Urban Hotspots Mining Based on Improved FDBSCAN Algorithm

Jinhua Chen¹, Can Zhao², Kai Zhang¹ and Zhiheng Li¹, ²,*
¹Tsinghua Shenzhen International Graduate School, Shenzhen, 518000, China
²Department of Automation, Tsinghua University, Beijing, 10000, China

Abstract. Enormous human activities happen in modern cities nowadays. These activities generate trajectory information, which can play a big role in urban planning, transportation improvement, behavior analysis, and many location-based services. Currently, GPS devices are installed in many cities’ taxies. These positioning devices are the ideal source of trajectory data. Urban hotspots can be extracted by applying spatial clustering algorithm to the taxi trajectory data. Since the classical DBSCAN clustering algorithm is short of execution efficiency and its variant FDBSCAN clustering algorithm fails to make fine-grained clustering on the taxi trajectory data, this paper proposes an improved FDBSCAN clustering algorithm, OFDBSCAN clustering algorithm. The core of the O-FDBSCAN clustering algorithm is, it adds some coordinate offsets to the original trajectory data point based on some background geographic region knowledge. The background knowledge contains the collected functional domain information in the nearby area. By adding this kind of weight to the trajectory data point, the clustering result will be closer to the real target place and therefore give a more fine-grained clustering result. In this paper, we extract the urban hotspots and their spatiotemporal pattern by using the O-FDBSCAN clustering algorithm. The comparison result shows that, the proposed algorithm has better fine-grained clustering results than FDBSCAN.

1. Introduction

Trajectory is the path followed by an object moving in space and time [1]. With the rapid development in positioning technology, trajectory data of individual activities is technically and economically obtainable nowadays. Among various trajectory data, GPS-enable taxi trajectory is an important kind of spatial data which can effectively reflect the spatial patterns of urban residents’ travel behavior [2]. Compared with other trajectory data like mobile phone trajectory data, the features of taxi trajectory data benefits a lot in data analysis [3]. First, the GPS logger in a taxi has better positioning quality than the one inside a mobile phone. Second, Taxis GPS loggers have a denser data collection interval (a few seconds), which provides better data accuracy. Finally, the pick-up and drop-off points of a single trip can be easily extracted from the trajectory data. It’s even hard to define a complete trip from the raw mobile phone trajectory data since they are all discrete positioning points without any semantic information of a trip.

The latest literature has demonstrated a growing interest in applying trajectory data to detect urban hotspots [2]. A hotspot is a place of more than usual interest, activity, or popularity [4]. In the context of taxi trajectory analysis, the hotspot areas mean places of more than usual occurrence. In other
words, these places have a higher density of demand. Mining hotspot areas provides instructive insight to transport management, urban planning and location-based services (LBS) [3]. Urban hotspots can be identified from spatial data by using certain clustering algorithm. In the application of taxi trajectory data, we use a clustering method to group taxi trajectory points to obtain meaningful patterns.

There are different kinds of clustering algorithms, which can be classified into four categories: Partitioning, hierarchical, grid-based, and density-based clustering algorithms [3]. The K-means algorithm is a typical partitioning clustering algorithm which partitions objects into K clusters by maximizing the similarity of intra-cluster and minimizing the similarity of inter-cluster [5]. Among different kinds of clustering approaches, the density-based clustering algorithm can find arbitrary shapes clusters while other approaches is only capable of finding spherically-shaped clusters. In the case of the urban taxi trajectory analysis, the trajectory data generally have irregular shape. So it’s reasonable to use the density-based clustering algorithm to identify the spatial structure of the city. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering algorithm is a typical density-based clustering algorithm [6]. The idea of this algorithm is that for each object of a cluster, the neighbourhood of a given radius has to contain at least a minimum number of objects [7].

This paper focuses on mining the urban hotspots from taxi trajectory data by applying the density-based clustering approach. There is a major drawback of DBSCAN algorithm that when the border objects of two adjacent potential clusters are relatively close, the algorithm fails. In this situation, these two adjacent clusters will combine to be a bigger one, which is an imprecise and unsatisfactory clustering result. Thus, an improved FDBSCAN clustering algorithm is designed in this paper for extracting spatial structure of urban hotspots and their spatiotemporal pattern. Some comparison experiments are made to show the advantages of our proposed algorithm to the DBSCAN algorithm and the FDBSCAN algorithm.

