Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 7, Issue. 3, March 2018, pg.51 – 67

Self-Adaptive Traffic Recommendation System Beijing City as a Case Study

Aiman Abu Samra[1], Ibrahim Al-Sharif[2], Ahmed Skaik[3], Fatma El-Rebai[4], Haneen El-Talli[5], Mohammed Shbair[6]

Computer Engineering Department, Islamic University of Gaza, Gaza, Palestine
aasamra@iugaza.edu.ps, Ibrahimht2000@gmail.com, ahmskaik@gmail.com, fatma1211994@hotmail.com, haneentalli@gmail.com, mwshbair@gmail.com

Abstract— Due to the widespread use of smart phones, location-based services (LBS) has become important in all aspects of human's life. LBS offer many advantages to the mobile users to retrieve information about their location. Some LBS services use the smartphone's location and service provider information to offer directions, targeted recommendations, or other location-specific information to the user. In this paper we proposed a novel location-based method that provides a traffic recommendation based on community contributed and collaborated movement history. Proposed model is designed to perform some computational processes on the data collected from real users and decide which path is better to follow to reach the desired destination from a given source location. Proposed method mine places and paths between places from the raw data and analyse these data to provide a recommendation to reach a specific target location. Proposed method is adaptable for new movement information. We use a dataset for Beijing city as a case study to proof our proposed method.

Keywords— Location Based Service (LBS), traffic recommendation system, trajectory, clustering, learning algorithm, self-adapting.

I. INTRODUCTION

The Location based Services (LBS) offer many advantages to the mobile users to retrieve the information about their current location and process that data to get more useful information nearby their location. With the help of A-GPS in phones, LBS can be implemented on Android based smart phones to provide these value-added services: advising clients of current traffic conditions, providing routing information, help people find nearby locations.

Location Based Social Networks (LBSN) is not only traditional social network with ability to add location-tagged photos, share current user location or comment on an event at specific place, but also consists of the new social structure made up of individuals connected by the interdependency derived from their locations in the physical world as well as their location-tagged media content, such as photos, video, and texts. Here, the physical location consists of the instant location of an individual at a given timestamp and the location history that an individual has accumulated in a certain period. Further, the interdependency includes not only that two persons co-occur in the same physical location or share similar location histories but also the knowledge, e.g., common interests, behaviour, and activities, inferred from an individual's location (history) and location-tagged data [1], [2].

They are three directions to deal with the location concept, as a point of latitude and longitude, as a region and as a trajectory which is a collection of points with their timestamps. For example, Foursquare LBSN which

© 2018, IJCSMC All Rights Reserved

consider the place as the main item from user's check-in [3], another example is Microsoft GeoLife LBSN which consider both place and trajectory as the main items [4]. LBSN provides users with places or trajectories recommendation based on other user's behaviour or based on both user history and other user's activities, for example LBSN system can recommend the fastest trajectory to reach certain place, can also recommend places to visit depending on other user's activities whom considered as similar to target user.

Recommending trajectory requires analysing big amount of data to mine the fastest trajectory to from specific location to the target location, many algorithms have been presented to analyse this type of data which consists of many points in each trajectory. Trajectory mining has many challenges, for example the different sizes of the trajectories make the similarity calculation harder, also the fact of that the one trajectory may embed many sub-trajectories complicate the analysis more and more.

There are many trajectory mining methods: clustering, classification, pattern mining, outlier detection and prediction, both clustering and classification considered as main methods [5].

At present, the trajectory clustering approaches include two types [6]: the first cluster the trajectory data based on the similarity of the full sequences. In other words, they take the whole trajectory as a unit to cluster the trajectory data. Those approaches have good effects on the clustering for the simple trajectories, however, they have negative effects for complex trajectories due to the fact they ignore the local detail sequences. The second type cluster the trajectory data based on the similarity of the sub-sequences. This means that the whole complex trajectory sequence is divided into several segments, which can be clustered with one segment as a unit. The second approaches have the ability to recognize the local features of complex trajectories.

The rest of this paper is organized as follow: The next section introduces the related work. Section three describe the motivation and problem statement. The research methodology detailed in section four. Section five listing the results while section six discuss the evaluation. Finally, section seven concludes the findings.

II. RELATED WORK

Systems that provide real time information to assist in planning routes and choosing the most appropriate paths are essential to make transport more effective.

As an alternative solution to problems related to mobility in cities, there are the so-called Intelligent Transportation Systems (ITS) which include the Route Recommendation Systems (RRS) and methodologies for congestion prediction that combine Information and Communication Technology (ICT) with Artificial Intelligence (AI) technology to improve the quality of transport systems [7].

A wide variety of ITS tools have played important roles in the effectiveness of transport. These systems provide information related to traffic, influencing in various aspects of transport in relation to urban mobility. Most of these ITS tools uses static information aided by the traffic infrastructure integrated technologies. Pires et al [7] introduced ACORoute, they propose the use of pheromone-based communication for building an ITS that offers information about real time traffic flow, taking into account the mobility of vehicles and passengers and the traffic dynamics.

The operating principle of their strategy occurs with vehicles that mark their path by dropping digital points which are perceived by all the vehicles that travel in the environment. This mechanism is used to allow the calculation of a route that avoids heavy traffic.

LBS is a concept that can be vastly utilized. LBS can be applied in public and private industry, such as emergency service in medical [8]; tracking industry [9]; navigation industry, such as digital map; payment; tourism guidance [10] and so on. It can be particularly powerful when combined with other user profile information to offer personalized and location sensitive responses to customers, this form is called the context aware system. Furthermore, the information recommendation systems based on personal interests is currently an active research area, keeping in mind that many applications have been recommended and produced by different authors and developers during last decade to effectively and efficiently control the location based recommendation system by giving different solutions. Maruyama et al.[11] propose a personal navigation system called P-Tour for tourism. Islamabad City Guide is one of the city guides. It uses static maps and pictures to show the places [12]. Byon et al. used a multi criteria shortest path algorithm which considers scenic zones, slope and crime ratio in the cost function [13]. Mckercher and Lau attempted to identify the movement patterns and styles of tourists within an urban destination [14].

Moreover, many vendors provide some geographic information services. Such as Google Maps show search results on a map interface, Yahoo! Local Maps, Google Earth and Microsoft Virtual Earth allow users to explore richer geographical contents.

TRACLU algorithm is a famous one presented as partition-and-group framework by Jae-Gil Lee et al. [15], TRACLU algorithms apply portioning process on the trajectories before clustering them by DBSCAN. Multiple similarity functions introduced, each one has different concept of the similarity based on the goal of mining, some consider the similarity between the whole trajectories and other consider the similarity between sub-trajectories also. Some functions consider two trajectories similar if they are similar in their destination, source or both, others consider the similarity as the same route and destination, while others consider the similar as the

© 2018, IJCSMC All Rights Reserved

same direction [16]. Closest-Pair Distance for example assumes the same length of the trajectories and considers the similarity as distance between points in the two trajectories [17].

