Mining Frequent Trajectory Patterns From GPS Tracks

Careelika Liisi Kuik
University of Tartu
Institute of Computer Science
Email: clk@ut.ee
Supervisor: Amnir Hadachi

***Abstract*—The aim of this paper is mining frequent trajectory patterns from GPS tracks collected from birds. Mining is done using DBSCAN (Density-Based Spatial Clustering of Applications with Noise)[1] clustering algorithm that is based on density and finds most common stay points along the trajectory. As a result, it is possible to see frequent patterns in birds' trajectories and estimate birds' behavior. The result is shown on QGIS [2] maps.**

I. Introduction

The aim of this paper is mining frequent trajectory patterns from GPS tracks collected from birds. The GPS data collected from the devices attached to the birds provide a good opportunity to collect and analyze birds' trajectories. The data used in this work is a dataset from Research Institute for Nature and Forest. The initial dataset contains of the tracks of 75 Lesser Black-backed Gulls and 26 Herring Gulls breeding at the Belgian and Dutch coast with all together 2.5 million data records.[3] As the data has been collected during a time period of more than two years, the records of all the birds are not collected on the same period. However, analyzing this data by clustering data points can give much information about the moving trajectories and the behavior of the birds. In this paper DBSCAN algorithm is used to cluster the GPS data. As the initial dataset contained data from a large time period and from birds with different starting points and destinations, it did not make much sense to try to cluster it all together. Moreover, due to the nature of the DBSCAN algorithm, the processing of a large amount of points would take unreasonably much time. For this reason in this paper data collected from one bird is analyzed. The analyses of the patterns of the trajectories and bird's behavior are done based on clustering the data points from the GPS trajectories.

II. The Algorithm

In this work the algorithm used for mining the frequent patters is DBSCAN. The DBSCAN algorithm takes two parameters. The first is the parameter *minPoints* that indicates the minimum number of data points needed to form a cluster. The second parameter is *eps* that indicates the maximum radius of the neighborhood of the point. Additionally, a set of points to run the clustering on have to be given to the algorithm. The algorithm goes through all the points in the dataset and for each one of them, in case the point has not been visited yet, finds from the initial points set the neighboring points using the distance given with the *eps* parameter and Euclidean distance [4]. In case the number of the neighboring points found is equal or greater than the number specified with *minPoints* parameter, the cluster is formed and the search continues with finding neighbors for the neighbors of the previously mentioned point. A point that has already been added to a cluster cannot be added to another cluster. In case there are not sufficient amount of points to create a cluster, the previously mentioned point is labelled as noise. This process continues until all the points in the set have been processed. The advantage of this algorithm is that it is easy to implement and the parameters are flexible in allowing to choose the size of the clusters to be created. One of the disadvantages of this algorithm is its rather poor performance. The average run time complexity of a single region query is O(log n). For each of the points of the database, we have at most one region query. Thus, the average run time complexity of DBSCAN is O(n * log n) [1]. However, when the number of the points in the set is large, the time required to run the algorithm increases rapidly.

III. The Implementation

The selection of this parameter depends on the size of the input data to a great extent. In case the amount of points in the data set is small, also the minimum number of points required to form a cluster should be small. If the number of points in the data set is big and the minimum number of points to form a cluster is too small, the algorithm returns noise points as clusters and the further analyses is more complicated.

A. Selecting suitable parameters

Perhaps the most important next to selecting the source data is the selection of the suitable parameters. Finding the suitable parameters for DBSCAN algorithm is not a trivial problem. There have been research papers written about it [5] as well as libraries [6] created to enhance finding suitable parameters for the input dataset. However, in this work the suitable parameters were found by first analyzing the distances between different points of the input dataset and secondly running the algorithm multiple times with different value combinations of *minPoints* and *eps*. As with most of GPS data, also in this case visualizing the results has a great part in helping to decide whether the

parameters are suitable or not. In this paper the parameters are chosen like the following: *minPoints* = 5 and *eps* = 10.0. This means that the point under processing has to have at least 5 neighbors for the cluster to be formed. These neighbors have to be located within the radius of 10.0 kilometers from the point.