The remainder of this paper is as follows: Section 2 reviews related works. Section 3 describes the detail of our proposed algorithm. A case study is reported in Section 4 to illustrate the application. At last we conclude the paper in Section 5.

2. Related Word
Spatial clustering is a major part of data mining techniques. In recent years, many different kinds of spatial clustering algorithms are developed to mine meaningful information from various spatial data. As stated in Section 1, the density-based clustering algorithm is the ideal choice in our work. In this section, we first introduce some researchers’ work of improved DBSCAN algorithm designs. Then we give a brief look at some work of spatial clustering algorithm applications in mining patterns from taxi trajectory data.

Fast DBSCAN (FDBSCAN), developed by Shuigeng Zhou et al. in 2000, is a fast DBSCAN algorithm which speeds up the original DBSCAN algorithm by selecting only a small number of representative points in a core point’s neighbourhood as seeds to expand the cluster [7]. In the same year, Shuigeng Zhou et al. presented another improved DBSCAN algorithm--PDBSCAN [8]. The PDBSCAN clustering algorithm divides data set into several partitions and set different Eps parameters for each partition. The global clustering result comes from the combination of these partitions’ clustering results. Jing Yang et al. presented an improved DBSCAN clustering algorithm based on data field [9]. In their work, the parameters of the original DBSCAN algorithm can be dynamically determined by introducing the concept of average potential difference. Liping Lai et al. developed the OPDBSCAN algorithm to get local Eps parameter instead of a global one by applying overlapping partitions [10].

Similar to improve the original DBSCAN algorithm, there are several work of improving the FDBSCAN algorithm. Guizhi Wang et al. proposed an improved FDBSCAN algorithm IF-FDBSCAN, which ensure not losing any object in the process of clustering [11]. Paper [12] developed an improved spatial clustering algorithm R-FDBSCAN, which adds a strict clusters range control to get a fine-grained clustering result.
A core drawback of the FDBSCAN algorithm is that it’s hard to generate a well fine-grained clustering result with dense adjacent clusters. Compared with the work of improving the original DBSCAN algorithm, the work of improving the FDBSCAN algorithm has not drawn much attention. To the knowledge of authors, the existing improvement approaches on FDBSCAN are focusing on modifying its internal algorithm process [8, 12, 13]. Our work is the first one to introduce external factors to modify the algorithm logic. So our work in this paper is a meaningful exploration in this area.

3. Methodology

3.1. Data Representation

Let $T = \{T_i\}$ be a taxi trajectory dataset and $N = |T|$ be the total number of trajectories.

$$T_i = (x'_0, y'_0, t'_0, s'_0), (x'_1, y'_1, t'_1, s'_1), ..., (x'_N, y'_N, t'_N, s'_N)$$

is an ordered sequence in the trajectory dataset, where $1 \leq i \leq N$, $x_j, y_j$ is a pair of position coordinate, $t_j$ is the timestamp of the position record where $t_0 < t_1 < ... < t_N$, $s \in \{0, 1\}$ is the status of the taxi where “0” represents the taxi is not carrying passengers while “1” represents the taxi is carrying some passengers. Let point set $P$ be the collection of records in $T$, where $P_i$ satisfies $P_{i-1} = (x_j, y_j, t_j, 0)$, $P_i = (x_j, y_j, t_j, 1)$, point set $P'_i$ be the collection of records in $T'$, where $P'_i$ satisfies $P'_{i-1} = (x_j, y_j, t_j, 1)$, $P'_i = (x_j, y_j, t_j, 0)$, which means $P$ is the collection of pick-up points (the status change from “0” to “1”) and $D$ is the collection of drop-off points (the status change from “1” to “0”).

