Prediction of Destinations and Routes in Urban Trips with Automated Identification of Place Types and Stay Points

Francisco Dantas N. Neto[1,2], Cláudio de Souza Baptista[1], Cláudio E. C. Campelo[1]

[1]Systems and Computing Department, Federal University of Campina Grande (UFCG)
Zip Code 58.429-900 – Campina Grande – PB – Brazil

[2]Federal Institute of Education, Science and Technology of Paraíba (IFPB)
Zip Code 58.432-300 – Campina Grande – PB – Brazil.

dantas.nobre@ifpb.edu.br, {baptista,campelo}@dsc.ufcg.edu.br

Abstract. *Predicting the destination and the route that someone is likely to take is useful for various purposes, such as to prevent people from going through congested routes. Most of existing approaches to this prediction problem only consider geographic patterns within their models, although this appears to be not enough for creating a robust predictor. This paper proposes an approach to improving the task of predicting route and destination which makes use of further semantic information associated with destinations and routes, apart from location patterns. Our model does not require user's active interaction and is able to automatically identify stay points (i.e., places users visit) and type of places. We evaluated our model with real world data collected from users' smartphones and obtained promising results.*

1. Introduction

Thanks to the possibility of gathering geographic position with current *smartphones* (since they have built-in GPS device embedded), the number of location-aware systems have increased considerably. There are several benefits that location-aware systems can provide to users for helping their daily routine, such as indicating Points of Interest (POIs) around their current location, by considering their preferences, and then displaying on a map the best path to reach a POI selected by the user.

Systems that provide such location-based services are commercially used nowadays. However, there are many other topics related to location-aware systems that are still under investigation. Of particular interest in this work is the task of automatically discovering the type of place a user is located (such as "home" or "work") [Alvares et al. 2007]; and the prediction of routes and destinations [Simmons et al. 2006].

A previous step of automatically discovering the type of a place, it is the task of identifying *stay points*, i.e., the geographic region where a user is stopped. This geographic region is composed by a centroid point and a radius that associate a GPS point to the stay point. The importance of identifying *stay points* is related to the possibility of analyzing the behavior of a user in visiting specific places, enabling to understand the semantics of a place to a certain user. Identify stay points can be achieved using spatial clustering techniques, such as *DBSCAN*, *OPTICS* or *K-Means* [Aggarwal and Reddy 2013]. K-Means algorithm is a *distance-based* method, i.e., it is necessary a parameter that defines the number of clusters previously. DBSCAN and OPTICS algorithms are *density-based* methods, where the number of clusters is identified on demand [Tork

2012]. Thus, for the model proposed by this work, *density-based* methods are more suitable, since we do not know previously how many stay points might be created.

When the stay points are identified, the next step is to identify the type of places. The task of automatically discovering the type of a place may be facilitated by the use of APIs services which return a POI given a certain location, such as *Google Places*[1] and *Foursquare*[2]. However, this task is not trivial as it seems, since a user might be at a restaurant for *leisure*, and another might be at the same restaurant for *working*. Thus, this discernment is one of the challenging that needs to be addressed. Therefore, gathering further information, such as day of the week and duration that a user spent in a place, can help understand the relationship between users and locations.

At the moment that a vehicle starts to move, predicting the destination and route is useful in several contexts. For instance, by having this information, along with real-time traffic data, a computational system could suggest the user to take a detour, because the route commonly used is jammed. Furthermore, it is also possible to suggest POIs, such as a bakery or a market located along the route to the user's destination. A remarkable feature of predicting is that both points of interest and less jammed routes could be suggested without an active user participation in the process, which could improve the daily use of this kind of system. Thus, by just starting the trip, the system should be capable of predicting the destination and the path.

There are two important observations related to user displacements that we empirically have identified:

- **People's daily driving follows a pattern**. Workday activities often include trips to work, to home, or to a leisure activity (e.g., beach, restaurant). Even in vacation times, people use to repeat certain trips, such as visits to some Shopping Center. Furthermore, for a significant number of daily trips, it can be observed repetitions of the paths traveled. For example, people tend to always take the same route to go from home to work. Thus, if the place of departure and the destination of a user are known, it is possible to estimate the path the user is likely to take.
- **Trips occurs at similar times**: Besides the repetition of trips (i.e., origin, destination and route), it can be observed a pattern of times and the days of the week in which the trips occur. Hence, it is reasonable to assume that certain contextual information, such as day of the week and time, could be useful variables to improve the destination prediction.

