Evaluating Clustering Techniques

Dmitri Timašjov
Institute of Computer Science
University of Tartu
b04886@ut.ee

Supervisor: Amnir Hadachi

ABSTRACT

The choice of particular clustering method very often depends on the origin of the dataset. Different clustering methods and try to evaluate what technique produces clusters of higher quality. In this work we concentrate on time-series data and as input took Open cell ID's dataset[1]. We try DBSCAN, OPTICS, k-means and hierarchical clustering algorithm and as an evaluation method calculate coverage, density and modulatiry of created clusters.

Keywords

clustering, DBSCAN, OPTICS, k-means, hierarchical clustering, modularity, coverage, density, cell trajectory

1. INTRODUCTION

With more space-related data becoming available it becomes more harder to distinguish useful information. Applying generalization techniques, such as clustering, can greatly help with the understanding and prediction of the domain environment. Clustering is a technique for grouping similar objects according to chosen characteristics. It has been addressed by researches by decades and is one of most essential and crucial techniques in text mining [1], cartography [2], social studies [3] and bioinformatics [4]. Therefore, the process of evaluating of formed clusters in an important issue.

Mainly there are four broad clustering categories: hierarchical clustering, centroid-based clustering, distribution-based clustering and density-based clustering. All clustering techniques have their benefits, drawbacks and limitations. For example, hierarchical clustering has on average $O(n^3)$ complexity, which makes it very slow for large datasets. For k-means clustering, most famous centroid-based clustering representative, the number of clusters should be known in advance, which is considered to be the biggest disadvantage of the algorithm. Also, it randomly chooses centroids at the beginning of the procedure, which often lead to the incorrectly cut orders in between the clusters. Density based clustering algorithms, such as DBSCAN and OPTICS, cannot always detect clusters of a good quality as they expect density drop to detect cluster borders. As a result, they cannot be applied for datasets when density is continuously decreasing. Distribution based clustering methods have a very strong assumption of the dataset as it is not always possible to determine a good mathematical model for clustering. However, practically for every clustering method there exist possible improvements, which may improve the process of clustering. For example, [5] provides a possible improvement for k-means clustering algorithm by applying categorical values. All this leads to the fact that multiple clustering techniques are applied in order to produce better results.

In this work we are going to concentrate on four well-known clustering algorithms, namely DBSCAN, OPTICS, k-means and agglomerative clustering. In this work we will concentrate mainly on trajectories, i.e clustering moving objects following through space as a function of time. So, the main question of this work is as follows: what is the most suitable and efficient clustering method for trajectories? Algorithms will be tested on real dataset representing user trajectories, i.e., time-series GPS data of (x, y) coordinates. Example GPS point trajectory is illustrated on Figure 1.

Cluster analysis is a very biassed and overweighted technique and often it is very hard to validate the cluster say whether formed clusters are optimal or not. There exist multiple validity criterias and indices, for example Silhouette index, Davies-Bouldin index, Maulik-Bandyopadhyay index, Dunn index, score function [7]. They have been studied a lot and examples of evaluating them can be found in [6]. However, in this work we will concentrate on calculating three most essential cluster properties, namely density, coverage and modularity.

The article is organized as follows. Section 2 describes above mentioned clustering algorithms in a greater detail. Cluster evaluation method is overviewed in Section 3. Section 4 concentrates on analyzing the evaluation results. Section 5 concludes our work and provides directions for future work.

2. CLUSTERING ALGORITHMS

In this section we briefly present clustering algorithms that we are going to evaluate, namely DBSCAN, OPTICS, k-means and agglomerative clustering.

