative
        - Divisive

1.1 Spatial Clustering

Spatial Clustering is the branch of Clustering methods that treats spatial data, it overlaps and partly inherits the classic Clustering techniques, but with some differences due to the specific field of application.

The most cited categorization in the literature of Space Clustering is that which follows that of Hard Clustering methods, subdividing them into:

- Partitive
- Hierarchial
    - Agglomerative
    - Divisive

Another classification of spatial Clustering algorithms that does not replace the previous one, but integrates it by emphasizing more specific features of spatial data is the following:

- Distance based [19]
- Grid Based [12] [10]
- Density based [12] [10]
- Based on local properties [17][5]
- Distribuitiion based [17]

Even in the spatial Clustering there are often overlaps and interoperability between categories due to the joint use of different techniques (figure 2) according to which the Clustering algorithms can be grouped into three macro groups: Partitioning, Hierarchical and Locality. [17]

I personally find this last classification very interesting because on the one hand it preserves the classical and widely shared taxonomy of hierarchical and partitious algorithms, but at the same time emphasizes a new category relating to local properties.

1.2 Density-based Clustering

According to the categorization of the figure 2 the density-based Clustering algorithms are part of the Locality-Based category where the items are grouped according to local relations, where in this case a particular type of local relation is density.

Density-based algorithms are characterized by their ability to identify arbitrary-shaped clusters, defined as regions with high density of arbitrarily distributed points, well separated by low-density regions (figure 3). These algorithms are able to efficiently manage noise and the presence of anomalous points and it is not necessary to know in advance the number of Clusters to be identified, thus lowering the degree of knowledge of the domain. [11] [12] [17] [18] [7]

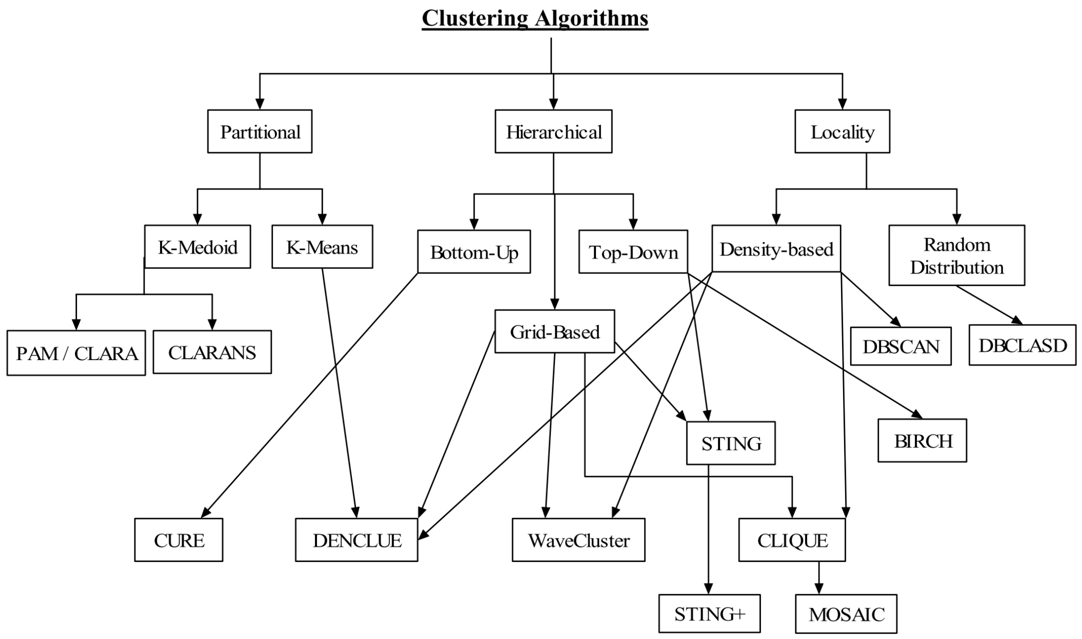

Figure 2: Classification of different Clustering algorithms [17]

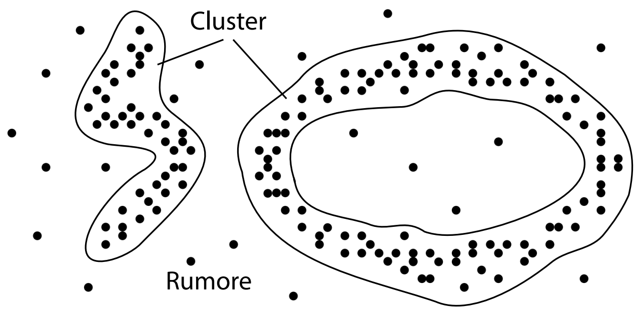

Figure 3: Cluster of high-density points and low-density noise [12]

A further characteristic is given by the fact that the identified Clusters can be both convex and concave, which are not able to do equally well the algorithms based on centroids that are more suitable instead to identify tendentially convex form Clusters. [12]

One of the critical points of the density-based clustering algorithms lies in the choice of the algorithm configuration parameters. [11]

Density-based methods are subdivided into Partitives or Hierarchies and most density-based approaches have been developed specifically for Spatial Clustering, density being a typically spatial property. [11] [12]

In according to [19] [12] [11], some of the most popular density-based clustering algorithms are: DBSCAN [7], DBCLASD, OPTICS [2], DENCLUE [13], BRIDGE [11], CUBE.

Among them the most popular is the DBSCAN [7] which according to most literature is considered the first density-based algorithm to have been introduced. [15]

While according to the work [24] it emerges that the density based Clustering algorithms have already been explored by Wishert in 1969. [28]

DBSCAN can boast numerous alternatives and variants representing one of the most relevant approaches based on density in Spatial Clustering. [21]

1.3 DBSCAN

DBSCAN is the density-based clustering algorithm that groups space points into arbitrary high-density clusters separated by low-density areas. [7]

DBSCAN is considered the first algorithm linked to the concept of density, through which the concept of density for spatial clustering was also presented. [21] [7]

DBSCAN is based on concept of density that in this case is defined as the ratio between the area circumscribed in a circle of radius r and of center in point p_0, and the number of points $p_1, ..., p_n$ present within it.

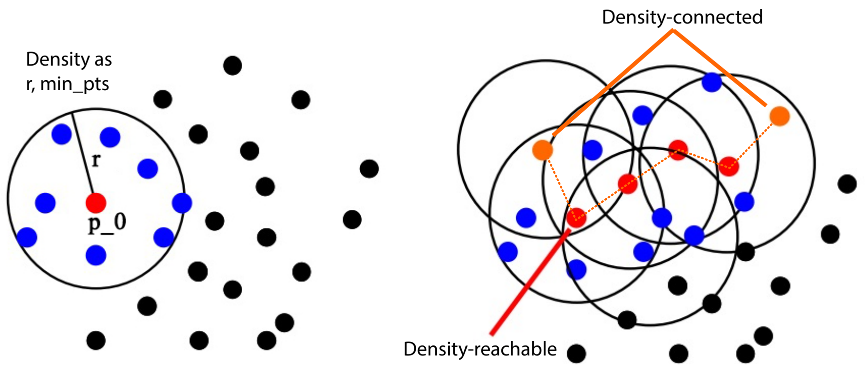

Figure 4: Density defined by Eps (r) and MinPts, Density-Connected and Density-Reachable properties [27]

Based on this concept of density, DBSCAN performs clustering through two main processes [7]:

- `DBSCAN()`: takes care of scrolling through the entire list of points to identify the Core-Point (see figure 4);
- `ExpandCluster()`: is referred to by the previous one and deals with expanding the Core-Points by searching for points that enjoy Density-Reachable and Density-Connected properties (see figure 4);

1.3.1 Limitations and advantages

The quality of Clustering algorithms is generally measured in terms of robustness, efficiency (time and memory), accuracy and amount of prior knowledge of the domain required for a good application [29]. Similarly, the quality of DBSCAN is measured in these terms. In generale, alcune varianti di DBSCAN sono nate per risolvere o migliorare alcune sue caratteristiche.