In this paper, we show that the community contributed and collaborated database can tackle these issues and provide an efficacious solution to automatic locations analysis.

III. MOTIVATION AND PROBLEM STATEMENT

Many of LBS applications providing information about locations and places and offering different kind of tracking mechanisms. However, unfortunately the proper GPS services is not completely supported in Gaza Strip, which could bring an imprecise location, places and directions detection. For instance, we attempt to acquire the best path from Al-Azhar square to Lababiddi square using google map application installed in Android mobile, and incorrectly it gets wrong directions for some particular streets. Fig 1 shows the right path (green), and the wrong google targeted path (red), since the red path is a one-way street path.

Fig. 1 Longer path suggested by google

Different problems are arising while users attempt to use LBS systems which we can address and avoid by developing the proposed system.

The user can use this application obviously for different purposes like searching a location, calculate distance between two locations and acquiring the best path between two locations considering a couple of circumstances like (weather status, traffic jam, and accurate street directions),

The approach in this paper is to develop an application based on a mobile device running the Google Android platform. The device used is 2.5 G technology. Definitely this is the key point of our work, where users unluckily couldn't be connected to the internet all the time. Furthermore, our intent and main purpose of this project is to come up with an Android application which efficiently helps people to create a commutative and cumulative data warehouse for their movements and activities. And the system will manage and manipulate this data along with regular intelligent algorithms to be available on demand.

Moreover, the system will produce a best-path recommendation guide, and helps the user navigate the history of their saved activities.

IV. PROPOSED METHOD

In this section we describe the proposed method which gives path recommendation based on users' collaborative GPS information.

The proposed method consists of four main steps as shown in Fig. 2: the first one is mining turning places using unsupervised learning algorithm. The second is segmenting and analysing paths based on turning places. The third one is recommend the optimal path for the user in a specific location and a specific time. The last step is adapting the data and updating the map after receiving new users' GPS information.

© 2018, IJCSMC All Rights Reserved

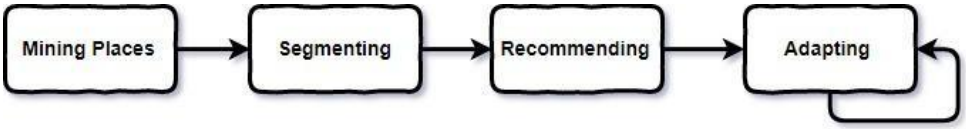

Fig. 2 The main steps of the proposed method

GPS information collected in a way that it may be for one hour time, one day or even one week. To be able to analyse these information we segment each path to chunks based on a criterion. Criterion may be Time or behaviour to segment a set of consecutive GPS location points.

In the proposed method we mainly consider the path direction change (bearing) as the core criterion to detect turning points which form the turning places, these turning places are used to segment paths.

A. Mining Turning Points

Turning point is a point where the direction of a path changes relative to the earth north direction, while turning place is an area of path/s turning points. Turning place may be a crossroad, an intersection, a curve or any direction change location. To extract turning points we used bearing concept, where the bearing of a point is a clockwise direction to the north of the earth from the earth center. To detect the direction of a path in specific scope (between P2 and P1) we calculate the difference bearing between current point (P2) and previous point (P1). See Fig. 3.

Bearing between two points =

$$\theta = \text{atan2}\left(\sin \Delta\lambda \cdot \cos \varphi 2 \, , \cos \varphi 1 \cdot \sin \varphi 2 - \sin \varphi 1 \cdot \cos \varphi 2 \cdot \cos \Delta\lambda \right) \ldots (1)[18]$$

Where:

$\varphi 1, \lambda 1$ is the start point, $\varphi 2, \lambda 2$ the end point ($\Delta\lambda$ is the difference in longitude)

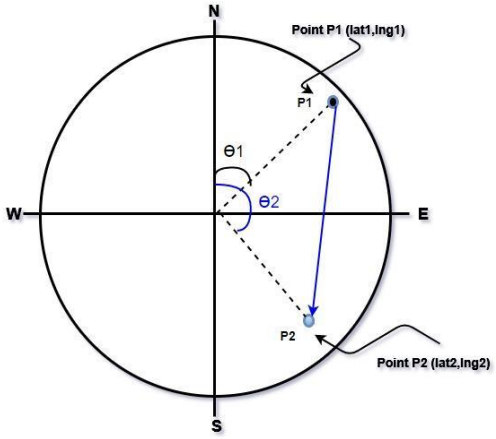

Fig. 3 Bearing of GPS points

To detect the possible change in direction like in fig. 4, we calculate the difference in direction between the current direction and the previous direction.

Previous direction (**pd**) = Bearing (P2) - Bearing (P1) ……..(2)
Current direction (**cd**) = Bearing (P3) - Bearing (P2) ……..(3)
Change in direction(**cid**) =Current direction(**cd**)–Previous direction (**pd**) …….. (4)

© 2018, IJCSMC All Rights Reserved

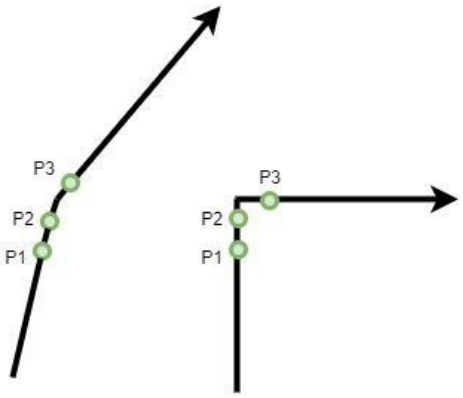

Fig. 4 Consecutive points with change in direction

If the change in direction is greater than some threshold ɵ, then we consider that there is a turning point. In fact the real change in direction happens in somewhere between the current point and the previous point, We choose the previous point as the turning point, because we don't need to generate new points. Fig. 5 shows example cases of detecting turning points in the same place but from different taxis.

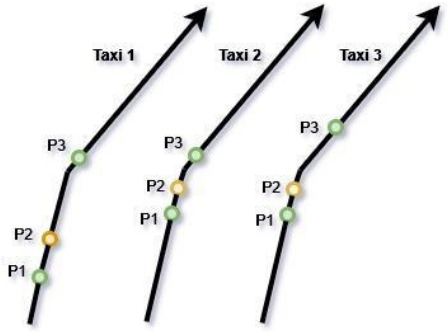

Fig. 5 Different cases of detecting turning point P2

To prevent extracting unnecessary points for the same curve, we check the next point only if its distance with the current point is more than EPS, this check also eliminates the stay points problem, as in fig. 6.

We eliminate the noise points by checking the speed, if the speed is not reasonable, then we neglect that point. Because noise point jumps suddenly far away from the real path points, as in Fig. 7.