B. Database

As the size of the data to be processed was rather large, it seemed reasonable to store both the source data and the clustering results in separate tables of a database. In this work PostgreSQL [7] database was used to store the data.

C. Visualization

The visualization of the results was done using QGIS visualization tool that enables to create layered maps where each of the layer can be created from the data extracted from the database.

IV. Results and Analyzes

It is difficult to detect patterns and staypoints from the input data of one week visualized on a map. The raw data of one week is visible on the figure 1.

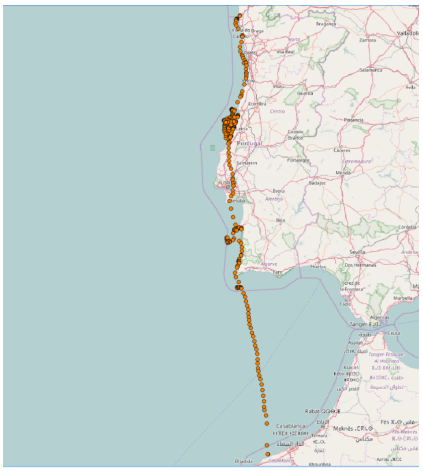

Fig. 1. The trajectory travelled by the bird in one week with all data points.

With the help of clustering the noisy points can be left out from the map and points can be clustered. Clustering helps organizing and analyzing the dataset of points. The points collected during the flights meant to move forward are not represented as clusters as the movement was fast enough to leave only few points on the track. This helps to reduce the noise and makes the results more visible. Next, the movement of the bird during one week was analyzed day by day. The clusters of one day are visible on the figure 2.

The dense areas of points above the sea can be described as the areas where the bird was hunting for food. The dense areas on the land can be described as the areas where the bird stayed for night. Clustering the data points gave as a result two different kind of patterns. It is possible to see that the bird usually has two types of daily behavior: either the bird stays at the same place for all day as can be seen on firgure

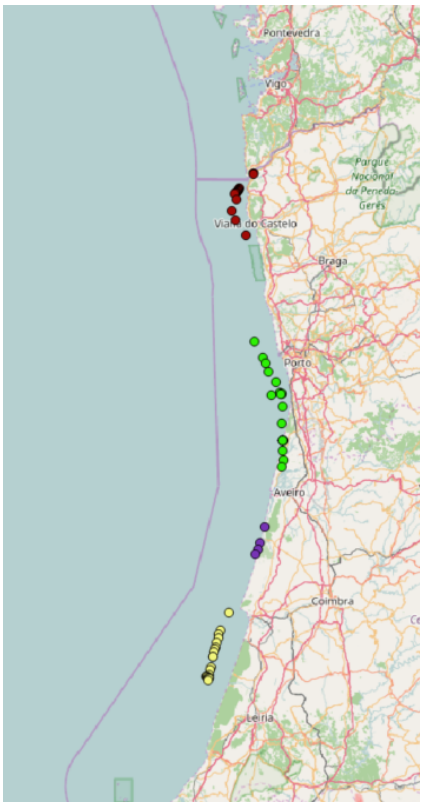

Fig. 2. The clustered data points of one day.

3 or moves forward making breaks on the sea as can be seen on figure 4.

Figure 4 represents a day from which the DBSCAN algorithm detected 4 clusters. Having clusters detected separately, it is possible to distinguish them during visualization.

Without clustering the movement pattern would not look so obvious. On figure 5 all datapoints of the same day without clustering can be seen.

Another thing that can be noticed from clusters is that the bird tends to spend larger time periods in national park areas. This behavior pattern can be detected comparing the clusters of different days. It can be seen that these clusters overlap, thus the bird spent various days at the same location. Clustering the data makes it easier to distinguish one day from another and this makes daily movement patterns more visible. The data what would seem as one densed area can be divided into layers. This kind of cluster distinction can be seen on figure 6.