The purpose of the taxi trajectory spatial clustering task is to group points in $P$ and $D$ into a number of clusters. These clusters reflect hot spots of urban activities. It’s under the assumption that the frequency of taxi visits is proportional to population density [3], a place with high population density means it has a lot of human activities.

3.2. FDBSCAN

The key idea of the FDBSCAN (DBSCAN as well) is that for each point of a targeting cluster, they need to contain at least a minimum number (given by an input parameter) of neighbourhood points in a certain radius (given by an input parameter). These two input parameters are named “MinPts” and “eps” respectively. The “MinPts” parameter must be set at least 3 in practice.

There are three key concepts in understanding the execution process of FDBSCAN. They are list as follows.

**Core Point:** A point which has more than MinPts points within eps.

**Border Point:** A point which is in the neighborhood of a core point (radius less than eps), but has fewer than MinPts points within eps.

**Noise Point:** A point which is neither a core point nor a border point.

The execution process of the FDBSCAN clustering algorithm can be described as follows:

1. Choosing arbitrary points which have not been visited and get their neighborhood points by eps parameter.
2. If one point contains MinPts neighborhood points within eps radius, a cluster is formed. Otherwise, the point is labeled as a noise (it can be a member of another cluster in later process).
3. If a point is a core point, the points of its eps neighbor is made part of the cluster.
4. The above process recursively works until all clusters are found.

3.3. Drawback of FDBSCAN

Some disadvantages of FDBSCAN are recognized these years [12, 13]. One of the core drawback of the algorithm is that there is not a well fine-grained clustering result on detecting border objects of adjacent clusters. In some cases where there are some objects besides each other, it’s hard to decide...
which cluster they should belong to. In this situation, a mixture of two adjacent clusters is generated, which is not the result it should be.

Now we give a sketch map to show how the drawback of FDBSCAN influence in the clustering result of taxi trajectory points and our approach to handle this situation. In Figure 1, one cluster (points marked as blue color) is obtained by the FDBSCAN algorithm. In a general view, this is a decent clustering result. However, the result can be subdivided to get a better fine-grained clustering result. Imagine we have two place of interests (POI) in the lower-left corner and the upper-right corner towards the points set respectively. In the context of taxi pick-up/drop-off points clustering, these POI can act as an important semantic information to smartly cluster taxi trajectory points. Each target point has its own trip purpose, so we attach each target point to certain POI. This is based on a supposition that the man at that point take taxi to go to the nearest POI. By setting POI as background geographic information, we add trip purpose to each point in the dataset to facilitate discovering trajectory data patterns and obtaining better clustering results. We add some coordinate offsets to the original trajectory data point to make them closer to the nearest POI. This kind of handling is based on the fact that in most cases, the point’s locations are not the staring or destination place of a trip, people need to have a short walk from the starting place or to the destination place, so the POI near the trajectory points is the real target place of a trip.

Compared with other traffic datasets such as the individual travel survey dataset and dataset involve much social media information, the taxi trajectory dataset contains little semantic information. By adding the trip purpose to data points, we can have a more precise classification in a big cluster and subdivide them into some smaller groups. The improved clustering result of Figure 1 is shown in Figure 2. Two sub-clusters are obtained (marked as blue and red color respectively).

The detail of the improved clustering algorithm is described in Section 3.4.

3.4. Improved FDBSCAN Based on Background Geographic Knowledge

The pseudocode routines of the revised FDBSCAN are presented in Figure 3. The original FDBSCAN algorithm differs from DBSCAN in two aspects: The first is the expansion procedure has a different
neighbour point’s selection strategy that it just selects some representative points of the core points to promote efficiency [7]. There are two selecting strategies described in [7]. More customized strategies can be designed by developers to meet their own needs. In most cases, a strategy that selecting four points (the leftmost, rightest, uppermost, and lowest border points) of a core point as its representative neighbour points is functional and efficient enough. The second difference is a procedure for lost points handling is added. However, this handing procedure is dispensable by careful considerations. The detail of reasons is analysed in [7], we just ignore this handling procedure as it did in its implementation.