If (d > EPS) and v < 2000 m/s and (**cid** > ɵ) then the previous point is turning point. ..... (5)

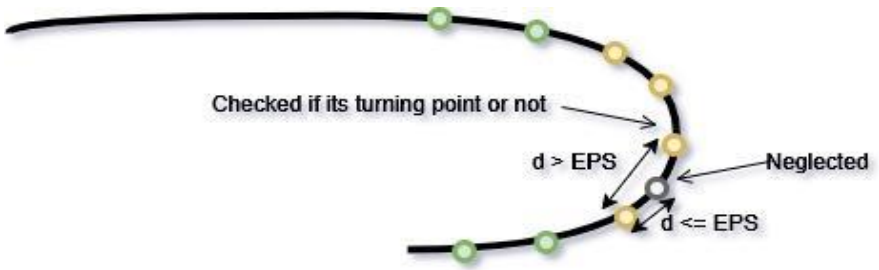

Fig. 6 Curve case truing points mining

© 2018, IJCSMC All Rights Reserved

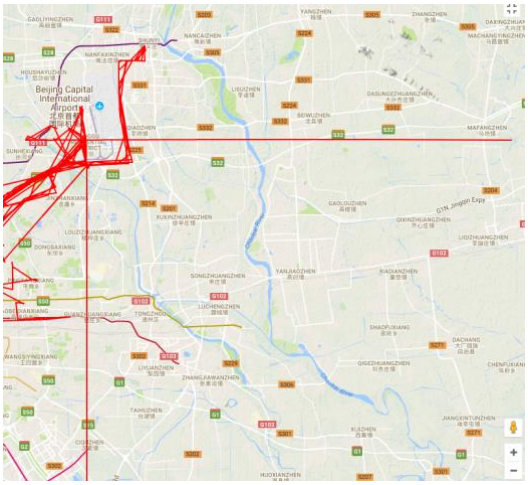

Fig. 7 Sample of noise points

B. Mining Turning Places

Many paths may pass through the same place with different points values closed to each other. These points are considered as one turning place with specific radius (cluster). We applied unsupervised learning algorithm to cluster these points in one place. Density-based clustering algorithm (DBSCAN) is applied to all extracted turning points to produce turning places [19].

Density-based clustering algorithm is considered as one model of various clustering analysis models It defines clusters as connected dense regions depending on two important parameters, first is (σ) which represents the radius around the point to search for the number of neighbourhoods. The second is (ξ) which represents the minimum number of neighbourhood points required to consider this point as a part of a cluster. Choosing these two parameters majorly affects the clustering quality [20].

In most applications (σ) and (ξ) estimation are considered as the main disadvantage of DBSCAN algorithm.

Fortunately, in our application (σ) and (ξ) are two advantages because even one turning point should be consider as a cluster not noise (ξ) =1, also we can estimate directly what is (σ). For every cluster we choose medoid point as cluster representative point. (medoid is one of the points not a new point) as in Fig. 8.

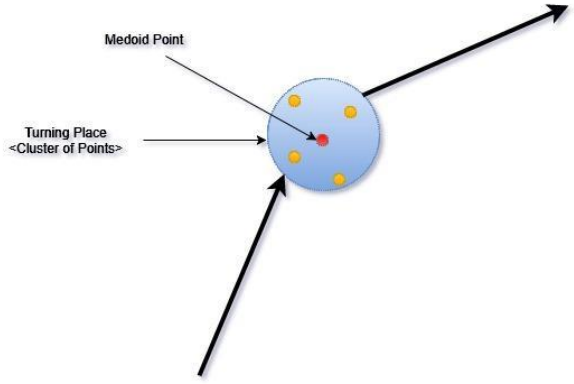

Fig. 8 Turning place with representative point

Segmenting all paths in the dataset is based on passing through the turning place (cluster of turning points). Cluster radius is the maximum distance between cluster points and the representative point.

C. Segmenting Paths:

We segment paths according to the extracted places. To segment a path we check turning points of the path. If the distance between the current point and any representative point less than or equal to its cluster radius, then this point is considered to be a location where the path will be segmented.

© 2018, IJCSMC All Rights Reserved

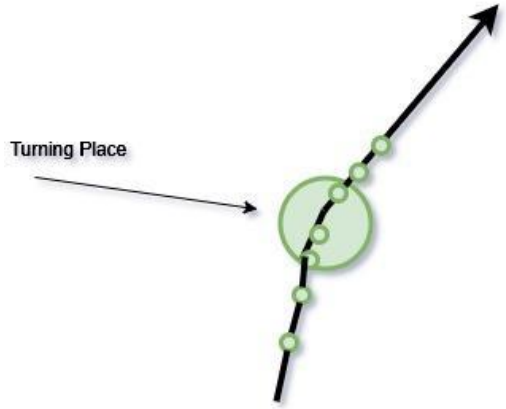

Fig. 9 More than one point pass through the place

In case if the cluster is only one point and the cluster radius is zero, or in case if the cluster radius less than (σ), we compare the distance with (σ). Another case if two or more consecutive points of the path are located in the cluster radius, this produces a segment from turning the place to itself. So, while segmenting a path, if the current turning place is the same as the previous one, then this place will be neglected.

If current place (cp) ≠ previous place (pp) and d<= max(cluster radius (cr), EPS) then the current point is considered as segment point. ……. (6)

One last note regarding to segmentation is that, because we have a huge number of representative points, we don't compare every point in the path to all representative points. Instead we first, check the turning points of the path, then compare each turning point to all representative points to segment the path.

D. Recommendation

To find the optimal path in a specific time for a specific user, we enhance the A* algorithm [21]. As we have a huge number of places and edges in our map, so we can't use any algorithm that visits all places (nodes), because this will cause low performance and consume more memory.

We need to check only places and paths in a scoped area, to achieve that we start, like A*, from the target places which prepare the next possible nodes. The A* then calculates the direct distance to the source node, then takes the node with minimum distance and repeat the previous steps until reach to the source node. But this way is not appreciate in our case, because there is no guarantee that the closest point to the source will lead to the shortest path. Noting that we don't use the Cartesian distance here because we have to consider the earth curve. Fig. 10 shows that the path with weight 1200 leads to the shortest path while the paths with weight 655 and 1000 don't.

So we make changes from the A* algorithm The proposed method doesn't measure the direct distance to the source place from the first step, instead we repeat the first step procedure for every place detected from the first step, and repeat the procedure again on the produced places in the second step. As fig. 10 shows, each iteration consists of three steps, in the first step we visit node2, node3, and node4, and in the second step we repeat the first step for node2, node3 and node4. From nod2 we visit node5 and node6, from node3 we visit node7 and node8 and from node4 we visit node9. In the last step we repeat the same procedure on the resulting nodes from step3, the final resulting nodes from the last step in the first iteration are (node10 to node 21).