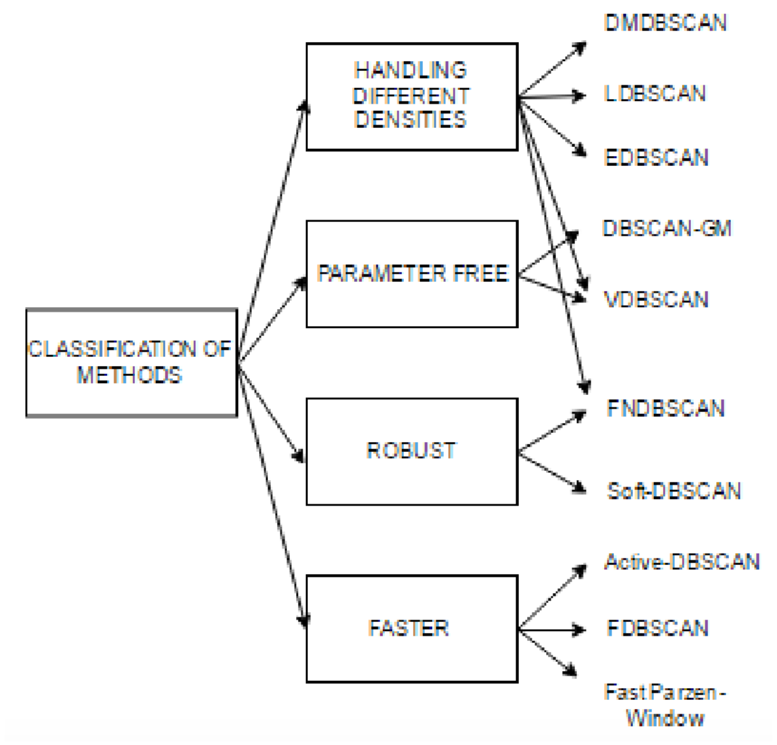

Figure 5: Comparison of DBSCAN variants provided by Yaşar and Ulutagay [29]

Limitations From the literature several critical points of the DBSCAN emerge, they are basic motivations for the DBSCAN variants proposed in the literature (see figure 5). [15] [1]

The most cited limitations are:

- The complexity is equal to $O(n^2)$ without optimizations for large datasets; [7];
- Limited efficiency in identifying Clusters datasets with very different densities; [20]
- High memory consumption; [7]
- DDifficulty in choosing the optimal combination of user-input parameters Eps and $MinPts$; [25];

Advantages DBSCAN is the most popular algorithm among spatial clustering approaches, this is mainly due to its ability to identify arbitrary-shaped clusters, while other advantages can be:[16] [1]:

- It is not necessary to know the number of Clusters in advance;
- The ability to efficiently identify noise and outliers;

2 Research questions

After sintetic overview of spatial and density-based clustering we can define the main reserch question. My master thesis, and so also this work, continue the works of [3] by trying to solve the main problems of using DBSCAN on geospatial data crawled by Foursquare on Milan areas. Some od these problema re directly derived by DBSCAN limitations and in general of data clustering approaches, detailed in the previous sections.

2.1 Mining the Social Media Data for a Bottom-Up Evaluation of Walkability

The main goals of paper [3] is try to measure the "Walkability" [26] of the urban areas over Milan city, but to make this for first they must identify and build the entities of analysis.

The approach used to identify the entities within the city, analogously as [14], is based on the application of a clustering technique on social media data.

The algorithm chosen to identify the entities within the city: it allows to identify and delineate regions of high density that are separated by regions of lower density is the DBSCAN (Density-Based Spatial Clustering of Applications with Noise). This over the intuition that "around areas of interest there are regions separating them from potentially nearby other areas, so a sort of border can be identified around the area."[3] [8] [7]

But, as explained in [3], the heterogeneous density that characterizes Milano does not allow to apply a one-shot algorithm that covers the entire city.

Their solution is based on manual iteration and regulation of DBSCAN to search empirically the good parametrization of each area of interest, trough visual evaulation.

The is the core of my research question, detailed in next section.

2.2 New input dataset: Google Place POI

To adding value of initial work [3] I've gathered a new dataset of geospatial data by Google Place with a specific algorithm developed for the Master Thesis: "An adaptive agent for Google Place crawling" [23].

2.3 Parameterization, recursion and zooming

Parameterization and manual iteration and evaluation are the core limitations of Berzi work.

Parameterization For first, Berzi brings out that it can't decide a priori the exact parametrization of DBSCAN algorithm that able to cluster in right way the dataset, in simple word it don't know the parametrization for general good one-shot clustering of Milan city.

Iteration For second, let's imagine we know the correct parameterization for the city of Milan, this is obviously not correct for all sub areas of milan. In other words every area of Milan needs a different parametrization of DBSCAN. Berzi name this "iteration", but the "iteration" is not the properly solution of this problem because they are not consedered the zooming problem, detailed in next paragraph.

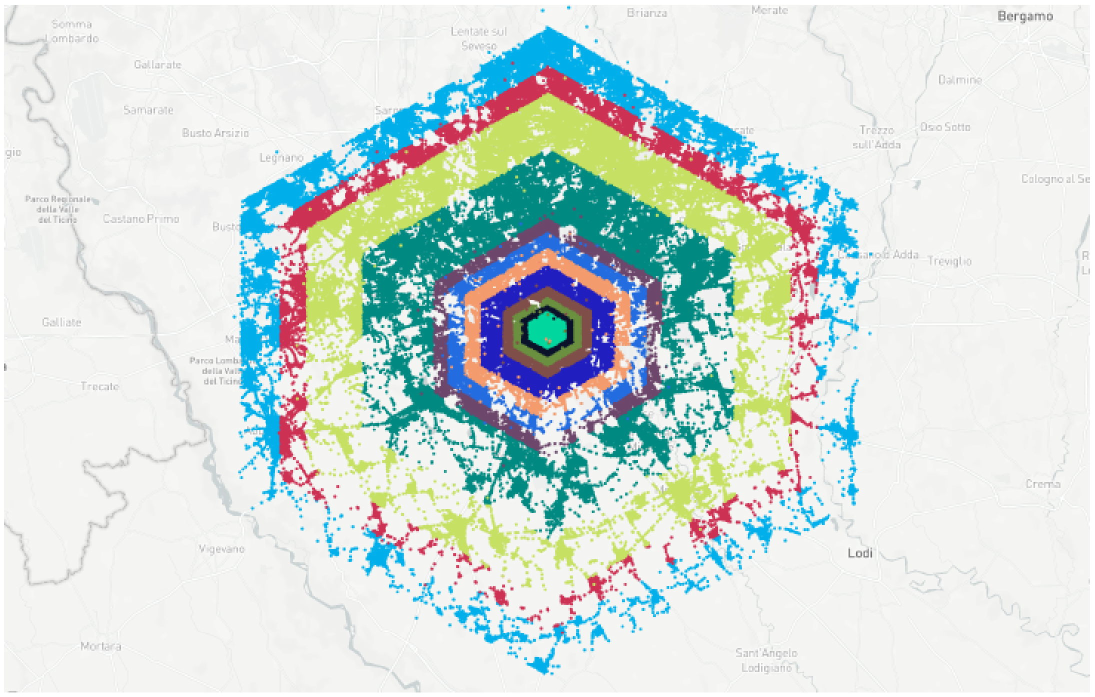

Figure 6: Points gathered with the adaptive agent over Google Place over Milan area; the colors are referred to differents steps of crawling; [23]

Zooming and recursion For third, in general at different level of zoom of analysis of an urban area, you must different level of clustering. It's probably that you must bigger and coarse clustering when you looking a big urban area, and in opposite you must a more fine clustering when you zooming to some urban details. This incorporate the problem named "iterartion" by berzi.