Algorithm O-FDBSCAN (data, Eps, MinPts, POI_Eps, POI_Gap)
// All points in dataset are initialized as UNCLASSIFIED
For \( x_i \) in the dataset
   Nearest_POI, Second_Nearest_POI = Retrieve_Neighborhood_POIs (\( x_i \), POI_Eps)
   // Add offset to data points by POI information
   \( \text{If } \text{Dist} (\text{Nearest}_\text{POI}, \text{Second}_\text{Nearest}_\text{POI}) > \text{POI}_\text{Gap} \)
   \( x_l, x = x_l, x + (x_l, x - \text{Nearest}_\text{POI}, x) \)
   \( x_l, y = x_l, y + (x_l, y - \text{Nearest}_\text{POI}, y) \)
End If
End For
ClusterId = 1
For \( x_i \) in the dataset
   If \( x_i \) is UNCLASSIFIED
      ExpandCluster (\( x_i \), ClusterId)
      \( \text{If } \text{ExpandCluster} \text{ success} \)
      ClusterId = ClusterId + 1
   End If
End If
End For

Algorithm ExpandCluster (\( x_i \), ClusterId)
seeds = Retrieve Representative Points (\( x_i \), Eps)
If |seeds| < MinPts
   mark \( x_i \) as noise point
   Return False
Else
   assign all objects in seeds list to ClusterId
   delete point \( x_i \) from seeds list
   For all \( x_j \) in seeds list
      \( N_{\text{Eps}}(x_j) = \text{Retrieve}_\text{Representative}_\text{Points} (x_j, Eps) \)
      \( \text{If } |N_{\text{Eps}}(x_j)| \geq \text{MinPts} \) // \( x_j \) is a core point
         For all \( x_k \) in \( N_{\text{Eps}}(x_j) \)
            If \( x_k \) is UNCLASSIFIED
               add \( x_k \) to seeds list
            End If
            If \( x_k \) is NOISE
               assign \( x_k \) to ClusterId
            End If
         End For
      End If
   End For
End If
End If
End For

Figure 3. Pseudocode of the O-FDBSCAN algorithm

In the processing flow of the improved FDBSCAN algorithm, we firstly explore a list of POI (the detail of how to get these POIs and the property of them is described in the case study in Section 4.
Two additional input parameters are added, they are “POI_eps” and “POI_Gap”. The former parameter is used as the searching radius of a target point to search its neighborhood POI(s). After obtaining a list of neighborhood POI(s), we then have a look at the nearest POI and the second nearest POI. If they are closer to each other than the latter additional input parameter “POI_Gap”, it’s hard to tell which POI is the targeting place of a trip. In this case, we do not perform any change in the coordination of the point. In other cases, we add an offset to the point to update its coordination, making it closer to its real targeting place. We name the proposed algorithm as O-FDBSCAN, the character “O” represents “Offset”.

4. Case Study

4.1. Data Set

A case study with a taxi trajectory dataset in Shenzhen, China is presented to demonstrate and evaluate the proposed method. There are 121,490 trajectory records generated by 561 taxis in the dataset. The dataset description is shown in Table 1.

| Data Fields | Description | Data Type |
|-------------|-------------|-----------|
| ID          | Trajectory point identifier | Integer |
| TIME        | Record time | String |
| LON         | Longitude of point | Double |
| LAT         | Latitude of point | Double |
| SPEED       | Real-time speed | Double |
| STATUS      | 1/0 for carrying passenger or not | Integer |

The first step of data handling is extracting all the pick-up and drop-off points. As stated in Section 3.1, they can be identified by observing the changes in the STATUS field. The extraction result of pick-up points and drop-off points is shown in Figure 4. It can be observed that several points are not located in the domain of Shenzhen. As part of our data-pre-processing work, these points will be eliminated from the dataset before executing the algorithm.