[1]Can be downloaded, for example, from `http://openstorage.gunadarma.ac.id/cellsIdData/`

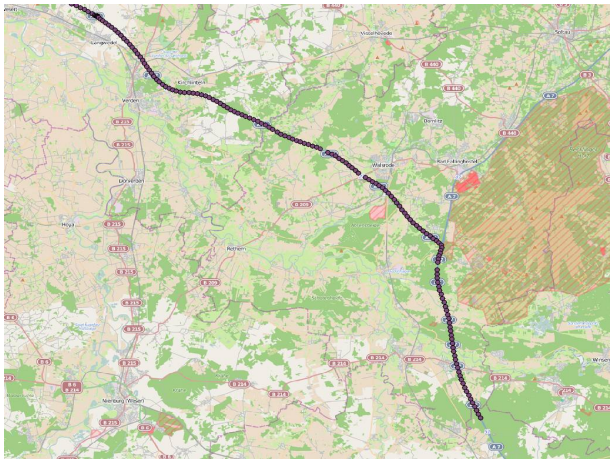

Figure 1: GPS point trajectory

Density-based spatial clustering of applications with noise (DBSCAN) algorithm uses notion of density reachability ϵ to discover clusters. Algorithm identifies all point p neighbours who are within distance ϵ. If number of such neighbours is greater than minimum predefined number, points are considered as a part of a cluster, otherwise p is considered as a noize. Algorithm terminates when all points are visited.

Ordering points to identify the clustering structure (OPTICS) algorithm idea is very similar to DBSCAN algorithm idea and it also belongs to family of density based algorithms. It allows more correctly to detect clusters in data of varying density by ordering points by spatial distance. Algorithm uses both core- and reachability distance for discovering point p neighbours and priority queue for holding set of discovered, but not yet visited points. Points in priority queue are linearly ordered by spatial distance, which means that spatially closest points with higher probability will become a part of a cluster. In such a way algorithm examines each point and terminates when all points are visited. Average complexity of both DBSCAN and OPTICS algorithms is $O(n^2)$. Also, both algorithms usually use Euclidean distance as metric for calculating distance between points.

K-means is probably the most popular clustering algorithm. It divides the dataset into known number of clusters k. Idea of the algorithm is to randomly pick k means and assign each point to the mean according to the Euclidean distance metric, and create k clusters. Then new centroids are recalculated as barycenters, assigned as new means and then step is repeated. Algorithm terminates when during the iterations cluster centroids have not been changed. Algorithm complexity is $O(nKId)$, where n is number of points, K number of clusters, I number of iterations needed until convergence, and d number of d-dimensional vectors. Clustering result greatly depends on value of k, thus challenge exist for selecting optimal k value. This can be achieved by running clustering algorithm multiple times with different k values and comparing results.

Agglomerative hierarchical clustering builds a hierarchy of clusters by merging clusters together according to the metric and linkage criterion between them. At the beginning each point in agglomerative method is represented as a separate cluster. Next, algorithm identifies a pair of most similar clusters A and B and merges them. And then the step is repeated. Algorithm terminates either when all clusters are merged or when distance between clusters is greater than threshold. So, algorithm merges clusters as one moves up the hierarchy. Euclidean distance is often used as a metric, however, other linkage criterias may also be used, e.g., Manhattan distance or Mahalanobis distance. Complete linkage clustering or single-linkage clustering is usually used as a linkage criteria between clusters A and B. Algorithm complexity is $O(n^3)$, thus it works too slow with large datasets.

There are exist two fundamentally different approaches of clustering trajectories: clustering by GPS coordinates and clustering by cell ID's. In the former case GPS coordinates are extracted from the trajectory and clusters are created based on the distances between GPS points, while in the latter case cell ID's are taken as an input and similar movements between different cells are clustered. In this work we will concentrate only on clustering trajectories by cell ID.

3. CLUSTER EVALUATION METHOD

In this work we will evaluate clustering algorithms by calculating *coverage*, *density* and *modularity* of created clusters. We will concentrate on these indices as they give basic understanding of the quality and type of clusters needed for prelusory conclusions. All these indices are widely used in the analysis of large graphs, for example, in analysis of social networks or biological data.

First, we will introduce basic definitions needed for proper understanding of evaluation parameters. *Intra-cluster* edges are the edges between vertices of the same cluster, while *inter-cluster* edges are the edges between vertices of different clusters. In all subsequent formulas C is defined as a cluster, n as a total number of vertices and m as a total number of edges in the graph.