I prefer to refer to this problem as "recursion" and "zooming", where some cluster must be reclusterd as single dataset to see some sub clustering.

Automation After defined these basics features of our spatial clustering based on density, it's important don't forgot that we must have sufficient automation of these operations to have applicable approach.

3 Algorithm in detail

These are main problems and limitations of previous works that they fed initially my Master Thesis, and in more detail the approach and algorithm described in this paper.

My solution that try to solve the previous problems are composed by this next concepts, developed in this case over DBSCAN algorithm::

- Space for solutions of Clustering algorithm;
- Definition of vague goal solution and relative method of search;

- Recursion of clustering process and relative rule of termination;

These are the three basic concepts of Recursive data clustering with finding Vague-solutions, that in this case was developed and applied over DBSCAN on geospatial data, these gathered by Google Place API with ad-hoc algorithm. [23]

In the next sections i try to describe the technique developed on DBSCAN, but in don't exclude the possibility of an its future generalization.

3.1 Space of solution and solution features

Space of solution is the basic concept of "Recursive data clustering through finding vague solutions" by are deriveds all next concepts ad approaches.

Space of Solutions Space of solutions is the set that contains all possible outputs of a specific clustering algorithm when you hold fixed a input dataset.

In other word, if you hold fixed a input dataset, you can have differents clustering output of a the same dataset that depends uniquely by variation of parameters of clustering algorithm, by using all possible combination of parameter you can ideally form all possible solutions of algorithm. This set is the "Space of Solutions".

Set of Clustering In the case of DBSCAN the output is an specific Clustering of input dataset and the input parameters, so we can see the Space of Solutions as "Set of Clustering". In the next sections we will use "Set of Clustering" as synonomous of "Space of Solutions".

Def. Set of Clustering Fixed an input dataset D and a process of DBSCAN to which we supply a set of values $Eps = \{\epsilon_1, \epsilon_2, ..., \epsilon_e\}$ and $MinPts = \{\omega_1, \omega_2, ..., \omega_m\}$ with $\omega_m \in N$, $\epsilon_m \in R$. Its possible to launch iteratively the DBSCAN to all couple of values ($\omega_m \in MinPts$, $\epsilon_m \in Eps$).

So we have $DBSCAN(D_{fixed}, \epsilon_e, \omega_m)$ that enables us to have $SetOfClustering()$, where this last is defined as equation 1:

$$SetOfClustering_{(D,Eps,MinPts)} =$$

$$\{C_{i,j} \mid C_{i,j} = DBSCAN(D_{fixed}, \epsilon_i, \omega_j)\ \forall \epsilon_i \in Eps, \omega_j \in MinPts\} =$$

$$= \begin{bmatrix} C_{1,1} & C_{1,2} & \dots & C_{1,m} \\ C_{2,1} & C_{2,2} & \dots & C_{2,m} \\ \vdots & \vdots & C_{i,j} & \vdots \\ C_{e,1} & C_{e,2} & \dots & C_{e,m} \end{bmatrix}$$

$$Equation:\ SetOfClustering\ definition \tag{1}$$

In figure 7 we can se three hypothetical solutions of set of clustering $C_{5,2}, C_{7,5}, C_{7,8}$ they will be three different solutions of the same dataset D with different values $\epsilon_i \in Eps, \omega_j \in MinPts$, each clustering has potential validity in according to the goals of analysis.

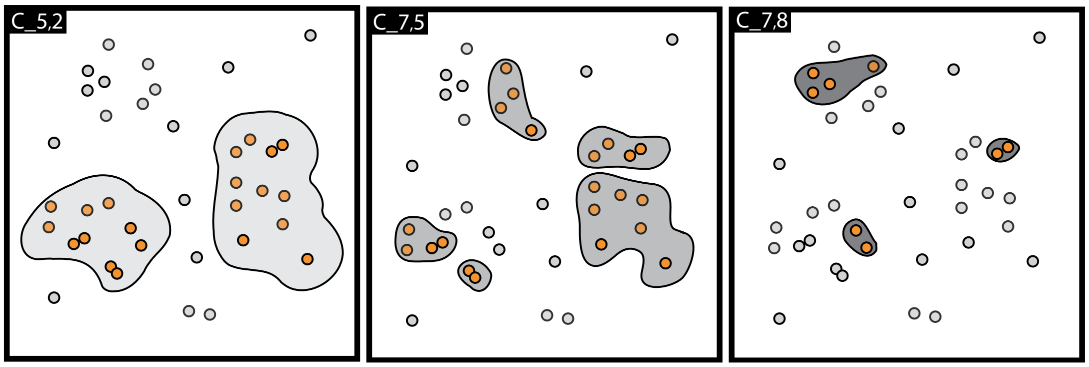

Figure 7: Different Clustering solutions for the same dataset taken from the Clustering set. Note: This image is for illustrative purposes only. The real outputs are too complex to be used at this point of paper.

3.1.1 Granularity and size of Space of Solutions

Computationally is not possible to compute all clusterings outputs because sometimes, as in the case of DBSCAN, the algorithm have have continous values as parameters or in other case some parametrization produce a too hard computation for the specific used hardware.

This is partially solved by varying the granularity of input set of parameters, this allow us to predicts the maximum number of outputs and their distance among them.

In the case of DBSCAN (equation 1)the maximum number of outputs present in a $n \times m$, so the complexity of building a SetOfClustering is $O(n \times m) \approx O(i)$, by ignoring the complexity of a single DBSCAN computation.

Granularity of Space of Solutions So, in more precisely way, fixed D, which and how many solutions $C_{i,j}$ we can have in the $SetOfClustering$ depends on the values present in the sets $Eps, MinPts$, in fact the same matrix $SetOfClustering$ has size $n \times m$ which depends by numerosity of $|Eps| = n$, $|MinPts| = m$.

With this concepts it's possible to build a low or a high granular $SetOfClustering$ by varying the number of values in the $Eps, MinPts$, for example considering two possible ranges of values $Eps_h, MinPts_h$ and $Eps_l, MinPts_l$ where:

$$|Eps_l| < |Eps_h|,\ |MinPts_l| < |MinPts_h|$$

Two clusters of different granularity are obtained:

$$SetOfClustering_{(D,Eps_l,MinPts_l)} =$$

$$= \begin{bmatrix} C_{1,1} & \dots & \dots & C_{1,m+r} \\ \vdots & \vdots & C_{i,j} & \vdots \\ C_{e+p,1} & \dots & \dots & C_{e+p,m+r} \end{bmatrix}$$

$$SetOfClustering_{(D,Eps_h,MinPts_h)} =$$

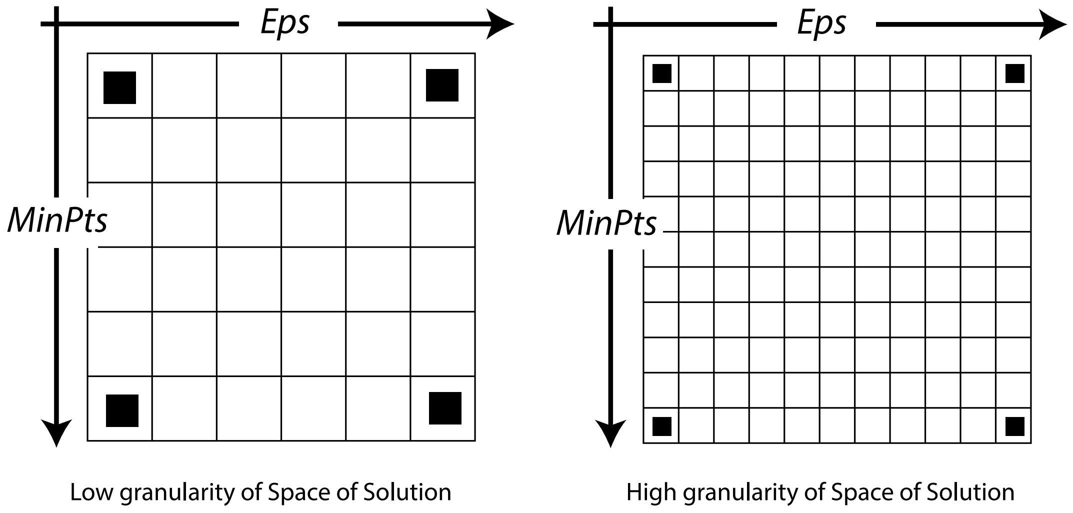
Eps
MinPts
Low granularity of Space of Solution
Eps
MinPts
High granularity of Space of Solution

