Experiment (2): Extracting Points of interest (preprocessing)

Introduction:

After extracting the stay points (locations) which the user spent time, the extracted points needed filtering, to suppress the noise, in this case the noise is considered as the stay points with low visiting frequency, these locations can't be predictable, the other points are the important locations that the user regularly or frequently visit, which we called points of interest.

The noisy stay points, can be considered for example as emergent locations like going to hospital in case of sickness, going to a concert, or going to an event, the task of predicting the human behavior (location is considered as behavior) may need more data about the users, such as phone calls duration, proximity, social interaction or tweets, gathered with location data. These complete data sets, can help predicting irregular behaviors of the user.

Problem definition:

Suppressing the noise using A clustering algorithm to combine the points located in the same region into single centered locations, and consider single stay points as noise (locations visited once) with probability of visiting again tends to zero, reducing the prediction accuracy.

Figure 1 shows the stay points of all the trajectories of Geolife for one user, before apply the DBSCAN algorithm. The map shows a concentrated area and a scattered points which DBSCAN consider them as noise.

In this experiment we will apply DBSCAN on every user separated, to find the POIs for the users, so that we can use it for individual predictions and inferences, we can also later measure the similarities between users in order to recommend new places for the users based on the social connections, and by applying DBSCAN for all users combined, we can identify the most popular places in the dataset, which are useful for the recommendations.

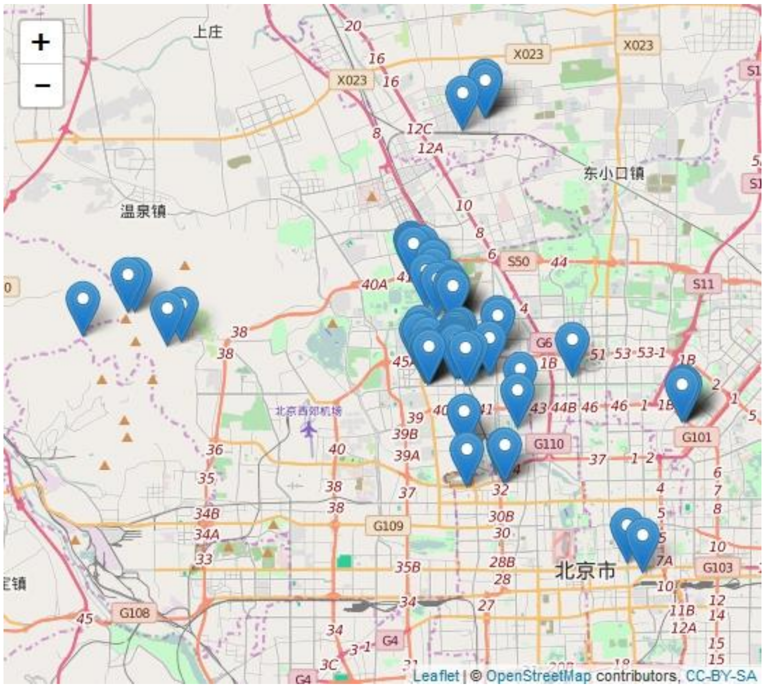

Figure 1 shows a user Stay points altogether of all his/her trajectories

Algorithm:

We are extracting the important points (points of interest) using the DBSCAN (Density-based spatial clustering of applications with noise) algorithm, it is a clustering algorithm to group nearby (distance) points into clusters, it depends on two parameters: minimum number of points to create a group, the maximum distance between each two points in the same group. Each point in the cluster must satisfy two conditions: first, all points should be mutual density connected, second a point is belong to the cluster if at least one point of the cluster maintain a distance less than the maximum distance threshold.

DBSCAN algorithm group the points which satisfying the two thresholds, the other points are considered noise and can be easily removed, after applying this algorithm we get the mean point of each cluster and consider it as the point of interest, in figure 2 we focus on two cluster of the same user in figure 1 after applying the DBSCAN algorithm, we showed the colored points as the stay points that the user visited more often, and the black ones are the stay points that considered as noise. In figure 3 we showed the map with the resulted points of interest, the center point of every cluster.