Given a set of GPS points, our model identifies the stay points, infer the type of place that a user is located, partition all the trips which users travelled, associate each GPS point to a road segment, which is called *map matching* technique [Quddus and Noland 2006], and predicts the destination and the remaining path. For route and destination prediction, we propose Prediction by Partial Matching (PPM) technique as the core of our model, which was originally conceived for the data compression context. Summarizing, the main contributions of the model proposed by this work are as follow:

[1] https://developers.google.com/places/

[2] https://developer.foursquare.com/

- Identify stay points and type of places automatically, with support of APIs services, such as *Google Places* and *Foursquare*;
- Enrich trajectories semantically, by the use of contextual information, improving the task of understand the behavior of users' displacement;
- Predict real-time route and destination as soon as user starts a trip, apart from the type of place prediction.

The experiment carried out in this work was focused on individuals who use the vehicle for personal transportations only, instead of those who use it as work, as is the case of taxi drivers. The route database was created from real displacements, captured by using an application installed into *smartphones* of the participants of this work. From the GPS points collected, information such as day of the week and departure time related to the points was also obtained, for helping to improve the model.

The rest of this paper is organized as follows. Section 2 addresses related works. Section 3 presents our developed approach. The collected data and experimental results are discussed in section 4. Finally, the last section concludes the paper and discusses future work.

2. Related Work

There are many works that can be found in the literature concerning the problem of short-term and long-term prediction of destination and routes, and several different techniques have been proposed. Simmons et al. (2006) used the Hidden Markov Model (HMM) and contextual information (day of the week, time and speed of the vehicle) in a corpus of 46 trips in the Michigan area, in the United States. The rate of correct predictions was of 98%. Nevertheless, only 5% of the transitions from one segment to another occurred in intersections between streets, while the other 95% were connected to only one other road segment, which reduces the difficulty in the prediction of the next segment. For the 5% of transitions occurred in corners, the rate of correct predictions was between 70% and 80%. In Krumm's (2008) work, the focus of his model is in predicting short-term, i.e., only next segments, instead destination prediction. His model uses Markov model for prediction, and after observing the last 10 segments traveled by a user, it is possible to predict the next one with 90% accuracy. For predicting the next 10 segments the accuracy rate decrease to 50%. In contrast with Krumm's work, our model predict both route and destination, instead of only the next road segments.

Froehlich and Krumm (2008) use a closest match algorithm, that identifies the similarity between an ongoing route and a route performed in the past, and, if they are similar, the remaining path and destination are predicted. They do not use *map matching* technique, which considerably increase the volume of data that they work. Tiwiri et al. (2012) use a similar methodology for predicting routes and destination as proposed by Froehlich and Krumm (2008). However, Tiwiri et al. (2012) perform map matching, and showed a reduction in the size of data worked, apart from a progress in the performance of the predictive algorithm. The works of Froehlich and Krumm (2008) and Tiwiri et al. (2012) have reached about 40% of accuracy rate in prediction. The PPM algorithm has already presented encouraging results in the work of Burbey and Martin (2008), which is also concerned with the prediction of future location. The training approach considers the time the users arrive at places, the amount of time they stay at those places, and their current location. The results present 92% accuracy. A main difference between Burbey and Martin (2008) work's and ours is that we consider route prediction, and uses automatic semantic identification of places.

Knowledge discovery techniques, such as association rules, have already been used as an approach to the prediction problem. When a vehicle starts to move, an association rule is obtained for the moving object (according to the streets it passes by). Then a pattern matching function searches for the set of segments of the path traveled in a paths tree. In Morzy (2006), a version of the *Apriori* algorithm is used to generate the association rules. Tanaka et al. (2009) present a hybrid method of predicting destination. Their hybrid method is capable of changing the approach to predicting the destination according to the type of road.