© 2018, IJCSMC All Rights Reserved

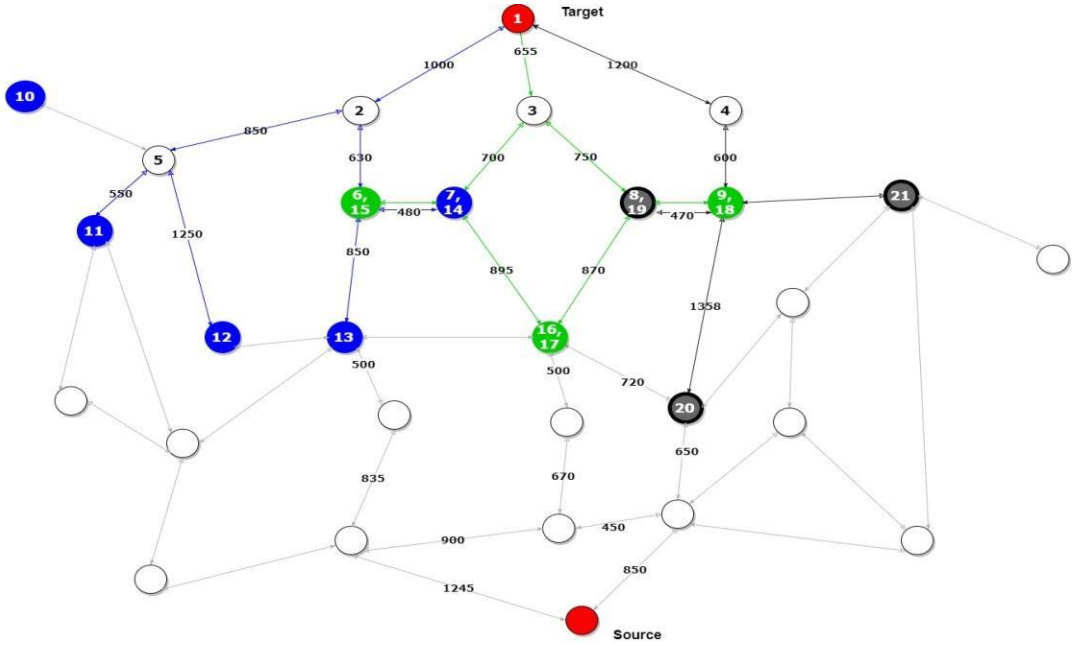

Fig. 10 First iteration of the search method

at the end of the third step of the iteration we calculate the direct distance to the source node from each place resulting from the first iteration then choose the most closest three nodes. As in fig. 10, we have 11 leaf nodes, three of them have the minimum direct distance to the source node: node13, node17 and node 20 (node16 and node17 are the same node). In the next iteration we neglect the other 8 nodes and repeat the first iteration procedure in the next iteration and add children of the selected three nodes, and so on until find the source. In the second iteration step2 one of the paths reaches the source before other ones, the two other paths reaches node (c) and still need more one movement to reach the source. See Fig. 11.

The stop criterion of this method is reaching the source node from all paths or reach a leaf node which doesn't lead to source from all/some paths. If the new leaf nodes diverge from the source in three iterations then there is no segment leads to the target source node.

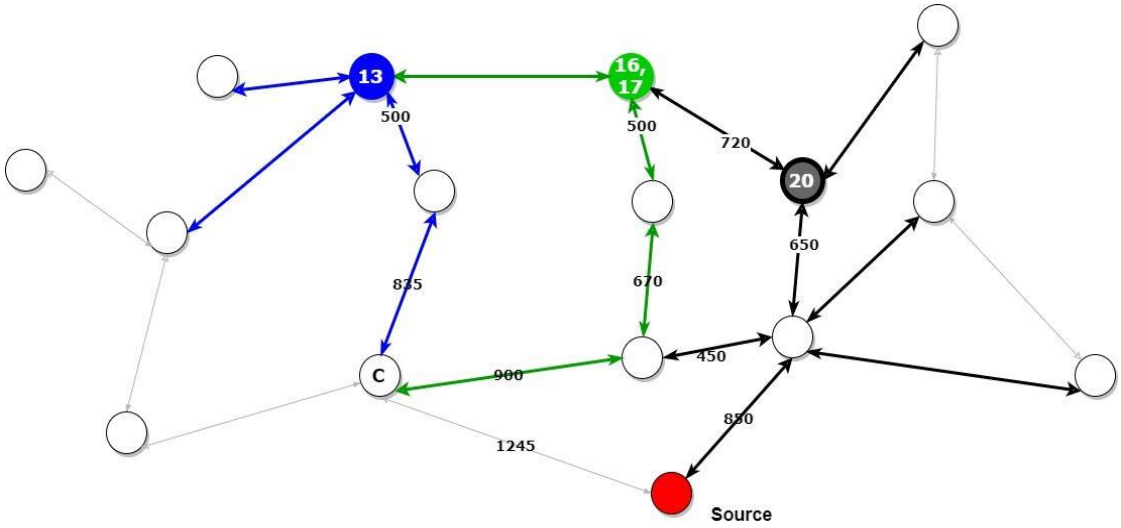

Fig. 11 The second iteration of the search method

Finally we retrieve the time for all paths which reach to the source node in the similar time and date, then introduce the recommend path (sequence of segments) based on the average time of each segment and the minimum total time in addition to the distance of the recommended path.

© 2018, IJCSMC All Rights Reserved

The proposed algorithm is scripted in the following pseudocode.

- At first call segment is equal null,
- After getting the tree of possible paths, calculate the time-based shortest path.

```
Tree.addRoot(destination)
getOptimalPath (source, destination, segment)
        While(segment.start != source)
         //Get all segments which end with the destination
         temp1= getConnectedSegments(destination)
         step1 += temp1
         Tree.add(temp1)
            //Get all segments which end with start of every segment in step1 list
            // For each segment segment1 in step1
                  While(segment1:step1 != source)
                  temp2 = getConnectedSegments(segment1.start)
                  step2 += temp2
                  Tree.add(temp2)
                     //Get all segments which end with start of every segment in step2 list
                     // For each segment segment2 in step2
                           While(segment2:step2 != source)
                              temp3 = getConnectedSegments(segment2.start)
                              step3 += temp3
                              Tree.add(temp3)
       //Get the three segments which have the minimum cost and direct distance to the source node
        closer= getThreeMinimum(step3)
         //Recursion call for every segment in closer segments list
            foreach close_segment in closer
                    return getOptimalPath (source, close_segment.start, close_segment)
```

E. Adapting

In this section we show how the proposed algorithm adapt/update the paths segmentation upon receiving new path information from users. Since our algorithm based completely on users information collaboration, it should be adapted according to any new information in continuous manner. When new path information uploaded to the server from any user, the proposed algorithm detects all turning points in the new path, classify each one to one of the existing clusters, update the cluster information and add the path segments information like: time, date and weather to the system.