Figure 8: Two ideal matrix $SetOfClustering$ with different ganularity of input parameters but with same extreme values (Black Squares).

$$= \begin{bmatrix} C_{1,1} & \dots & \dots & \dots & \dots & C_{1,m+u} \\ \vdots & \vdots & \dots & \dots & \dots & \vdots \\ \vdots & \vdots & \dots & \dots & C_{k,z} & \vdots \\ \vdots & \vdots & \dots & \dots & \dots & \vdots \\ C_{e+t,1} & \dots & \dots & \dots & \dots & C_{e+t,m+u} \end{bmatrix}$$

This allows you to choose the degree of precision with which to generate the $SetOfClustering$. This, by balancing computational complexity, accuracy of the result and the number of solutions.

How to and how many values in the Eps, $MinPts$ depends by specific functions that enables us to have the exact list of values that we must have in according to our complexity and performance restrictions.

3.1.2 Feature of Solutions

At this points we have a Set of Clustering that represent a Space of Solutions of our specific input dataset over variation of parameters of DBSCAN algorithm.

We need to differentiate every single Clustering outputs by computable values to automate the selection of better solutions, because the humans can't evaluate hundreds clustering by visual evaluations (see evaluation of clustering in [3]).

Therefore every single Clustering is correlated to a set of features as a measurable properties (numerical o verbatim) that characterize it.

The union of one Clustering and its features produce our Solution (of Clustering), that practically enables us automate the: searching, comparison, evaluation and choice, of a specific Clustering. Because the Solution is an "understandable" Clustering.

In my master thesis i have used the name "Indicators" to name these features, in according to the ability of "indicate" the quality of specific clustering.

I'm sorry for continuous changing names, but for continuity to my master thesis some times I must used originals names.

Idicators of Clustering An indicator is like a feature, but more crude and limitated.

In my work at the current state of work an indicator is a numerical value that characterize a specific cluster.

So, given a specific Set of Clustering

$$SetOfClustering_{(D,Eps_l,MinPts_l)} =$$

$$= \begin{bmatrix} C_{1,1} & \dots & \dots & C_{1,r} \\ \vdots & \vdots & C_{i,j} & \vdots \\ C_{e,1} & \dots & \dots & C_{e,r} \end{bmatrix}$$

We can produce some indicatores $I_x()$ for each $C_{i,j}$ Clustering that are present in Set of Clustering. This enables us to build a single solution $s_{i,j}$ of Clustering $C_{i,j}$ defined as:

$$s_{i,j} = \{C_{i,j}, I_1(C_{i,j}), .., I_x(C_{i,j}), ..., I_s(C_{i,j})\}$$

More precisely an indicators must be calculated in some ways, so an indicator i_X is directly related to its funciotn $I_x()$ that produce it, where the input is the clustering an $C_{i,j}$. So we can use as synonimous $I_x(C_{i,j})$ and i_x

Developed Indicators Based on a fact that $C_{i,j}$ is a set of Cluster $\{c_0, ...c_y\}$ we can define few indicator $< I_l, ...I_g >$ for eacy Clustering. In this work was developed the follow examples of Indicators (see figure 9):

- Number of Clusters : $I_0 = |Clustering| - 1$;
- Number of points not flagged as noise: $I_1 = \frac{|c_1|+|c_1|+\dots+|c_y|}{|c_1|+|c_1|+\dots+|c_y|+|c_0|}$
- Avg of numbers of point present in each cluster: $I_2 = \frac{|c_1|+|c_1|+...+|c_y|}{|Clustering|-1}$
- Number of points of bigger cluster $I_3 = max(|c_1|, |c_2|, ..., |c_y|)$
- Number of points of smaller cluster: $I_4 = min(|c_1|, |c_2|, ..., |c_y|)$
- Standard deviation of number of point present in each clusters: $I_5 = std(|c_1|, |c_2|, ..., |c_y|)$
- Number of point present in a first 3 bigger clusters: $I_6 = |c_h0, c_h1, c_h2|$ where c_h0, c_h1, c_h2 are the first 3 bigger clusters;
- Number of Clusters that have a number of point less the avg of Clustering: $I_6 = |c_{under\_avg}|$, $c_{under\_avg} = \{c_k \ t.c \ |c_k| < I_2\}$

These are only developed and used Indicators in my Master Thesis, but is possible to have more indicators that characterize more precisely each cluster.