Clustering showed that the bird followed similar travelling patterns throughout its journey. Clustering erases a great amount of noise points and brings out the densest areas on the map enhancing the detection of the similar trajectory segments.

V. Conclusion

In this paper DBSCAN algorithm was used to mine frequent trajectory patterns from the GPS data collected from birds. As raw GPS data is noisy, it is hard to detect patterns from it without trying to organize the data. The algorithm clustered the most frequent locations of the bird and made it possible

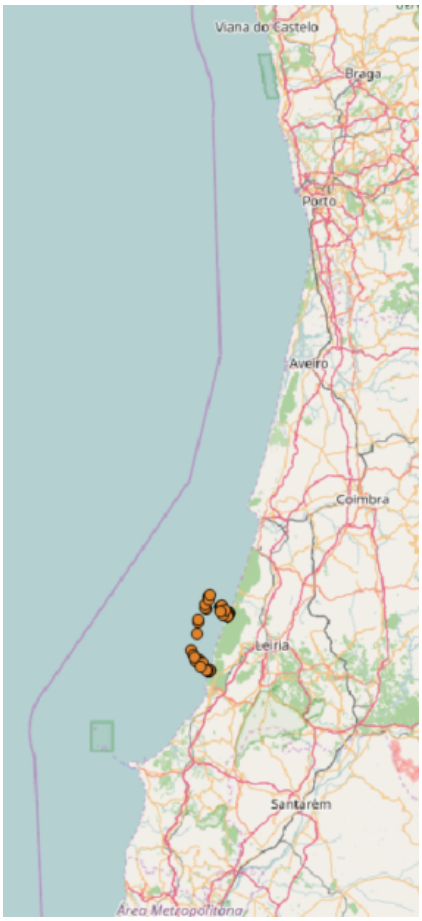

Fig. 3. Example of a day when all points belong to one cluster.

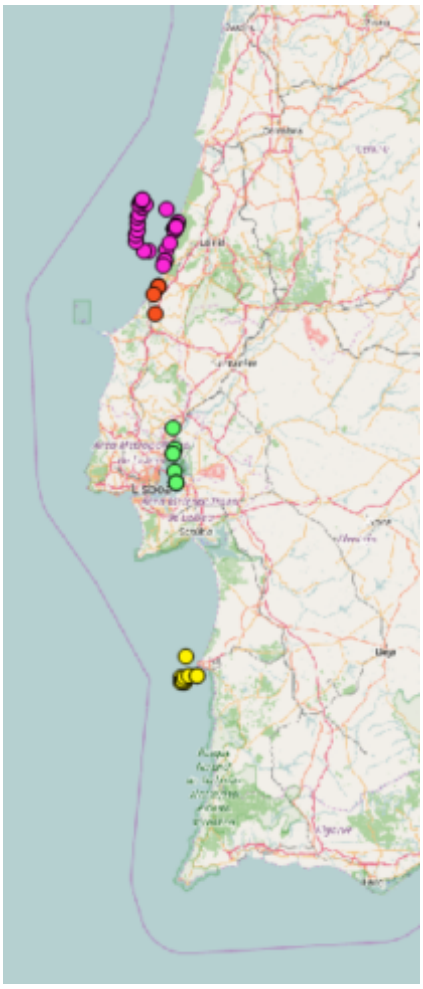

Fig. 4. The clustered data points of one day.

to analyze bird's daily behavior as well as the behavior during longer period of time. This information can be very useful for ornitologs that are interested in observing the birds. For better results it would be possible to analyze more data from a temporal, geographical and displacement aspects. To make stronger conclusions based on the clustering, it would be necessary to have data form a larger amount of birds.

REFERENCES

[1] Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. *A density-based algorithm for discovering clusters in large spatial databases with noise* Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press. pp. 226–231. (1996)

[2] *QGIS Geographic Information System* Source: http://www.qgis.org/en/site/ Last visited: 28th November 2016.