If even one of the new turning points couldn't be classified to any existing cluster, we consider it as new cluster. After adding a new turning place as a new cluster, we check if this new turning place holds updates to any of the existing segments, we check if the new turning place splits one or more of the existing segments.

If the a new segment starts from the same place as an old segment, but the new segment end doesn't exist as a start of other new segments, then no need to update the old segment. But if exist then there is a possibility for updating. Fig. 12 and Fig.13 show two cases of which there is a possibility for no need/need of updating the segments.

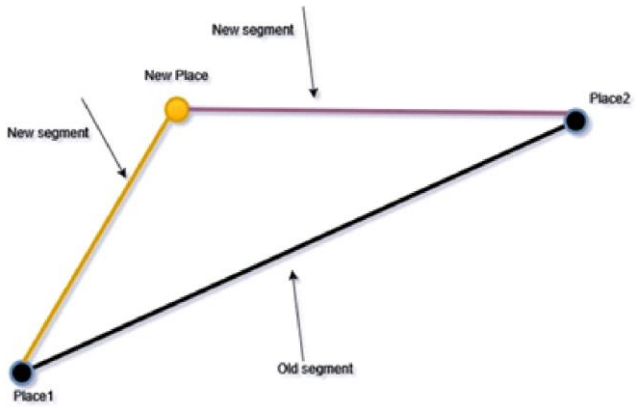

Fig. 12 Case of no need to update the old segments

© 2018, IJCSMC All Rights Reserved

In the case in Fig 12 and fig. 13 we retrieve all old segments which satisfy this condition, then cluster them based on the distance and direction of the segments. If clusters found, then segmentation is required for all old segments in the result cluster.

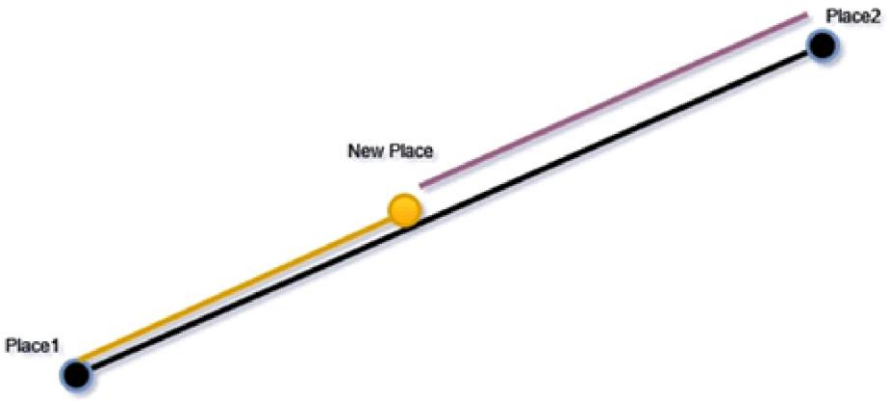

Fig. 13 Case of need to update the old segment/s

V. DATASET AND CHARACTERISTICS

Microsoft T-Drive trajectory dataset contains a one-week trajectories of 10,357 taxis in Beijing, with 15 million points and total distance of 9 million kilometres [22], [23]. We used this dataset to prove our proposed algorithm, but it doesn't contain any information about the weather state.

Data format is show in Fig. 14 consist of taxi id, date time, longitude and latitude. Time interval between consecutive points is irregular and rate from few seconds to half hour. Most points have intervals from 1 minute or 5 minutes as Microsoft shows. Fig. 15 is a Microsoft graph for intervals between consecutive points.

Taxi id,	Date time,	Longitude,	Latitude
1,	2008-02-02 15:36:08,	116.51172,	39.92123
1,	2008-02-02 15:46:08,	116.51135,	39.93883
1,	2008-02-02 15:46:08,	116.51135,	39.93883
1,	2008-02-02 15:56:08,	116.51627,	39.91034
1,	2008-02-02 16:06:08,	116.47186,	39.91248
1,	2008-02-02 16:16:08,	116.47217,	39.92498

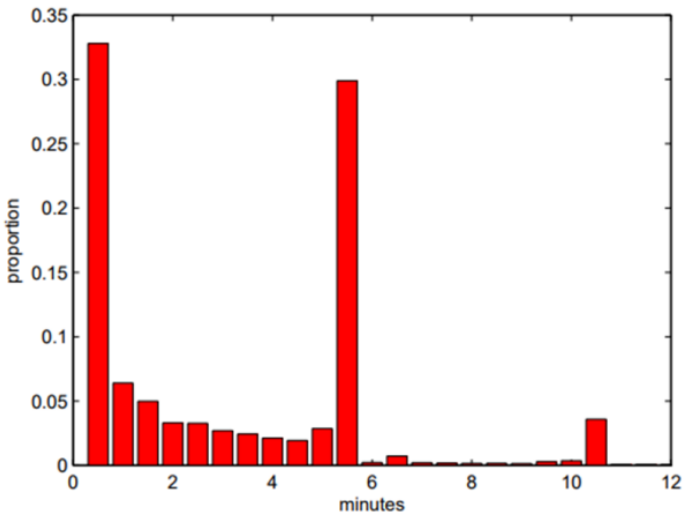

Fig. 14 Microsoft graph of the points intervals

© 2018, IJCSMC All Rights Reserved

Irregular time periods of points, causes jumps over the roads, see Fig. 15.

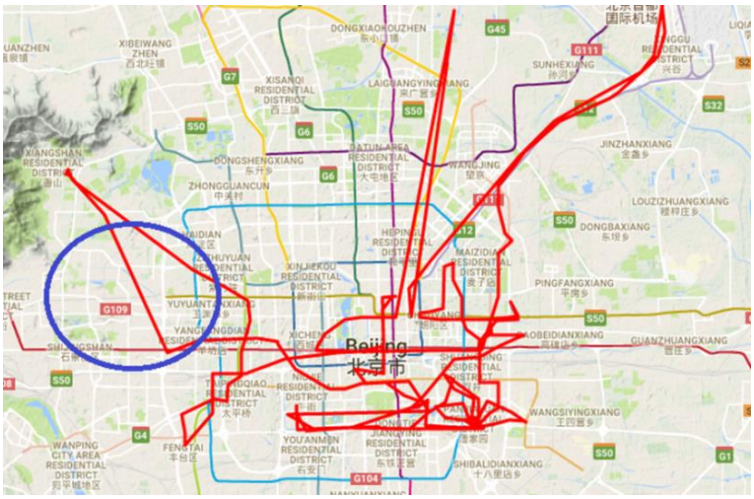

Fig. 15 Effect of irregular point's time

Another characteristic of this dataset is existence of noise points which cause confusion and far jumps in the path provided from the taxi, see Fig. 16. Noise points are detected and eliminated from our calculations. We will show that when explaining turning points. Last thing we toke in our consideration is the stop points which occur when the taxi stops for a while in the same location or around the same location. We eliminated this problem and didn't consider the time of stop points in the time of the path.