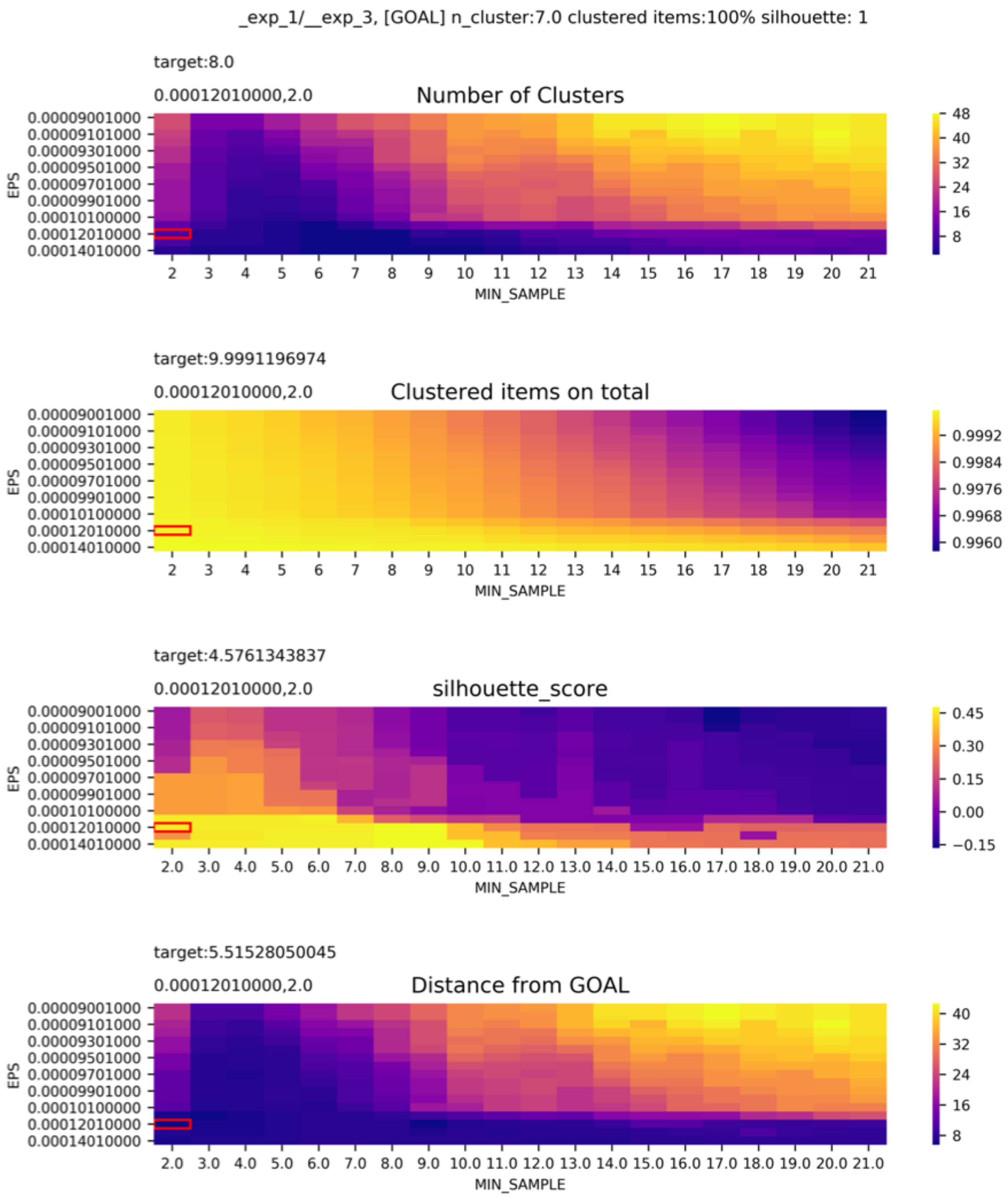

Figure 9: Real example of charts that represent different indicators over same Set of Clustering

3.2 Vague-solution and finding method

Now, we must select the best clustering in this space. But, as how we talked about previously we don't know what we want as solution and in addition the exact solution may not exist, this because the granularity that we selected is not sufficiently high or simply the exact solution really don't exist.

In this case we can't simply search "the solution", we must be content with find the solution closest to what we are looking for.

The term of "vague-solution" is derived by [4] work, but in this work is used in some differents ways.

Vague indications in real life Its like we we going to shoes shop to buy "running shoes" but we dont know what they have in a store and if they have what we want. In this case our behaviour is characterized by a general desctiption of shoes such as colors, size and use of thedesired shoes. So, the salesperson use this "vague indications" to search a best "running shoe" for us. This is the work of salesperson, the succes of "search" dependes of ability of processing of "vague indications" by salespersons.

Vague-solution This is an example of a Vague-solution in real life. In a context of this work the "Vague-solution" is a set of features that can be used to find the more closest solution in according our goals.

But, how we talked previously we can't simply search a solution because can be doesn't exist the solution.

For this reason in accordly to the "Vague-solution" we must define "Vague-solution finding Method" that enables us to select the more closest solution.

"Vague-solution" values and "Vague-solution finding method" are directly linked because the features that we choice to use impose the computation method that we can use with them.

Vague-solution can be viewed as set of "Goals Indicators" definded as:

$$s_{goal} = \{I_{goal_0}, I_{goal_1}, ..., I_{goal_k}\}$$

that represent the input parameters for a specific Vague-solution finding Method.

Vague-solution finding Method Specific Vague-solution finding method is a function defined as

$Select_q(S, s_{goal}) \xrightarrow{select} s_{i,j}$ where $S = \{s_{0,0}, ..., s_{i,j}, ..., s_{e,r}\}$ and $s_{i,j} = \{C_{i,j}, I_1(C_{i,j}), .., I_x(C_{i,j}), ..., I_s(C_{i,j})\}$

where s_{goal} and $s_{i,j}$ can be are not exactly equal, but $s_{i,j}$ in according to $Select_q$ implemention is the more closest solution to s_{goal}, as discussed previously.

Example of Vague-solution finding method In the context of my Master Thesis have been developed four different Vague-solution finding method, but after some empirical checks only one was used (see figure 10).

Vague-solution finding Method n.3 The used solution aims to choose that Clustering with a considerable number of points that are not marked as noise, and at the same time we did not want the presence of super-huge clusters that engulfed the others in terms of number of points and size.

This is idea hav been developed in Vague-solution finding method that aims to select that clustering which, ignoring "the first bigger cluster", is characterized by the "next first 6 more numerous clusters" and at the same time the number of points that are not marked as noise must be are over 87% (see figure 11).

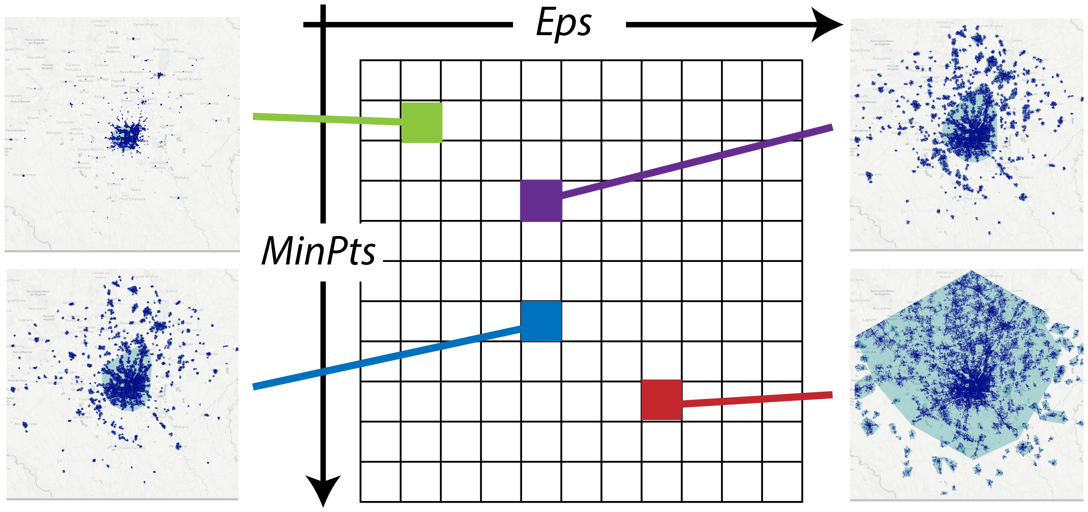

Figure 10: Illustrative figure of four different outputs of Vague-solution finding Method developed for the Master Thesis [22]; The matrix and the position of Clustering inside it is for illustrative purpose, but the clustering are the real output of four methods;

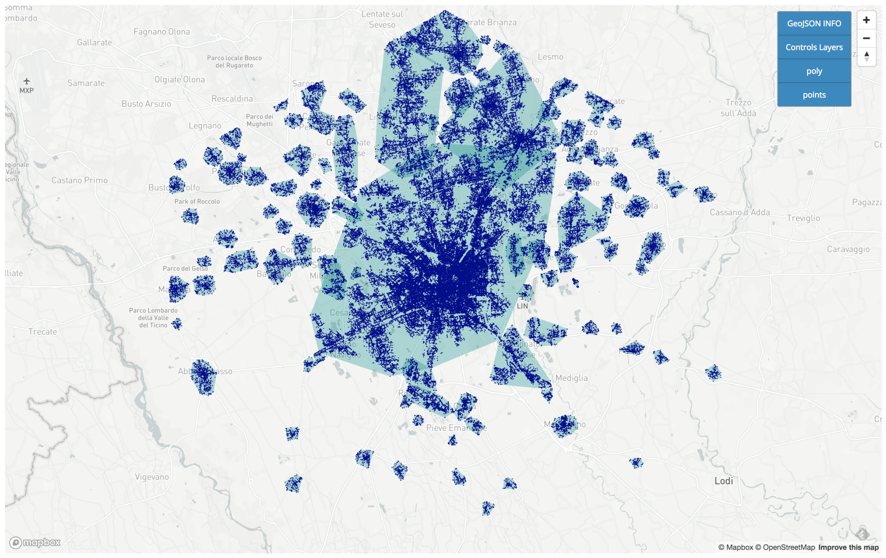

Figure 11: First level of Clustering selected by Vague-solution finding Method n.3; The apparently overlapping of clusters is caused by convex shape of polygons, this effect can be solved with a convex polygons visualization;

This Clustering is consedered by me as good clustering at the first level of observation of Milan area. But is clearly not good for more high detail of observation such as specific road or area. For this reason i developed the recursion of this approach to detail, in a same ways of first, every area of city. The recursion is described in the next sections.