![Figure 4. Distribution of taxi GPS position in Shenzhen](image)

As stated in the description of the proposed algorithm, POI collection acts as a key role in modifying original points’ coordination. Thus a collection of POI in Shenzhen is needed to promote our work. With the aid of “Baidu Open Map” project [14], we can write a crawler to collect different
kinds of POI in Shenzhen through “BaiduMapWebApiSpier” Application Interface. The description of POI data collected from the crawler is shown in Table 2.

| Data Fields   | Description                          | Data Type  |
|---------------|--------------------------------------|------------|
| Name          | Name of POI                          | String     |
| Province      | Province of POI                      | String     |
| City          | City of POI                          | String     |
| Area          | District of POI                      | String     |
| Address       | Address of POI                       | String     |
| Telephone     | Telephone number of POI              | String     |
| Uid           | POI identifier                       | String     |
| Street_Id     | Street of POI identifier             | String     |
| Detail        | 1/0 has detail information or not    | Integer    |
| Detail_Info   | Detail information of POI            | Key-Value Pairs |
| Location      | Longitude and latitude of POI        | Key-Value Pairs |

4.2. Urban Hotspots Mining

Our proposed algorithm is performed on the dataset described in Section 4.1. Different clusters are marked as different colors. The result of pick-up points clustering and drop-off points clustering is shown in Figure 5 and Figure 6 respectively. The biggest cluster has drawn much larger space than any other clusters. We believe that the cause of this phenomenon is places covered by cluster 1 do have much denser pick-up / drop-off points than other places. The time intervals of data acquisition are short as well, which leads to closer gap of adjacent points. There are many successive plains and no high mountain to separate these areas. These geographic features also impacts. In addition, different settings for the input parameters of the algorithm do make big differences in cluster result. A macro-view of the cluster distribution of the whole city is not the core of our target. In fact, famous hotspots like CBD of a city is well-known by its citizens. We focuses on some smaller hotspot areas in a city rather than those well-known city hotspots in order to find more meaningful spatial patterns in city. So the clustering result in Figure 5 and Figure 6 is not quite balanced, but we think this is acceptable in our research. There is no obvious difference in the overview of the whole map compared with the original algorithm since some cluster colors are overlapped by others. A closer look on small portion of the map is made in Figure 7 and Figure 8 to illustrate the difference.

![Figure 5. Cluster result of distribution of taxi pick-up GPS position in Shenzhen](image-url)
Figure 6. Cluster result of distribution of taxi drop-off GPS position in Shenzhen

We have stated the drawback of FDBSCAN in previous section. Due to its drawback, if several clusters are close to each other, they will group together to form a larger one (we can call it a “hotspot core”). The improved FDBSCAN algorithm can avoid the gathering of such hotspot cores, and maintaining a high clustering efficiency at the same time. To have a quantitative evaluation of the clustering result by different clustering algorithms, we introduce the notion of Cluster Separation [15].

The cluster separation of a clustering system’s output is defined by

\[
Sep = \frac{1}{C(C-1)} \sum_{i=1}^{C} \sum_{j \neq i}^{C} \exp \left( -\frac{d^2(x_i, x_j)}{2\sigma^2} \right)
\]

where \(\sigma\) is a Gaussian constant, \(C\) is the number of clusters, \(X_i\) is the centroid of the cluster \(c_i\), \(d()\) is the distance metric used by the clustering system, and \(d(x_i, x_j)\) is the distance between the centroid of \(c_i\) and the centroid of \(c_j\). The Gaussian function and the L1-normalization normalizes its value to between 0 and 1. A smaller cluster separation score indicates a larger overall dissimilarity among output clusters. Experiments with different number of clusters generated are carried out by applying FDBSCAN and O-FDBSCAN. We calculate the separation score of their outputs. The experiment result is shown in Table 3. All values are shown with the mean and the standard deviation over ten runs.

|                  | Cluster number = 10 | Cluster number = 20 |
|------------------|---------------------|---------------------|
| FDBSCAN          | 0.4207 0.0486       | 0.4011 0.0197       |
| O-FDBSCAN        | 0.3799 0.0278       | 0.3650 0.0259       |

The experiment result shows that the cluster separation scores of O-FDBSCAN are better than those of FDBSCAN. It verifies our argument that our proposed algorithm has better fine-grained clustering results than FDBSCAN algorithm. The cluster separation scores decrease with the increase of cluster size, which is reasonable because when there are more clusters generated under a same dataset, they will get closer with each other and be less dissimilarity.