91,2008-02-08 17:07:28,116.52503,40.05253
91,2008-02-08 17:07:29,117.03711,40.05253
91,2008-02-08 17:08:15,116.52505,40.05256

91,2008-02-07 22:49:07,116.58973,40.03194
91,2008-02-08 04:23:12,116.58974,25.17147
91,2008-02-08 04:23:12,116.58974,25.17147
91,2008-02-08 09:45:06,116.58972,40.03184

Fig. 16 Sample of the noise points

VI. EXPERIMENTS AND RESULTS

A. Mobile Application and Web Services

In order to build an LBS system, the following components are required: mobile devices, applications, communication network, positioning component, and service servers. Mobile devices are tools used by users to access LBS services, to send requests and retrieve results. Application is the interface for users to access the LBS service. Communication network refers to the mobile network or Wi-Fi internet connection which transfers service request from user to server and requested information back to the user. A positioning component is usually needed in a LBS application to determine the location of user's mobile device. Servers calculate positions, search for a route, or search specific information based on user's position. Fig. 17 shows interaction among these components, and the process of a LBS service.

© 2018, IJCSMC All Rights Reserved

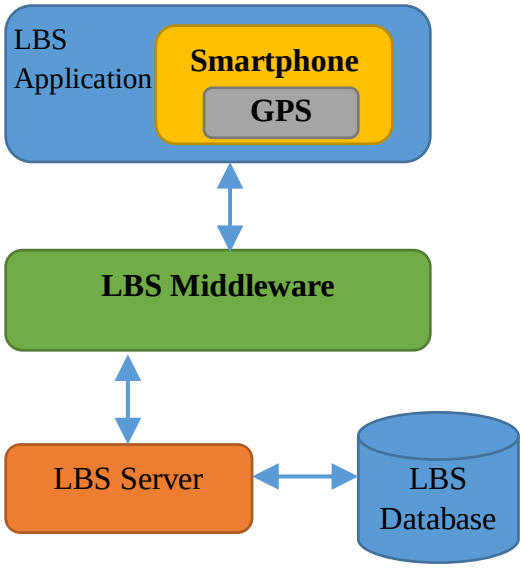

Fig. 17 components of LBS system and interaction

In the data collection mode, we used the GPS service to capture the current location every 10 seconds or 1 meter and store it on the local mobile database. Each captured location denoted by STEP node. The STEP node contains the following information: sequence number, latitude, longitude, weather status and the exact date/time. Each list of steps is linked to the user used during the capturing process.

Using the broadcast receiver, our application notified that the wireless communication to the internet is established. Once the mobile have the access to the internet, it starts pushing all the captured information other than the current trip to the server through a web service. This information prepared to be used by the backend algorithm service to classify the roads and to estimate the time required for each piece of the road.

In the testing mode, the user selects the source and destination locations on the map, then the mobile asks the backend server for the recommendation. The server should send back the best path taking into consideration all factors such as distance, traffic conjunction, weather and the user's experiences. Then the mobile application draws the recommended path on the map and print out the estimated time required for this recommended path.

The application could not be run without authentication, it asks the client to enter his username and password for the first time. All information gathered from any given mobile will be linked to the signed in user.

Web service is one of the most appropriate tool to share the information between two or more parties. It also makes easy to integrate the application with any endpoint regardless of platform. We have employed the web service to facilitate the communication between all parties in both training and testing modes.

In training mode, it is used to collect the data from the mobile application, by capturing the location in a timely manner and send all this information to the server to be stored in the database. The mobile application can use the web service to push all information stored in the local database to the central database.

This information will include sequence number, latitude, longitude, weather status and the exact date/time along with the user name collected from. The server should use this information during the estimation process.

In testing mode, the user should be able to ask for the recommendation on a given Source-Destination trip. All what is required from the mobile user, is to select the source and destination locations after then the mobile will send a request to the server through the web service to ask for the recommended path. The server will respond with a list of steps (geographical point) to be followed with the estimated time for this trip.

B. Results

We applied the proposed method one week period on 150 taxi paths information. The algorithm detected 100266 turning points, after clustering those turning points we got 67273 places. Turning places have 1-111 points and radius from 0.85-111 m, Fig. 18 shows a turning place (cluster of turning points) which have only 4 points and radius of 10.6 meter.

© 2018, IJCSMC All Rights Reserved

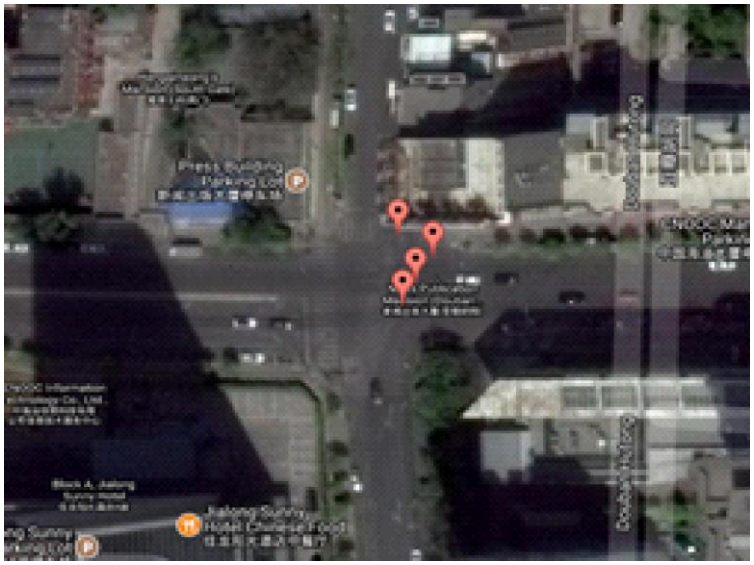

Fig. 18 Extracted (Mined) turning place with radius about 11m

While Fig. 19 shows a turning place (cluster of turning points) which have 87 points and radius of 111 meters .The difference between clusters in radius and number of points is due to the density of the points provided in these places, clusters information like radius and number of points are updated during the time since users send their paths information continuously.