3.3 Recursion and Hierarchical Clustering output

Now, in the previous sections we have processed the first level of space of solution (in this case Set of Clustering), and we obtained the desired solution trought Vague-solution and linked finding method to it.

But, the other question of Master Thesis is: build and select hierarchically the good clustering for each area of city.

This goal is resolevd by "recursion" over clustering process that ables us to build tree of clusters. Therefore the output of recursive clustering is an hierarchial clustering, based on Vague-solution finding method.

In simple word, the approach and algorithm based on Space of Solution was is applyed recursively by starting a single input dataset (in this case 290000 points over Milan area).

3.3.1 Recursive clustering

The idea of "Recursive clustering" is based on the fact that if a clustering process's output or a part of its, it's of the same type of requested input of clustering process, we can use this as new input recursively (see figure 12).

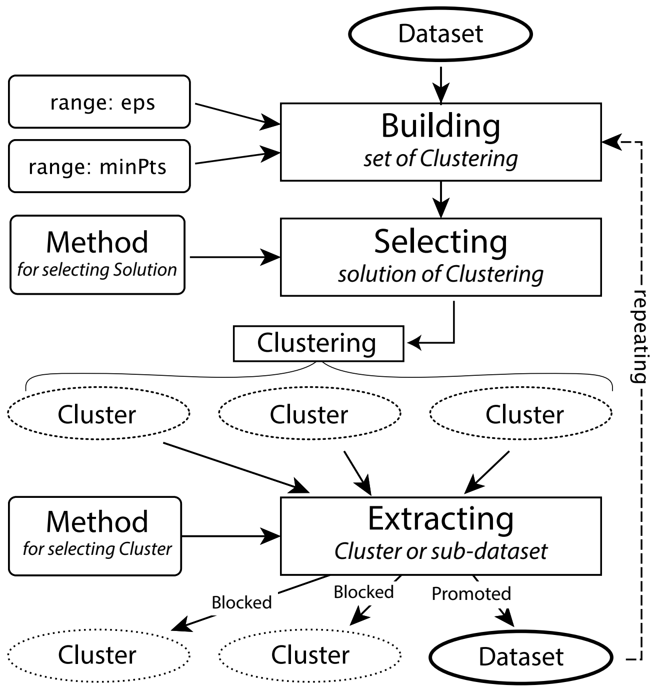

Figure 12: General overview of recursive data clustering with vague-solution finding developed over DBSCAN

Recursion is facilitated by the "Vague-solution finding method" that prevents blockage of recursion if the exact solution sought does not exist, because it is impossible to predict at levels subsequent to the first what can be found and what is not.

But is necessary a set of termination rules of recursion to prevent a looping situation or too deeply processing.

There fore the parametrization of "Recursive clustering" are:

- Set of Indicators that are used to promote Cluster as new input (Dataset) for the main clustering process (in figure 12 is named "Promoted" a Dataset"); These indicators are in "Extracting" process;
- Set of rules that stop the recursion; These are used to passing Dataset to "Building" process;

Normally the set of Indicators for promoting cluster as dataset can be sufficient to block the recursion but empirical observation pushed me to set some rules to prevent extreme looping, these added between "Extracting" process and "Building" process.

Recursion process To transform the linear process of Clustering to an recursive clustering I've used an FIFO queue that list the type of process that must executed and the link to the dataset (see figure 13).

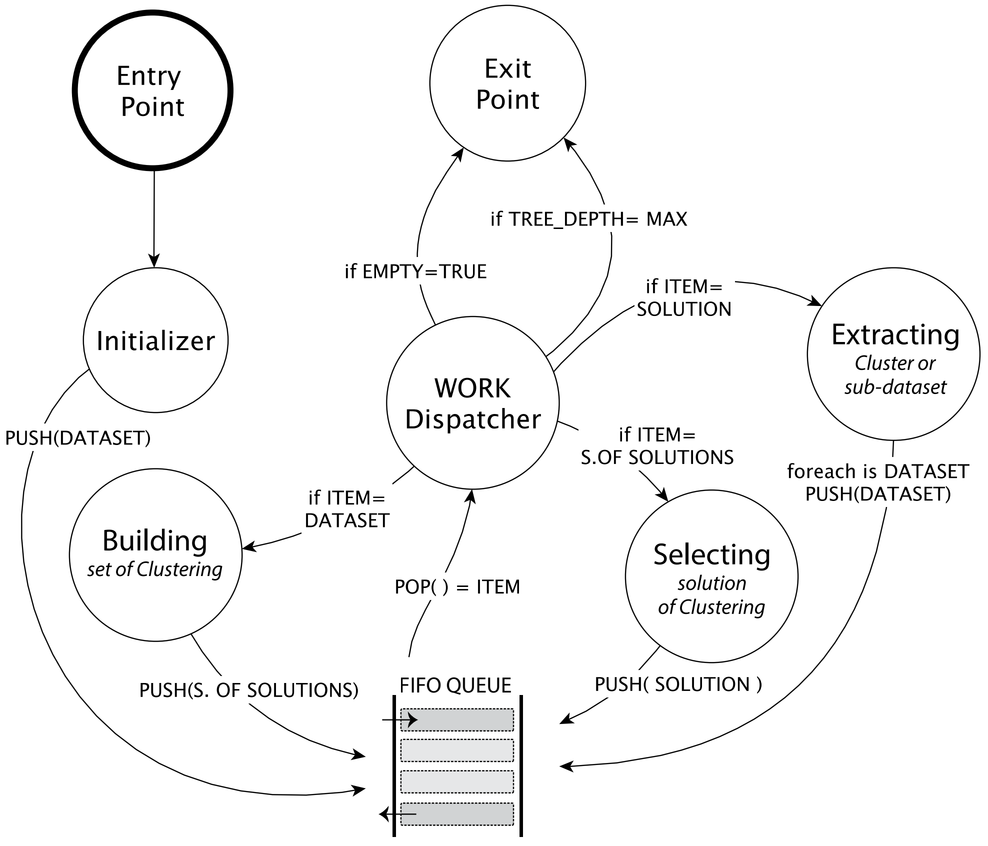

Figure 13: Representation of recursion implementation applied in the Recursive Data Clustering through finding vague solution

Simply, I've splitted the main linear process (see figure12) in three autonomous process: Building, Selecting and Extracting; these supported by an "Work dispatcher" and a FIFO queue.

The three main processes as defined as:

- $Build(Dataset_{input}, Eps, MinPts) = SpaceOfSolutions_x$
- $Selecting(SpaceOfSolutions_x, VagueSolution, VagueFindingMethod()) = Clustering_y$
- $Extracting(Clustering_y, RulesOfPromotingDataset) = \{Clusters, Datasets\}$ where $Clusters = \{C_0, ..., C_j\}$ and $Dataset = \{D_0, ..., D_i\}$; with $Clusters \cup Dataset = Clustering_y$ and $Clusters \cap Dataset = \emptyset$

The core of recursion consist in the autonomy of every processes, that are able to process and work in autonomy; the correct concatenation of recursion is guaranteed by Work Dispatcher and by FIFO queue.

3.3.2 Hirerchical Clustering

With these techniques we are able to build a tree of cluster based on a initial single input dataset, that are splitted in more-ad-more smaller dataset if needed. In according to the specific solution that I developed the tree of cluster have specific charateristic explained in this section.