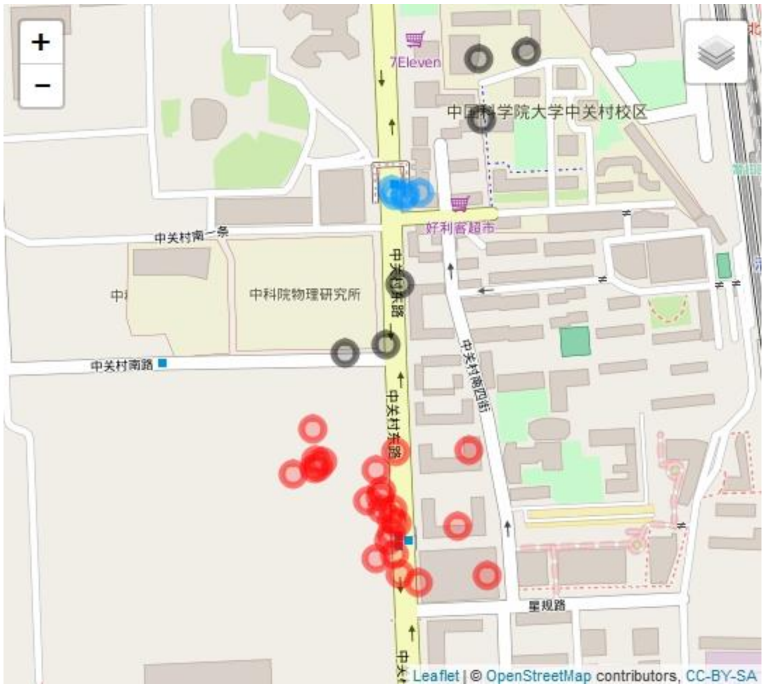

Figure 2 shows two clusters of a user where he/she visited Stay points more than four times (red and blue), the black points are the noisy stay points.

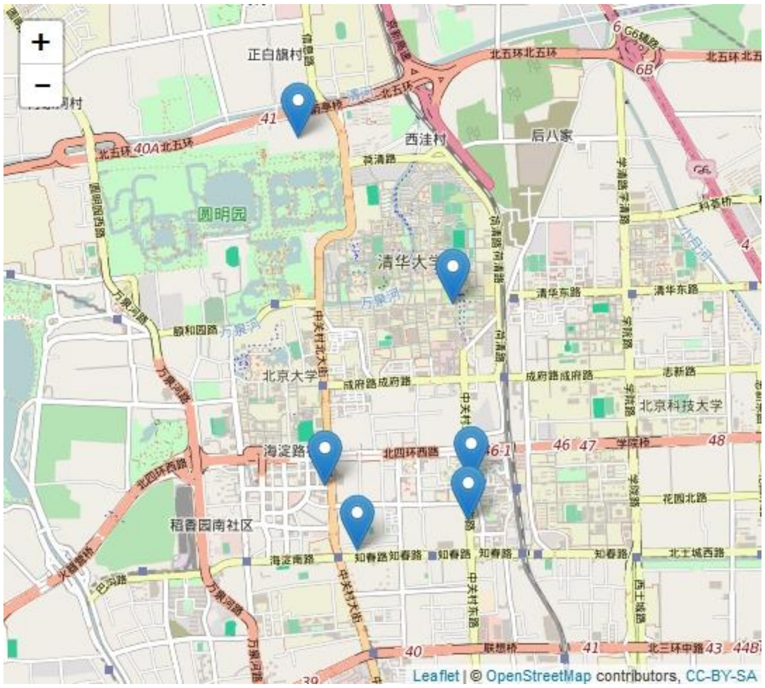

Figure 3 the resulted points of interest

Results:

We applied the DBSCAN for the first ten users from Geolife dataset to compare with the results of report [1], also we used a threshold minimum number of points of 4 and a maximum distance of 100 meters between points, to find the results is comparable but not the same as the report used a cleansing algorithm first, table 1 shows the comparison between the result we apply and the report's result

User	Stay Points	Extracted POIs	Report's POIs
1	434	13	14
2	99	6	6
3	248	10	9
4	973	25	22
5	1175	23	22
6	173	8	8
7	40	2	1
8	152	4	4
9	67	3	4
10	95	4	2

Table 1 the results of DBSCAN for ten users compared with the report [1]

References:

[1] Peixoto, Douglas Alves, and Lexing Xie. "Mining trajectory data" (2013).