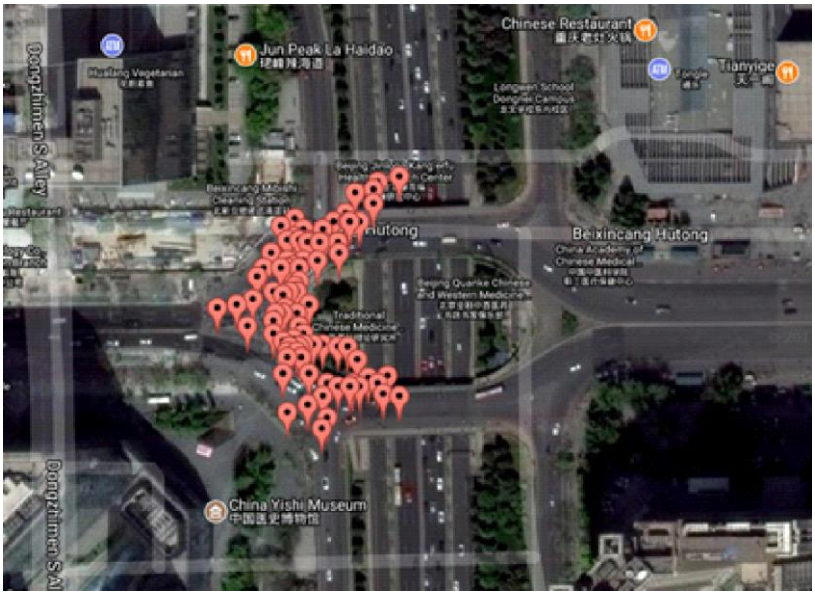

Fig. 19 Extracted place with radius of 111 meters

To show how we segment the paths, we visualized one of the paths which belongs to the taxi number 89 in Fig. 20, but since the path represents one week of movements it look very complex to understand, so we snap movements of one day then visualized these movements on the map, like in Fig. 21.

© 2018, IJCSMC All Rights Reserved

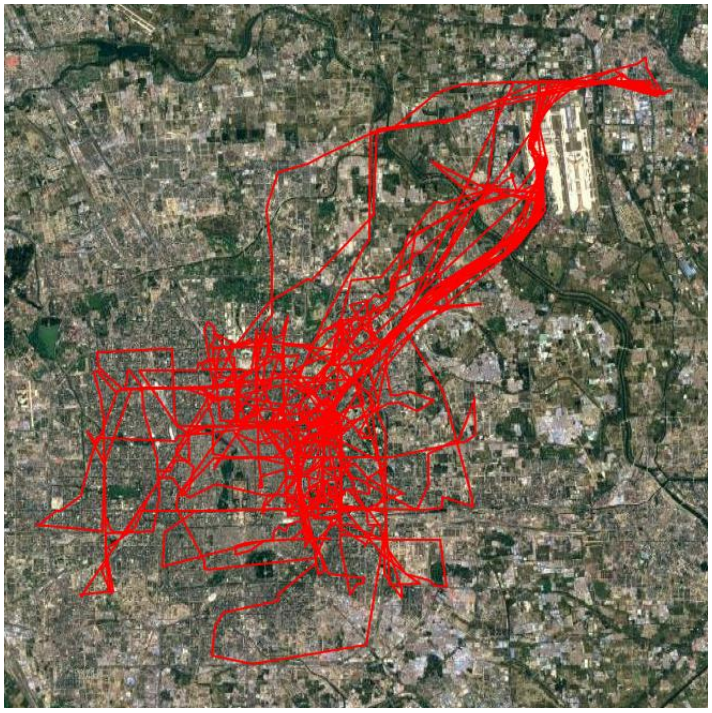

Fig. 20 Path of taxi number 89

Fig. 21 One day movements of the taxi 89

As a sample data we show the result of the previous part segmentation based on the extracted turning places, the result was 157 segments as in Fig. 22.

© 2018, IJCSMC All Rights Reserved

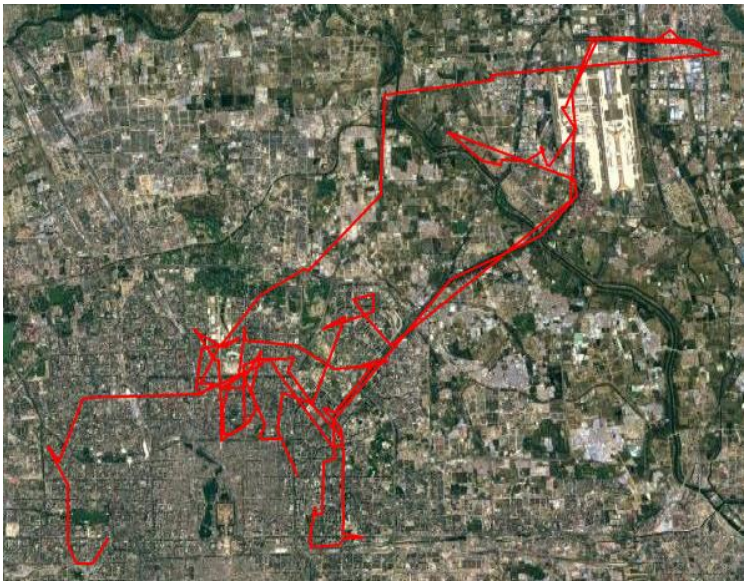

Fig. 22 Segmentation of the one day movements of taxi 89

In the fig. 22 we may note that there is something like missing segment which located below in the map, but because we deal with a part of the taxi data which happens in 2008-02-03, we cut the actual behaviour of the taxi and since there is no turning place found from the starting point to the next turning place, our algorithm neglect this part, see in Fig. 23 snap of the actual data 4 minutes before 2008-02-03 start.

89,2008-02-02 23:56:11,116.57563,40.08694
89,2008-02-02 23:58:26,116.58111,40.09751
89,2008-02-03 00:01:13,116.58955,40.11687
89,2008-02-03 00:03:28,116.61482,40.11719

Fig. 23 Movements information between two days

For the recommendation part we choose for example two places, source place= (39.99897, 116.46592) and destination place (39.87215, 116.43326) which has a direct distance of 14.36 kilometers, the two points are available on the map in Fig. 24.

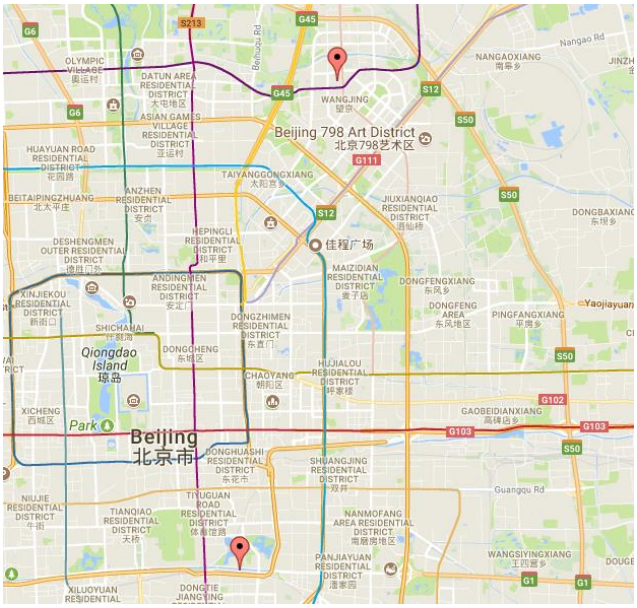

Fig. 24 Asking for recommendation to go from source point to destination one

© 2018, IJCSMC All Rights Reserved

The recommendation from our method according to the available data is one path with 9 segments with total distance of 17433.85 meters and expected time of 60 minutes, the result plotted on the map in Fig. 25.