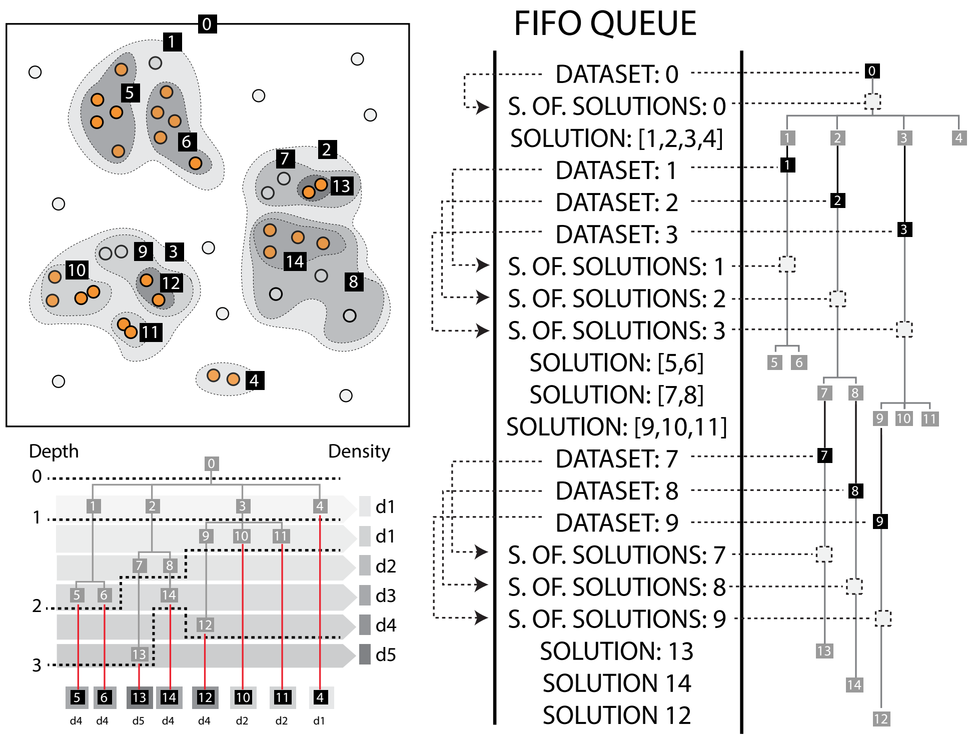

Figure 14: Figure for illustration purpose of Hierarchical Clustering obtained by Recursive Clustering with final leaf cluster selection (red lines)

Properties of Hirarchical clustering based on DBSCAN

- Every subtree has potentially different height;
- Every subtree has potentially different minuimum density;
- Every leaf clusters has potentially different maximum density;
- At a specific depth of main tree, all clusters not overlapping each other;
- Every sons of specific cluster can't be bigger than father or characterized by less density;

Leaf Clusters and history of clustering The leaf clusters are ideally our desired detail of clustering of initially dataset, composed by different cluster with different density and the same time with the low losing of points becase every leaf cluster are derived by previous clustered data; as exacample if a spefic leaf cluster is too small is possible push back and use the father cluster (see figure 14).

4 Conclusion and future work

To recap, the main question this work as continuity of the Berzi works is: "How to automatically create a good clustering on spatial dataset with hight different local densities?" Such as explained in previouse sections some of these problem are directly derived by DBSCAN limitations and in general of data clustering approaches.

The technique described here, and initially developed in Master Thesis, is splitted in two comprensive problems to be solved:

- In the case of DBSCAN clustering algorithm that can create a set of differents clustering potentially equally valids: What is the best? This is resolved by concepts of "Space of Solution", "Vague-solution" and "Vague-solution" finding Method";
- The second problem is described as "Not all clusters can be founded in a one-shot Clustering process, more often we must re-apply the process to some part of datataset". This is resolved by "Recursive Clustering".

These solutions was studied and developed into this work, producing the idea and draft alforithm of "Recursive data clustering with finding Vague-solutions", applied on 290000 Place gathered over Milan area and by using the DBSCAN clustering algorithm.

4.1 Hierarchical Clustering over Milan area

The output of "Recursive data clustering through finding vague solutions" is an Hierarchical Clustering of Milan Area based on POI's deinsity. Before clustering the datase, this, was cleaned by "road's name POIs" that Google create to indicate the name of roads, because these POIs not real place and are too regualrly positioned over geographial area; this extreme regularity of position can create wrong output of density clusteting.

The tree that is generated is composed by 5 level of depth with over 300 leaf clusters, with more than 25 places (in red, figure 15).

For each depth and clusters that was promoted as dataset was used the same:

- Granularity and value list of Eps and $MinPts$;

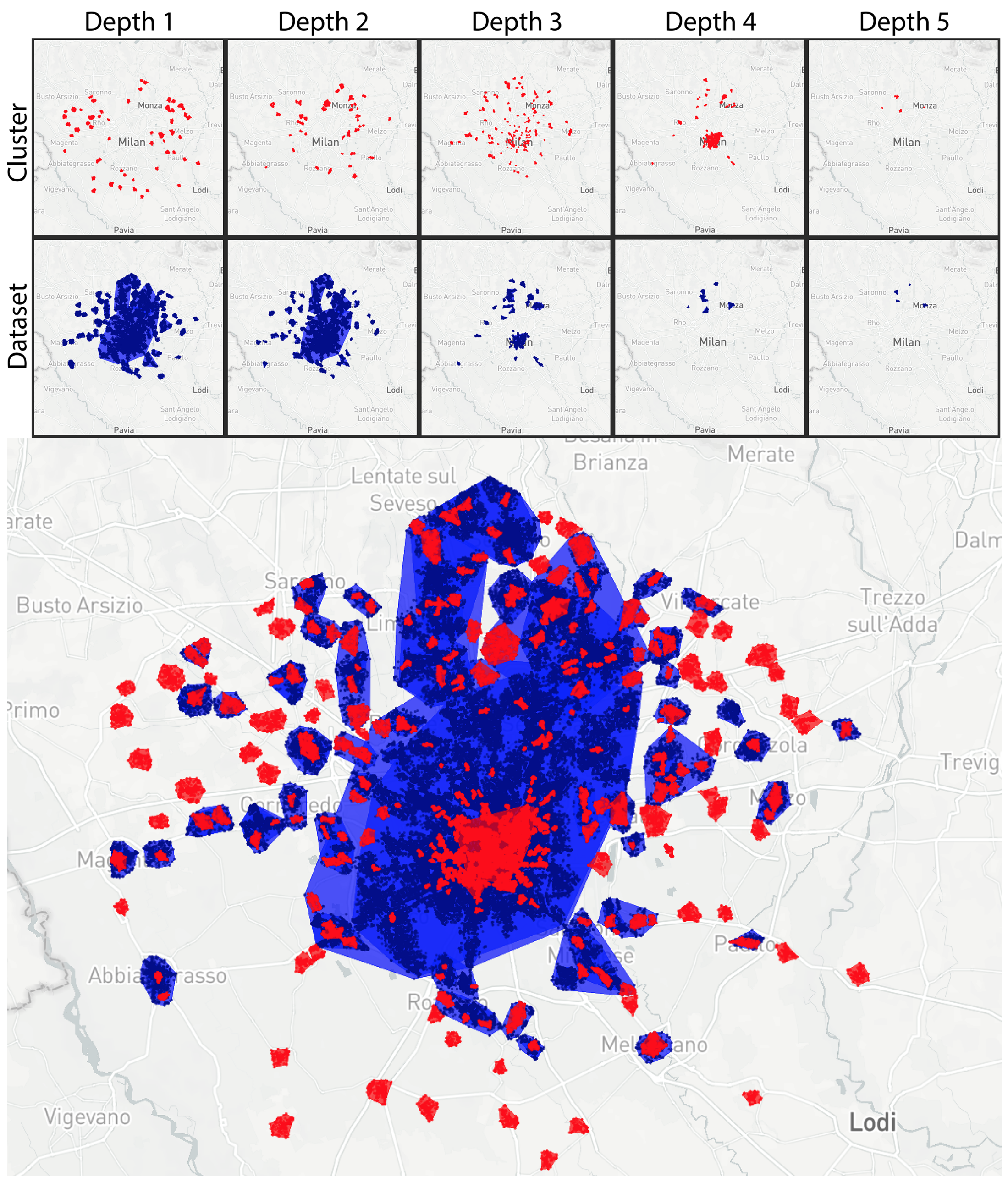

Figure 15: Figure for illustration purpose of Hierarchical Clustering obtained by Recursive Clustering with final leaf clusters selection (red lines)

- Vague-solution and Vague-Solution finding Method;
- Rules of Cluster promotion as Dataset;

What i found with this approach and algorithm represent: the most densely served areas by services over Milan (and nearby cities) with different local level of density.

