Pattern Recognition Letters 53 (2015) 77–84

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A grid-growing clustering algorithm for geo-spatial data ☆

Qinpei Zhao [a], Yang Shi [a], Qin Liu [a,*], Pasi Fränti [b]

[a] School of Software Engineering, Tongji University, Shanghai, 201804, China
[b] School of Computing, University of Eastern Finland, Joensuu, 80101, Finland

ARTICLE INFO

Article history:
Received 15 March 2014
Available online 22 October 2014

Keywords:
Grid-based clustering
Grid-growing
Geo-spatial data
GPS devices
Regions of interest

ABSTRACT

Geo-spatial data with geographical information explodes as the development of GPS-devices. The data contains certain patterns of users. To dig out the patterns behind the data efficiently, a grid-growing clustering algorithm is introduced. The proposed algorithm takes use of a grid structure, and a novel clustering operation is presented, which considers a grid growing method on the grid structure. The grid structure brings the benefit of efficiency. For large geo-spatial data, the algorithm has competitive strength on the running time. The total time complexity of the algorithm is $O(N \log N)$, where the time complexity mainly comes from the seed selection step. The grid-growing clustering algorithm is useful when the number of clusters is unknown since the algorithm requires no parameter on the number of clusters. The clusters detected could have arbitrary shapes. Furthermore, sparse areas are treated as outliers/noises in the algorithm. An empirical study on several data sets indicates that the proposed algorithm works much more efficiently than other popular clustering algorithms.

1. Introduction

Due to the emergence of GPS-devices, variants of location-based applications are developed. Geo-tagged multimedia, such as trajectory, photo, video and text, are collected by users through the applications. The wealth of geo-spatial data provides an opportunity for performing further research topics.

Because of the convenience of collecting geo-spatial data, the size of the data usually explodes. Therefore, there comes up problems related to storage, visualization and detection of meaningful patterns. Enormous amounts of GPS trajectories, which record users' spatial and temporal information bring heavy burdens for both network transmission and data storage. A compression algorithm in [1] optimizes both the trajectory simplification and the coding procedure using the quantized data. The way of visualization [2] on the geo-tagged data needs to be well designed in order to avoid problems such as clutter problem.

GPS trajectories and other geo-spatial data often contain large information and unknown pattern. For example, a bunch of geo-spatial data can be collected from photos and trajectories uploaded by one user. From the data, it would be interesting for the user to know, manage and share his/her activity area. For other users, the activity area can be a suggested place to visit. In [3], research on extracting associative points-of-interest patterns from geo-tagged photos in Queensland, Australia is introduced. A system to provide both location and activity recommendations is introduced in [4], which is based on the location information of GPS data and some available user comments. Based on the concept of semantic trajectories, trajectories are observed as a set of stops and moves [4,5]. The start and end points of a trajectory could be interesting places with very high probability [6].

There is an increasing need for methods to extract knowledge from the data [7]. Clustering algorithms are applied on users' GPS data to discover spatio or temporal patterns. Typical clustering algorithms include model-based clustering (e.g., EM algorithm [8]), partition-based clustering (e.g., *k*-means and its variants [9,10]), graph-based clustering (e.g., spectral clustering [11]), density-based clustering (e.g., DBSCAN) and grid-based clustering. Different types of clustering algorithms have their advantages and disadvantages. Two aspects are usually considered when the algorithms are applied in geo-spatial data. One is the efficiency because the geo-spatial data is usually large. The other one is the adjustment on the algorithm to adapt in real applications.

A variant of *k*-means algorithm is proposed in [12] to cluster places where GPS signal is lost from the satellites into significant locations. The clusters are initially centered at K chosen points with a given radius, and iteratively move to a denser area. However, *k*-means related methods need to determine the parameter K beforehand. A hybrid clustering algorithm [13] that combines hierarchical method and grid-based method is presented to discover frequent spatial patterns

☆ This paper has been recommended for acceptance by D. Dembele.

* Corresponding author. Tel.: +86 (0)21 69589976.
E-mail addresses: zhao@cs.joensuu.fi (Q. Zhao), qin.liu@tongji.edu.cn (Q. Liu).

http://dx.doi.org/10.1016/j.patrec.2014.09.017
0167-8655/

among trips. Algorithm *SMoT* [14] and its alternative variant [5] are proposed to compute stops and moves from trajectory sample points. A clustering algorithm based on changing parts of the concepts in *DBSCAN* [15] is proposed in [5] to find low speed regions from trajectories. Because the DBSCAN has features of detection on clusters of arbitrary shape and noise detection, the algorithm is also commonly used for processing geo-spatial data. A *DBCLASD* [16] is designed with the advantages of discovering clusters of arbitrary shape and requiring no parameters. A new clustering algorithm based on DBSCAN [17] is presented specially for the problem of analysis of places and events using large collections of geo-tagged photos.

It is demonstrated in [18] that the grid-based technique obtains better results than the density-based one. The grid-based clustering algorithm [19] partitions the data space into a certain number of cells and performs clustering operations on the cells. Therefore, the algorithm is efficient when the number of cells (n) is much less than the size of the original data (N). The grid-based algorithms require at least one scan of all individual objects (points), which makes the time complexity of the algorithms at least O(N). A statistical information grid-based method (*STING*) was introduced in [20], which reduced the time complexity of O(N) to O(n) for each query. Since the method constructed a hierarchical structure by going through the whole data, the overall complexity is still linearly proportional to the size of data with a small constant factor. A grid-based hierarchical clustering algorithm was proposed in [13] for large-scale and event-based telematics data sets. A merge operation among neighborhood clusters is employed.