In location-aware systems, semantic information is the action of linking contextual data about geographical places with raw position data collected [Parent et al. 2013]. Thus, a cluster where many geographic points are located can be useful for identifying pattern of displacements, but limited for identifying the reason why the person stays in such place. Thus, semantic information can enrich a trajectory with information such as name and type of place. Ying et al. (2011) are among the pioneers in considering semantic data for improving place prediction. The data that they collected are from both GPS and cell tower signals. For creating semantic tags, they populate the geographic semantic information database (GSID), which contains semantic information from *Google Maps*[3]. Their system comprises two modules: one offline, which is responsible for tagging the semantic locations; and another online, which is responsible for a real time location prediction. A limitation of this procedure relates to updating of the information. Ying et al. (2014) improved their previous work with item recommendations, i.e., when the system identifies that a person should stay in some place, it can suggests some items that are sold at that establishment.

Lung et al. (2014) developed a model for predicting destinations and for detecting the transportation mode. They use *Google Maps API* to search for a location, and enrich the trajectory. Their prediction model, which is based on Hidden Markov Model, was tested with real world data, and an accuracy rate of 68.3% was obtained for identifying the next location. Cao et al. (2010) proposed a model that first identifies the *stay points.* When the object remains stationary for a long period of time at the same place, a *stay point* can be identified. Then, they try to tag that place retrieving the name and type of place from the *Yellow Pages*. They do not perform location prediction, but they create a ranking for the most visited locations.

Our work differs from works that only use geographical information because we also consider semantic information for enriching the trajectories. We are not only interested in identifying the patterns of movements, but also in understanding the reason why the user is at a certain place. The difference between our work and the work of Ying et al. (2014) and Lung et al. (2014) is that we predict not only destination, but also the route user will pass.

Table 1 demonstrates the works most related to ours, and summarizes them by the following features: if the type of place is automatically identified; whether both route and

[3] https://www.google.com.br/maps

destination (or place) are predicted (or one of them); the method applied for route and destination prediction; the accuracy rate. Each line represents one work analyzed.

Table 1: Summary of works most related to ours

Authors	Identify type of place auto?	Route and Destination Predition?	Method for Prediction	Accuracy Rate
Simmons et al. (2006)	No	Both	Hidden Markov Model	95% / 70-80%
Krumm (2008)	No	Segment	Markov Model	90%
Burbey and Martin (2008)	No	Place / Destination	PPM	92%
Tiwari et al. (2012)	No	Both	Closest Match Algorithm	40%
Mazhelis (2011)	No	Both	Longest Common Subsequence	87%
Ying et al. (2011)	Yes	Place / Destination	Partial Matching and Longest Common Sequence	53% - 68%
Monreale et al. (2009)	No	Place / Destination	Prefix Tree Pattern Matching	~54%
Froehlich and Krumm (2008)	No	Place / Destination	Closest Match Algorithm	40%
Lung et al. (2014)	Yes	Place / Destination	Hidden Markov Model	68.3%

It can be noticed that a few works draw attention to join semantic information with geographic location. Most of the papers that we encountered in the literature only consider geographical information for predicting route and destination. The exploration of geographic semantic information can be an important feature to improve the prediction.

3. The PredRoute Prediction Model

This section describes our predictive model. First, we formally introduce important concepts used along this paper: *route*, *partial route*, *remaining route, stay point, contextual information* and *trajectory model*. These definitions are stated below.

- A *route R* comprises a sequence of segments $(S_1,S_2,S_3,\ldots,S_n,\ n > 0)$, i.e., $R = (S_1,S_2,S_3,\ldots,S_n)$, with $n > 0$ and S_i representing the i^{th} road segment of a route;
- Each *road segment*, or just *segment*, has exactly two geographic points $(P_{i1},P_{i2},P_{i3},\ldots,P_{ik},\ k > 1$ and $1 \leq i \leq n)$, i.e., $S_i = (P_{i1},P_{i2},P_{i3},\ldots P_{ik})$, with $k > 1$, and P_{ik} representing the k^{th} point on the i^{th} road segment. A point (x, y) represents a geographic coordinate (latitude, longitude);
- A *partial route T* represents a subset of segments of a route R $(S_1,S_2,S_3,\ldots,S_m,\ 1 \leq m < n)$, i.e., $T = (S_1,S_2,S_3,\ldots,S_m)$, with $1 \leq m < n;$
- A *remaining route F* $(S_{m+1},S_{m+2},\ldots S_{m+p},S_n,\ m + p + 1 \leq n)$ represents the predicted subset of segments to a certain destination, i.e., $F = (S_{m+1},S_{m+2},\ldots,S_{m+p}S_n)$, with $m + p + 1 \leq n$. Figure 1 depicts the concepts of *route, partial route, remaining route* and *road segments*;