Coverage is the ratio of intra-cluster edges to the total number of edges in graph. It shows how many edges are within the cluster and number of vertices interpreted as noize.

$$\text{coverage(C)} := \frac{\#\text{intra-cluster edges}}{m}$$

Modularity is a measure for determining the strength of the division of the graph into modules. In other words, high modularity means there are more edges within the module that you expect by chance. It lies in range [-1, 1]. There exist multiple methods for calculating modularity and we will use following formula:

$$\text{modularity(C)} := \sum_{i \in C} \left(\frac{e_{ii}}{2m} - \left(\frac{\sum\limits_{j \in C} e_{ij}}{2m}\right)^2\right)$$

Expression on the left of the subtraction sign is the fraction of the edges within cluster i, expression on the right is the fraction of outgoing edges from a cluster i.

Third parameter is density. Authors of [1] define density

as a paradigm of intra-cluster density versus inter-cluster sparsity. Higher density means that edge-connectivity inside clusters is very strong, while between different clusters is very weak. Formula for calculating is as follows:

$$\text{density(C)} := \frac{1}{2}\left(\frac{1}{|C|}\sum_{c\in C}\frac{\#\text{intra-cluster edges of c}}{\binom{c}{2}}\right)$$

$$+\frac{1}{2}\left(1-\frac{\#\text{inter-cluster edges}}{\binom{n}{2}-\sum_{c\in C}\binom{|c|}{2}}\right)$$

Expression on the left of the addition sign is an intra-cluster density, expression on the right is an inter cluster sparsity.

4. IMPLEMENTATION AND RESULTS

4.1 Dataset

Initial dataset contain ≈ 70 million GPS points and ≈ 2 million of cells. Each GPS point has variety of different properties, however, we are going to concentrate on following: latitude and longitude coordinates, cell ID where point was recorded, time when it was measured, point signal strength and user ID. Each cell point has a signal, latitude and longitude, time when tower discovered a point, mobile country code, mobile network code and location area identity code.

Due to high complexity of clustering algorithms and time constraints, we concentrated on a subset of 1 million points. Next, we implemented clustering algorithms, clustered the dataset and analyzed it using above mentioned techniques. Groovy language was used for implementation.

4.2 Workflow

Foremost, initial dataset was preprocessed. The main idea of preprocessing was to avoid ping-pong handover - effect when users appears several times in multiple neighbouring cells due to the various conditions of the signal propagation. For GPS point trajectories this effect does not has so significant impact and multiple cell ID's were just connected to one GPS point. For clustering trajectories based on cell ID's situation completely differs: in order to construct proper trajectory it is needed to distinguish in what cell user was at particular time. In theory, cell geometry has a form of rhombus, however, due to uneven terrain and buildings geometry may change and, thus we do not know the exact geometry, and consequently coverage area, of the cell. Complicated algorithms based on coordinates and signal strength [7] exist for determining the most probable cell ID, however, in this work we will use much simpler algorithm. We calculate how many times cell was has recorded the particular GPS signal and choose cell with higher number of records. If there is no such cell, we choose the cell who first managed to discover and record the signal. In this work we concentrated only on clustering trajectories based on cell ID.

Our goal was to find similar trajectories in different time periods. This would lay the foundation for possible data mining, e.g., determining the most popular routes or finding human mobility patterns. Before doing any clustering and analysis, we divided dataset into periods of 4 hours and handled each period independently. Such division into periods would provide us with clusters more suitable for analysis of time-series data. So, workflow is as follows. It was repeated for all four considered clustering algorithms.

- Cluster trajectories in each time period using clustering algorithm. In this step we distinguish similar groups of points in trajectory. Example clusters of trajectories can be viewed on Figure 2.
- Cluster created clusters using clustering algorithm. In this step we recognize similar groups of clusters considering all trajectories. Example cluster of clusters of trajectories can be viewed on Figure 3.
- Analyze created clusters by calculating coverage, density and modularity.
- Project results on a map.