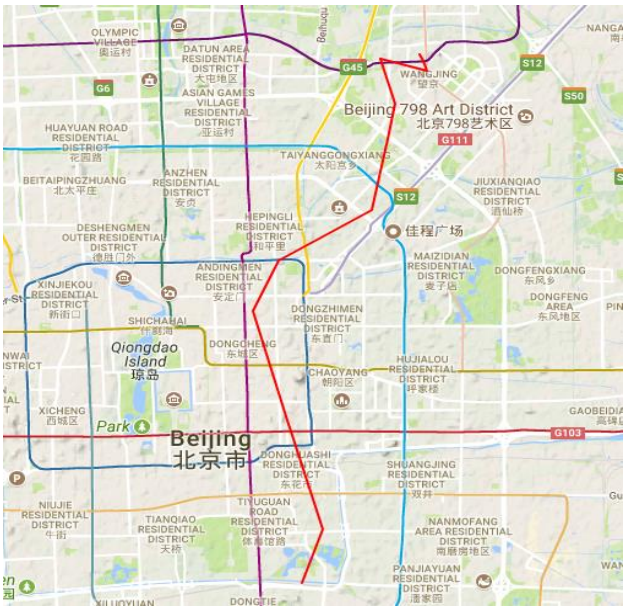

Fig. 25 Result of the recommendation process

- Segment 1: 1499.31 meters with average time of 3.15 minutes,
- Segment 2: 6068.67 meters with average time of 14.016666 minutes
- Segment 3: 1473.51 meters with average time of 6.1666665 minutes
- Segment 4: 2625.29 meters with average time of 13.95 minutes
- Segment 5: 2896.30 meters with average time of 6.1666665 minutes
- Segment 6: 1246.36 meters with average time of 3.8833334 minutes
- Segment 7: 1137.61 meters with average time of 6.15 minutes
- Segment 8: 285.64 meters with average time of 3.8833334 minutes
- Segment 9: 201.11 meters with average time of 3.3666666 minutes

The first look to the results shows that the number of segments is 8 not 9, but if you zoom in the map we find the last segment which appear in Fig. 26.

Fig. 26 Zoom on small segments

© 2018, IJCSMC All Rights Reserved

VII. CONCLUSION AND FUTURE WORK

In this paper we proposed a novel location-based method that provides a traffic recommendation based on community contributed and collaborated movement history. Proposed model is designed to perform some computational processes on the data collected from real users and decide which path is better to follow to reach the desired destination from a given source location. Proposed method mine places and paths between places from the raw data and analyse these data to provide a recommendation to reach a specific target location. Proposed method is adaptable for new movement information. We use a dataset for Beijing city as a case study to proof our proposed method. We applied the proposed method one week period on 150 taxi paths information.

REFERENCES

[1] Zheng, Yu, and Xiaofang Zhou, eds. *Computing with spatial trajectories*. Springer Science & Business

[2] Zheng, Yu. "Tutorial on location-based social networks." *Proceedings of the 21st international conference on World wide web, WWW*. Vol. 12. No. 5. 2012.

[3] Foursquare App., available: https://foursquare.com.

[4] Chris Pendleton, GeoLife 2.0: A Location-Based Social Network, available: https://blogs.msdn.microsoft.com/virtualearth/2009/02/25/geolife-2-0-a-location-based-social-network/.

[5] Mazimpaka, Jean Damascène, and Sabine Timpf. "Trajectory data mining: A review of methods and applications," *Journal of Spatial Information Science* 2016.13 (2016): 61-99.

[6] Ma, L.B.; Li, P. "Spatio-temporal Trajectory Clustering Based on Automatic Subspace Clustering Algorithm" Geogr," *Geo-Inf. Sci*. 2014, 30, 7–11.

[7] Pires, C.S., de Aguiar, M.S. and Ferreira, P.R., 2015. Mobile ACORoute-Route Recommendation Based on Communication by Pheromones. *Polibits*, (51), pp.27-32.

[8] Ahn, Jinsoo, et al. "A study of healthcare system for patient location data based on lbs." *Consumer Electronics, 2008. ICCE 2008. Digest of Technical Papers. International Conference on*. IEEE, 2008..

[9] Marmasse, Natalia, and Chris Schmandt. "Safe & sound: a wireless leash." *CHI'03 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2003.

[10] Shuja ul Hassan, Irshad Ali, " Skardu City Tourist Guide - An Android Application".

[11] A. Maruyama, N. Shibata, Y.Murata, K. Yasumoto and M. Ito, "A Personal Tourism Navigation System to Support Traveling Multiple Destinations with Time Restrictions," In *AINA* 2004, pp. 18–22, 2004.

[12] Umair Ahmed. Pakistan travel guide. Available: http://intellisenseglobal.com/, December 2014.

[13] Byon, Y.-J., Abdulhai, B., & Shalaby, A. (2010). Incorporating Scenic View, Slope, and Crime Rate into Route Choices. Transportation Research Record: *Journal of the Transportation Research Board*, 2183, 94-102.

[14] Mckercher, Bob, and Gigi Lau. "Movement patterns of tourists within a destination." *Tourism geographies* 10.3 (2008): 355-374.

[15] J. Gil Lee, J. Han and K. Y. Whang. (2007). Trajectory clustering: a partition-and-group framework. Proceedings of the 2007 *ACM SIGMOD international conference on Management of data* (SIGMOD '07), (pp. 593-604).

[16] Rokach, Lior. "A survey of clustering algorithms." *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2009. 269-298.

[17] R. Agrawal, C. Faloutsos, and A. Swami. 1993. Efficient similarity search in sequence databases. Springer, 69-84.

[18] U.S. Army, Advanced Map and Aerial Photograph Reading, Headquarters, War Department, Washington, D.C. (17 September 1941), pp. 24-25

[19] Ester, Martin, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." *Kdd*. Vol. 96. No. 34. 1996.

[20] Ibrahim Z. Alsharif, Wisam M. Ashour, Density-based Clustering Algorithms Parameters Estimation, 2015. Technical report Available: https://www.researchgate.net/publication/280920146_Density-based_Clustering_Algorithms_Parameters_Estimation

[21] Hart, Peter E., Nils J. Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths." *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968): 100-107.

[22] Yuan, Jing, et al. "Driving with knowledge from the physical world." *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011.

[23] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '10*, pages 99-108, New York, NY, USA, 2010. ACM.

© 2018, IJCSMC All Rights Reserved