- We consider many variables as *contextual information*, among them: *day of the week* of the departure, which is represented by an integer (0 = Sunday, 1 = Monday, ..., 6 = Saturday); the *time interval* of departure which is represented by an integer that corresponds to an interval i between two times (0 for $0 < i \leq 1$; 1 for $1 < i \leq 2$; ...; 23 for $23 < i \leq 24$); *origin* and *destination*, which represents, respectively, the place of origin and the place of destination of a route; *type of place*, which represents the type of location that a user remains. The possible values for the variable *type of place* in our work are home, work, other, sports, education, leisure and unknown;
- A *stay point, cluster* or *stop*, is a geographic area which represents a place that a user spent a time interval greater than a *threshold D*. The value for D considered in our work is 10 minutes. For finding out the time interval that a user spent in a cluster, it is necessary that the GPS points are ordered by timestamp, and that the distance between consecutive points are less than X meters. The value for X considered in our work is 40 meters. Both values for D (10 minutes) and X (40 meters) were empirically defined;
- A *trajectory model* comprises a list of road segments and contextual information.

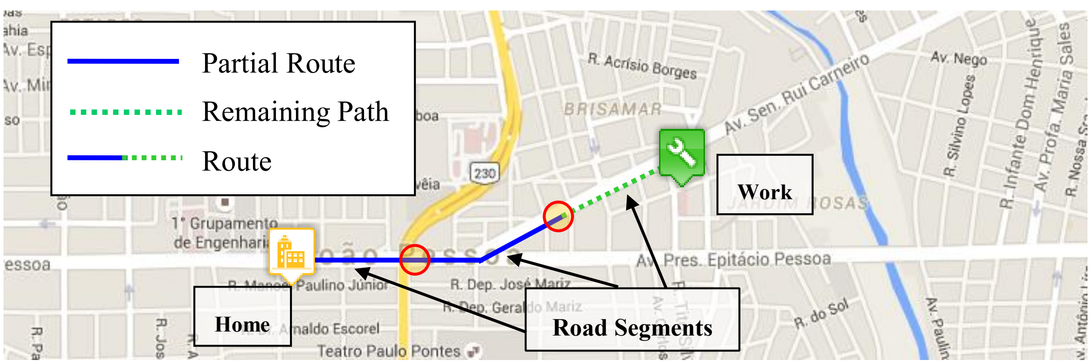

Figure 1: Definition of route (or trip), partial route, remaining path and road segments

3.1. Prediction by Partial Matching

The Prediction by Partial Matching (PPM) algorithm is a sophisticated method for data compression based on statistical models, and is among the most efficient techniques concerned with compression without loss of information [Salomon 2004]. The key idea of this method is the use of an adaptive symbolic model in a finite context. That is, a probability is assigned to a symbol not based on its frequency in the information source, but on its frequency in the context formed by the last *n* characters. For each order of, there is a table of symbols, which is updated for each new symbol codified.

PPM has some features which can be useful in classification and prediction tasks, since it has the capability of rapidly elaborating a symbols tree, adapted to the information source. The symbols tree is called a PPM symbols tree, or simply PPM tree. Further details about the behavior of PPM, including a step by step of an example and the creation of the PPM tree, can be found in Nobre Neto et al. (2015). Because of the features and behavior of PPM, we use it as the core of our model for predicting route and destination.

3.2. Identifying Stay Points

An important step of our predictive model is the process of identifying stay points automatically, which is based on *clustering* techniques. An stay point comprises a centroid point (latitude, longitude) and a radius of 40 meters, and it is created when the object remains stationary inside this area more than 10 minutes. The algorithm of identifying stay points proposed by this work is based on DBSCAN [Ester et al. 1996], a *density-based* algorithm for clustering spatial points [Tork 2012]. Algorithm 1 details the procedure for creating the stay points. The algorithm takes as input a list of users (line 2). For each user (line 6), the algorithm retrieves the set of GPS points ordered by timestamp, which represents the trajectories performed by that specific user (line 7). From those data, the clusters are extracted (line 8). For creating of stay points from GPS points, it is necessary that a user remains stationary for a minimum of 10 minutes, and the distance between the points may not be superior 40 meters. When the stay points are identified, they are associated with the current user (line 9). Then, based on the stay points recently created and on the set of GPS points, the algorithm calculates the routes performed by the user (line 10). Afterwards, the map matching procedure is performed, which associate a geographic point (latitude, longitude) with road segments (line 11). The advantage of doing map matching is that the data to be handled by our model is reduced [Tiwiri et al. 2012]. The output of the algorithm is the same list of users, however containing information about their stay points and the routes performed (in terms of road segments).