Figure 2: Clustered cell points with DBSCAN algorithm. Points with same color are in one cluster.

4.3 Results

Quality of created clusters and algorithm running times greatly depends on the algorithm input parameters. For our experiments, we decided to choose following parameters:
- DBSCAN (epsilon: 3, minimum points in cluster: 2)
- OPTICS (epsilon: 3, minimum points in cluster: 2)
- K-means (number of clusters: 4, converge distance: 0.01)
- Hierarchical (threshold: 5)

We conducted all tests with first 1 million points from the initial dataset. Source code can be accessed via URL: `https://bitbucket.org/timasjov/clustering-algorithm`. Results can be found in Tables 1-8.

It is evident that results greatly depend on input parameters we used. Clustering algorithms from diverse categories produce different number of clusters of different quality and size. For example, combination of OPTICS and k-means algorithm produce most compact clusters, where each cluster contains the smallest number of sub-clusters in it. K-means

algorithms cannot discover noize and puts each point in some cluster and, thus, produces the greatest number of clusters. It is especially noticeable in comparison with applying hierarchical algorithm twice - combination of k-means algorithm produced $\approx$ 130 times more clusters. We used convergence to a local minimum in the algorithm implementations, which may in some cases produce wrong results.

Results produces by both density-based clustering algorithms differ markedly. Applying DBSCAN algorithm twice allows to detect 10 times more clusters than applying two times OPTICS algorithm. This may happen because latter one uses more sophisticated approach for detecting meaningful clusters in data of varying density: each cluster contain $\approx$ 3 times less points. This means that in comparison with DBSCAN, OPTICS interprets $\approx$ 50% of points as noize.

Despite the fact that hierarchical and OPTICS algorithms use completely different approaches for clustering, their results are very similar. Also, the order of applying algorithms does not play a significant role: hierarchical & hierarchical algorithms produce $\approx$ 99 clusters while OPTICS & hierarchical algorithms produce $\approx$ 93 clusters, OPTICS & OPTICS algorithms produce $\approx$ 64 clusters while hierarchical & OPTICS algorithms produce $\approx$ 69 clusters.

Table 1: Average number of clusters in cluster and average number of points in cluster.

Algorithm	No. clusters	No. points in cluster
DBSCAN & DBSCAN	548.383	62.072
DBSCAN & OPTICS	126.813	45.374
DBSCAN & k-means	5046.846	54.147
DBSCAN & hierarchical	189.192	42.998

Table 2: Average number of clusters in cluster and average number of points in cluster.

Algorithm	No. clusters	No. points in cluster
OPTICS & DBSCAN	232.937	22.060
OPTICS & OPTICS	64.624	21.191
OPTICS & k-means	4142.162	17.382
OPTICS & hierarchical	93.927	18.794

Table 3: Average number of clusters in cluster and average number of points in cluster.

Algorithm	No. clusters	No. points in cluster
k-means & DBSCAN	969.106	54.695
k-means & OPTICS	201.334	44.346
k-means & k-means	13892.066	23.001
k-means & hierarchical	109.009	23.250

Table 4: Average number of clusters in cluster and average number of points in cluster.

Algorithm	No. clusters	No. points in cluster
hierarchical & DBSCAN	323.937	36.079
hierarchical & OPTICS	69.933	24.790
hierarchical & k-means	3919.533	19.166
hierarchical & hierarchical	99.491	22.826

Another metrics that we were evaluating were coverage, density and modularity of created clusters. Results produced by all combination of algorithms are very similar and contain no anomalies. First, in all cases average density is higher than 0.5, which means that edge-connectivity inside clusters is stronger than average. Highest density is achieved with combination of k-means and hierarchical algorithms, while lowest with combination of OPTICS and k-means algorithm. Second, in all cases average modularity of clusters is negative and is very close to zero. Negative values indicate the possible absence of community structure. Third, average coverage of all algorithms is also very close to zero, which means that the number of intra-cluster edges inside cluster in very small. The only exceptions are k-means & k-means and hierarchical & k-means algorithms, which has coverage greater than 0.3.