[3] Stienen EWM, Desmet P, Aelterman B, Courtens W, Feys S, Vanermen N, Verstraete H, Van de walle M, Deneudt K, Hernandez F, Houthoofdt R, Vanhoorne B, Bouten W, Buijs RJ, Kavelaars MM, Mller W, Herman D, Matheve H, Sotillo A, Lens L *Bird tracking - GPS tracking of Lesser Black-backed Gulls and Herring Gulls breeding at the southern North Sea coast* Research Institute for Nature and Forest (INBO). Dataset/Occurrence. http://doi.org/10.15468/02omly Data paper: http://doi.org/10.3897/zookeys.555.6173 (2014)

[4] *Euclidean and Euclidean Squared* Source: http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Euclidean_and_Euclidean_Squared_Distance_Metrics.htm Last visited: 27th November 2016.

[5] Karami, A.; Johansson, R. *Choosing DBSCAN Parameters Automatically using Differential Evolution* International Journal of Computer Applications 91(7):1 –11, (2014).

[6] Litouka, A. *Spark DBSCAN* Source: https://github.com/alitouka/spark_dbscan/wiki Last visited: 27th November 2016.

[7] *PostgreSQL database* Source: https://www.postgresql.org/ Last visited: 28th November 2016.

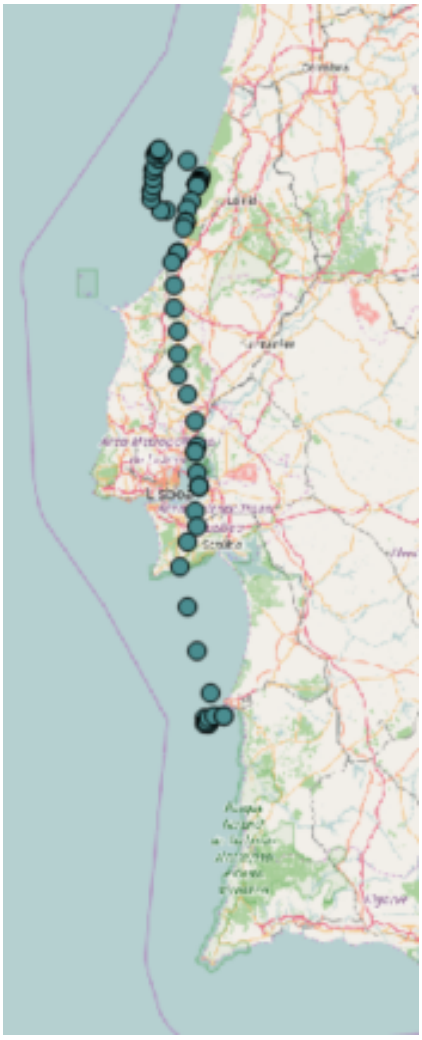

Fig. 5. All the data points of one day without clustering.

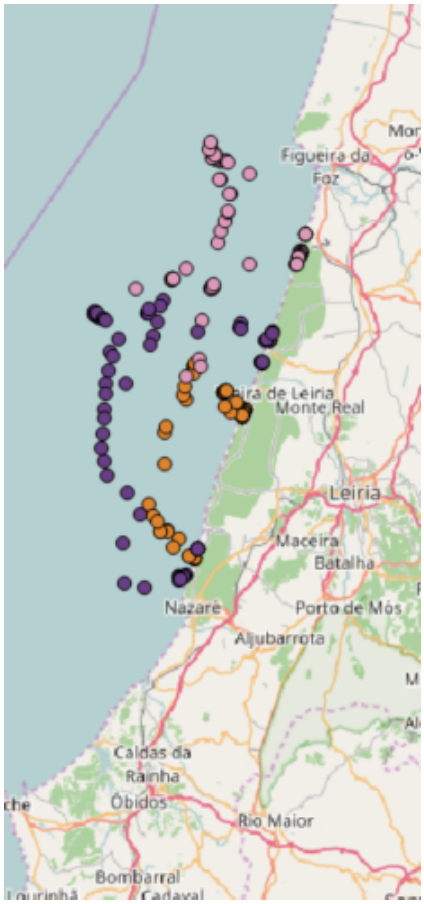

Fig. 6. Overlapping clusters of three days.