Besides quantitative evaluation of the algorithm, we also provide a visual demonstration in Figure 7 to better illustrate the feature of O-FDBSCAN algorithm. This is a selected section from the result of clustering on the dataset described in Section 4.1 using O-FDBSCAN. It can be noticed in Figure 8 that four groups on the right side, marked as deep-green, purple, yellow and green color respectively, are separated properly though there are close to each other while FDBSCAN fail to separate them.
5. Conclusions
In this paper, we looked at the drawback of FDBSCAN algorithm and proposed a novel improved FDBSCAN clustering algorithm. A case study is made to use our proposed algorithm to mine hotspots from taxi trajectory data in a big city of China. The core of our algorithm is adding some background geographic knowledge (POI) to modify the coordination of original points in dataset, which has more precise understanding of trip purpose. Our work is the first one to introduce external factors to modify the FDBSCAN algorithm logic. The proposed algorithm has a better fine-grained clustering result than the original algorithm does. The algorithm comparison experiment is carried out by calculating the cluster separation scores of two algorithms. The experiment result shows that the proposed algorithm has better performance in cluster separation than the original one.

Acknowledgments
This work was supported in part by the National Key R&D Program of China (No.2018YFB0105100) and by Shenzhen Municipal Science and Technology Innovation Committee (JCYJ20170412172030008).

References
[1] Alvares, Luis Otavio, et al. "Dynamic Modeling of Trajectory Patterns using Data Mining and Reverse Engineering." Twenty-sixth International Conference on Conceptual Modeling-er-tutorials 2007.
[2] Zelong Xia, Hao Li, Yuehong Chen, Weisheng Liao. "Identify and Delimitate Urban Hotspot Areas Using a Network-Based Spatiotemporal Field Clustering Method. " ISPRS International Journal of Geo-Information 8(2019):344.
[3] Feng Mao, Minhe JI, and Ting LIU. "Mining Spatiotemporal Patterns of Urban Dwellers from
Taxi Trajectory Data." *Frontiers of Earth Science* 10.2(2016):205-221.

[4] Chang, Han Wen. Y. C. Tai, and J. Y. J. Hsu. "Context-aware taxi demand hotspots prediction." *International Journal of Business Intelligence and Data Mining* 5.1(2010):3.

[5] Macqueen, J. "Some Methods for Classification and Analysis of Multivariate Observations." *Proc of Berkeley Symposium on Mathematical Statistics & Probability* 1965.

[6] Ester, Martin, et al. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise." *International Conference on Knowledge Discovery & Data Mining* 1996.

[7] Fud, Ayzhou. "FDBSCAN: A Fast DBSCAN Algorithm." *Journal of Software* (2000).

[8] Geng, Zhou Shui. Z. A. Ying, and C. Jing. "A Data-Partitioning-Based DBSCAN Algorithm." *Journal of Computer Research and Development* (2000).

[9] Yang, Jing, et al. "An Improved DBSCAN Clustering Algorithm Based on Data Field." *Journal of Frontiers of Computer Science & Technology* (2012).

[10] Li-Ping, Lai, et al. "Improved DBSCAN Algorithm Based on MapReduce." *Computer Science* (2015).

[11] Wang, Gui Zhi, and G. L. Wang. "Improved Fast DBSCAN Algorithm: Improved Fast DBSCAN Algorithm." *Journal of Computer Applications* 29.9(2009):2505-2508.

[12] Panpan Liu. Research on Mining Taxi Pick-up Hotspots Based on Spatial Cluster and Weka [D]. Jilin University, 2014.

[13] Tran, Thanh N., K. Drab, and M. Daszykowski. "Revised DBSCAN algorithm to cluster data with dense adjacent clusters." *Chemometrics & Intelligent Laboratory Systems* 120.2(2013):92-96.

[14] Baidu Corporation. Baidu Open Map. [EB/OL]. https://lbsyun.baidu.com.

[15] He, Ji, et al. "On Quantitative Evaluation of Clustering Systems". *Clustering and Information Retrieval*. Springer US, 2004.