In some cases are center of cities, in other case are neighborhoods, in other are roads and some special case are shopping centers. What this "entities" have in common is just initially explored by other work developed in a Master Thesis that are not explained in this work (too more details see [22]).

4.2 Strengths and Weaknesses

"Recursive data clustering with finding Vague-solutions" solve some problems opened by Berzi, but generate others.

Vague-Solution, in my opinion, is good technique when you don't kwno what you want and the precision of results is not needed. But is not good when you must have precision.

In addition the quality of Vague-Solution and Vague-Solution finding Method are related to what features or Indicators you have developed and the quality of finding Method.

In general is difficults to predict what granularity and size of input parameters must have before of complete processing.

Even if the DBSCAN is characterized by $O(n)$ complexity, with "Recursive data clustering with finding Vague-solutions" is possibile to predict the exact maximum execution of DBSCAN to each Dataset.

While it is difficult to predict how many cluster will be promoted to dataset, but this can be mitigated by more strongly rules in Vague-Solution and it finding method.

In general, in my opinion, the biggest problem of this algorithm are the much numbers of new input parameters created to solve the main research questions.

This can be resolved or tuned by an intelligent approach that sets all the input values, this in real time during the entire process. So we can have ad-hoc parametrization for each depth of tree or others different condition of input.

4.3 Future works

This algorithm must be tested with more attention also with different input datasets to enables us to find other strengths and weaknesses and in final to decide if is a powerfulapproach or only different approch of clustering. To make this, is necessary an ingegnerization of this algorithm to enable other people to use, test, debug and extends the algorithm. Other future works can found in a appliication of this algorithm in different topic research that can found usefull this approach.

Generalization of algorithm as general approach The algorithm desribed here is born on tipically problems of DBSCAN, and is developed on it. But, in my opinion some problems of DBSCAN are in common with different clustering algorithms, therefore i think this algorithm can be generalized as general approach and applicable over different algorithm with a same characteristics of DBSCAN.

For exaple, I can think to K-MEANS [12], powerful and fast clustering method for spatial data, that require only one parameter: the numbers of Cluster that must be generated. But, this method can't create a hierarchy of clusters and in case we don't know the numebers of clusters of our desired output we must launch the K-MEANS algorithm with empirical parametrization and evaluete every single outputs. This is clearly unusable on large problems.

In my opinion, the approach described here can be used to solve probelems as these also on differents clustering algorithm as the K-MEANS.

References

[1] Tariq Ali, Sohail Asghar, and Naseer Ahmed Sajid. Critical analysis of dbscan variations. In *Information and Emerging Technologies (ICIET), 2010 International Conference on*, pages 1–6. IEEE, 2010.

[2] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. In *ACM Sigmod record*, volume 28, pages 49–60. ACM, 1999.

[3] Andrea e Vizzari Giuseppe Berzi, Christian e Gorrini. Mining the social media data for a bottom-up evaluation of walkability. *arXiv preprint arXiv:1712.04309*, 2017.

[4] P. Brindley, J. Goulding, and M. L. Wilson. Generating vague neighbourhoods through data mining of passive web data. *International Journal of Geographical Information Science*, 32(3):498–523, 2018. doi: 10.1080/13658816.2017.1400549. URL https://doi.org/10.1080/13658816.2017.1400549.

[5] Lian Duan, Lida Xu, Feng Guo, Jun Lee, and Baopin Yan. A local-density based spatial clustering algorithm with noise. *Information systems*, 32(7):978–986, 2007.

[6] Richard C Dubes. How many clusters are best?-an experiment. *Pattern Recognition*, 20 (6):645–663, 1987.

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[8] Martin Ester, Hans-Peter Kriegel, Jrg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[9] Vladimir Estivill-Castro. Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75, 2002.

[10] BS Everitt, S Landau, M Leese, and D Stahl. *Cluster analysis: Wiley series in probability and statistics*. Wiley Chichester, 2011.

[11] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*, volume 20. Siam, 2007.

[12] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[13] Alexander Hinneburg and Daniel A Keim. A general approach to clustering in large databases with noise. *Knowledge and Information Systems*, 5(4):387–415, 2003.

[14] Yingjie Hu, Song Gao, Krzysztof Janowicz, Bailang Yu, Wenwen Li, and Sathya Prasad. Extracting and understanding urban areas of interest using geotagged photos. *Computers, Environment and Urban Systems*, 54:240–254, 2015.

[15] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and Sababady Sarasvady. Dbscan: Past, present and future. In *Applications of Digital Information and Web Technologies (ICADIWT), 2014 Fifth International Conference on the*, pages 232–238. IEEE, 2014.

[16] Slava Kisilevich, Florian Mansmann, and Daniel Keim. P-dbscan: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. In *Proceedings of the 1st international conference and exhibition on computing for geospatial research & application*, page 38. ACM, 2010.

[17] Erica Kolatch et al. Clustering algorithms for spatial databases: A survey. *PDF is available on the Web*, pages 1–22, 2001.

[18] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011. doi: 10.1002/widm.30. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.30.

[19] Vipin Kumar. *Data Mining and Knowledge Discovery Series*. Chapman & Hall/CRC, 2014.

[20] Peng Liu, Dong Zhou, and Naijun Wu. Vdbscan: varied density based spatial clustering of applications with noise. In *Service Systems and Service Management, 2007 International Conference on*, pages 1–4. IEEE, 2007.

[21] Jeremy Mennis and Diansheng Guo. Spatial data mining and geographic knowledge discovery—an introduction. *Computers, Environment and Urban Systems*, 33(6):403–408, 2009.

[22] Domenico Monaco. Analisi di dati geospaziali per applicazioni di urban informatics: il caso dei google place nella citt di milano. Master's thesis, Universita' degli Studi Milano-BICOCCA, 10 2018. URL https://easychair.org/publications/preprint/vRQ6.

[23] Domenico Monaco. An adaptive agent for google place crawling. 10 2018. doi: 10.13140/RG.2.2.34045.82406/1. URL https://easychair.org/publications/preprint/P1kN.

[24] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.

[25] Abir Smiti and Zied Elouedi. Dbscan-gm: An improved clustering method based on gaussian means and dbscan techniques. In *Intelligent Engineering Systems (INES), 2012 IEEE 16th International Conference on*, pages 573–578. IEEE, 2012.

[26] Jeff Speck. Walkable city: How downtown can save america, one step at a time nova york: North point press, 312 p. isbn 978-0865477728. *Documents dAnàlisi Geogràfica*, 61(2):437, 2015.

[27] Thanh N Tran, Klaudia Drab, and Michal Daszykowski. Revised dbscan algorithm to cluster data with dense adjacent clusters. *Chemometrics and Intelligent Laboratory Systems*, 120:92–96, 2013.

[28] D. Wishert. Mode analysis : a generalization of nearest neighbour which reduces chaining effects (with discussion). *Numerical Taxonomy*, pages 282–311, 1969. URL https://ci.nii.ac.jp/naid/10012395375/en/.

[29] Fatma Günseli Yaşar and Gözde Ulutagay. Challenges and possible solutions to density based clustering. In *Intelligent Systems (IS), 2016 IEEE 8th International Conference on*, pages 492–498. IEEE.