In this paper, we focus mainly on the efficiency of the algorithms. A grid-growing clustering algorithm is proposed for geo-spatial data specifically. To verify the validity of the algorithm, artificial data sets are tested firstly. Then, the efficiency is demonstrated by geo-spatial data which are sampled from trajectories. The experimental results demonstrate that the proposed algorithm is more effective and much faster than traditional clustering algorithms such as a k-means variant (*litekmeans*), Greedy EM, DBSCAN, a spectral clustering (*LSC*) and a pairwise random swap clustering (*PRS*).

The rest of the paper is organized as follows: Section 2 introduces the grid-growing clustering algorithm. An analysis on the time complexity of the algorithm is also included in the section. The experimental results are displayed in Section 3 and the conclusion is followed in Section 4.

2. The grid-growing clustering algorithm

The proposed grid-growing clustering algorithm (see Algorithm 1) is mainly designed for geo-spatial data. Let $D(x, y)$ be the location-based data with N points and P be the partitions as the result from the clustering algorithm.

Given the data D, the first step is to generate a grid structure $I(n, n)$, where n is the number of rows and columns in the grid structure. The number of n decides the size of the grids. For each data point, it is assigned to appropriate grids according to its locations.

The second step is to perform a region growing on the grid structure, which generates a certain number of groups. In this step, m seeds are firstly selected. With the selected seeds, regions are grown to adjacent points. K number of regions or clusters are formed after then.

Finally, the clustering partitions P are obtained from the K number of regions. We explain the details for each step in the following sections (Algorithm 2).

2.1. Grid construction

For each point in data set $D(x, y)$, the row ($t \in 1, 2, \ldots, n$) and column ($s \in 1, 2, \ldots, n$) number of the grids that the point belongs are calculated as follows:

$$t = \left\lceil \frac{x_{\max} - x}{x_{\max} - x_{\min}} \cdot n_x \right\rceil$$
$$s = \left\lceil \frac{y_{\max} - y}{y_{\max} - y_{\min}} \cdot n_y \right\rceil \quad (1)$$

where, $x_{\max}$ and $y_{\max}$ are the maximum values among x and y coordinates, whereas, $x_{\min}$ and $y_{\min}$ are the minimum values. Each grid $I(t, s)$ is represented by t and s.

We define the capacity of grid $I(t, s)$ as n_{ts}, which represents the number of points that are located in the grid. For each point, n_{ts} is increased by one ($n_{ts}++$) after its assignment to the grid. After then, the grid structure is constructed.

An example of a grid structure is shown in Fig. 1. The original data distribution is also shown. The number of rows and columns in the grid structure are 8 and 7 respectively. The grid structure $I(n, n)$ is similar with a gray image structure, where each grid can be considered as a pixel and the intensity of the pixel is the capacity of the grid. The choice of n is important because the time complexity of the grid-based algorithm is linearly increased with n. However, a large n does not necessarily bring a good performance of the algorithm.

Input: $D(x, y)$, n, m
Output: P
1 Step 1: Construction on grid structure I
$I(n, n) \leftarrow GridConstruct(D, n)$;
2 Step 2: Grid Growing on I,
$R \leftarrow GridGrowing(I > 0, seed)$ (see Algorithm 2);
3 Step 3: Get partitions $P \leftarrow GetPartition(R)$;
4 return P;

Algorithm 1: *Grid-growing* clustering algorithm

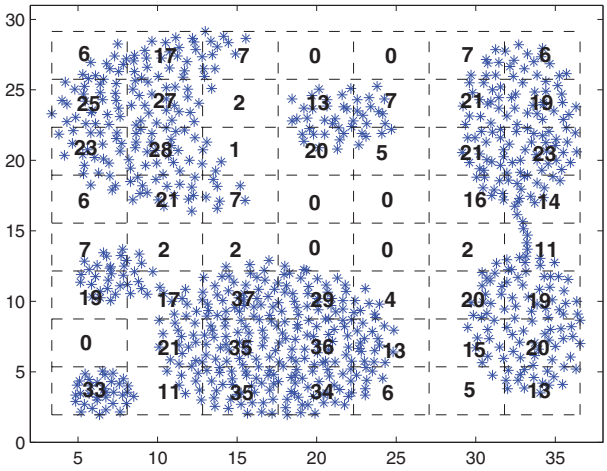

Fig. 1. An example of a grid structure on data *aggregation*. The numbers in the grids are the capacity of the grids.

2.2. Grid growing

With the grid structure $I(n, n)$ constructed, a grid growing step is performed. The step begins with selecting m initial seed points on I and the initial regions begin with the exact location of these seeds. For a spatial data, high density locations indicate points of interest. Therefore, the seeds are selected based on the top m capacity values in I. There are also alternatives for selecting of seeds. A more straightforward and efficient way is to randomly select the seeds.

The regions start to grow from each seed by searching adjacent points. The search can be performed in 4-neighbors or 8-neighbors points, where the latter one brings more accurate result. If the