Table 5: Average coverage, density and modularity of created clusters.

Algorithm	Coverage	Density	Modularity
DBSCAN & DBSCAN	0.036	0.590	-6.479e-009
DBSCAN & OPTICS	0.008	0.598	-1.168e-009
DBSCAN & k-means	0.343	0.601	-3.816e-007
DBSCAN & hierarchical	0.012	0.601	-1.900e-009

Table 6: Average coverage, density and modularity of created clusters.

Algorithm	Coverage	Density	Modularity
OPTICS & DBSCAN	0.018	0.526	-1.881e-008
OPTICS & OPTICS	0.005	0.529	-3.758e-009
OPTICS & k-means	0.335	0.522	-7.003e-007
OPTICS & hierarchical	0.008	0.528	-2.731e-009

Table 7: Average coverage, density and modularity of created clusters.

Algorithm	Coverage	Density	Modularity
k-means & DBSCAN	0.025	0.618	-5.535e-009
k-means & OPTICS	0.005	0.611	-9.366e-010
k-means & k-means	0.387	0.554	-3.961e-007
k-means & hierarchical	0.008	0.648	-7.292e-010

Table 8: Average coverage, density and modularity of created clusters.

Algorithm	Coverage	Density	Modularity
hierarchical & DBSCAN	0.025	0.617	-5.576e-009
hierarchical & OPTICS	0.005	0.609	-9.135e-010
hierarchical & k-means	0.318	0.576	-5.186e-007
hierarchical & hierarchical	0.008	0.610	-8.497e-010

From out point of view combination of hierarchical and DBSCAN clustering algorithm produced the clusters of highest quality. Reasons are twofold: 1) created clusters are very dense, which means that they may be located close to each other and 2) the number of sub-clusters in each cluster is higher than overall average.

5. CONCLUSION

Cluster analysis is an iterative process, which involves lots of trials and failures. The selection of the algorithm directly depends on the dataset and the origin of data. Performing different optimizations, selecting and tuning algorithm parameters is also not an automatic task and greatly depends on the intended use of the results. In this article we implemented four clustering algorithms, namely DBSCAN, OPTICS, k-means and hierarchical, clustered proposed dataset and evaluated created clusters by calculating density, coverage and modularity of clusters. From our point of view, combination of hierarchical and DBSCAN clustering algorithms produced clusters of highest quality.

Figure 3: Clustered clusters of cell points with combination of DBSCAN and hierarchical algorithms. Ellipses represent main clusters. Points with same color are in one sub-cluster.

6. REFERENCES

[1] SanJuan, E., Ibekwe-SanJuan, F.: Text mining without document context. Inf. Process. Manage. 42(6) (2006) 1532–1552

[2] Huarong, L., & Yi, Z. (2011, December). Vectorization of linear features on color scanned map. In Computer Science and Network Technology (ICCSNT), 2011 International Conference on (Vol. 2, pp. 1148-1152). IEEE.

[3] Nina Mishra, Robert Schreiber, Isabelle Stanton, and Robert E. Tarjan, Finding Strongly-Knit Clusters in Social Networks, in Internet Mathematics, 2009

[4] Geng, Huimin, Dhundy Bastola, and Hesham Ali. "A new approach to clustering biological data using message passing." Computational Systems Bioinformatics Conference, 2004. CSB 2004.

[5] Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2(3) (1998) 283–304

[6] Traag, V. A., Browet, A., Calabrese, F., & Morlot, F. (2011, October). Social event detection in massive mobile phone data using probabilistic location inference. In Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom) (pp. 625-628). IEEE.

[7] Saitta, S., Raphael, B., & Smith, I. F. (2007). A bounded index for cluster validity. In Machine Learning and Data Mining in Pattern Recognition (pp. 174-187). Springer Berlin Heidelberg.