Algorithm 1: Procedure for spatial clustering creation

```
1   INPUT
2     users      // List of users for creating spatial clustering points
3   OUTPUT
4     users      // List of users updated, with their respectively list of clusters
5   METHOD
6     FOR EACH users as anUser DO
7        gpsPoints = anUser.getGpsPointsOrderedByTimestamp();
8        clusters   =  extractClustersFromGpsPoints(gpsPoints);
9        anUser.clusters = clusters;
10       anUser.trips = extractTripsFromClustersAndGpsPoints(clusters, gpsPoints);
11       anUser.tipsRoad = mapMatchPointsRoad(anUser.trips);
12  // End of FOR EACH
```

It is important to notice that our methodology for identifying stay points does not involve any procedure for identifying the type of place. Up to this moment, we just identify the length of time a user remains stationary in a stay point and the time the user reached the destination. Thus, we are dealing only with geographical data.

3.3. Type of Places Identification

Our approach to automatically identifying type of places of the stay points is detailed in Algorithm 2. This algorithm takes as input a list of users with their respective stay points, as showed in the procedure of Algorithm 1 (line 2). For each stay point of each user (lines 6 and 7), the algorithm retrieves contextual information (the day of the week, the time interval and the length of time remained stationary in the stay point) (line

8). Then, external services API (*Google Places*, *Foursquare* and *Factual*[4]) are online queried for reverse geocoding the stay point (centroid point), gathering information about the POIs around it (lines 9-12). The information collected of the POIs include the name, type of place, the distance between the stay point and the POI. After that, the algorithm identifies the nearest POI among the three retrieved to the stay point (line 13). Then, the type of POI is retrieved, and mapped to the types of location that our model considers (line 14). For instance, if the POI chosen was from *Foursquare* service, and his type is *Restaurant*, then our inference engine might identifies whether the type of place of the stay point is for *Leisure* or for *Work*. The inference engine considers the contextual information retrieved related to the stay point that the person remains stationary to discover the type of place (line 14). Our inference procedure works as follows:

- **Home**, if a user spends more than 10 hours at a 90% of the days;
- **Work**, if a user spends between six and eight hours at a location, and there are some days of the week that the user does not go to that place;
- **Leisure**, if a user goes to a place that he/she does not go frequently, and spends between two and four hours;
- **Sports**, if the type of the POI retrieved is related with sports (such as “gym”, “soccer”, “football”), and user spends between one and two hours;
- **Education**, if the type of the POI retrieved is related with education (such as “library”, “university”, “high school”) , and user spends between two and four hours certain days of the week;
- **Other**, when the user is supposed to be sorting things out and spends between ten and sixty minutes at a place;
- **Unknown,** if none of the types of place above has occurred.

Algorithm 2: Procedure for automatically type of places identification

```
1   INPUT
2     users        // List of users with their respectively clusters
3   OUTPUT
4     users        // List of users updated, with their the clusters enriched with semantic
5   METHOD
6     FOR EACH users as anUser DO
7         FOR EACH anUser.stayPoint as stayPoint DO
8           Info = getContextualInformation(stayPoint);
9           centroidPoint = getClusterLocation(stayPoint);
10          googleInfo = getGooglePlaceInfo(centroidPoint);
11          foursquareInfo = getFoursquareInfo(centroidPoint);
12          factualInfo = getFactualInfo(centroidPoint);
13          serviceChosen = getNearestPOI(googleInfo, foursquareInfo, factualInfo);
14          stayPoint.placeType = inferType(serviceChosen, info, centroidPoint);
15  // End of both FOR EACH
```

3.4. Route and Destination Prediction

This sections is divided into two, which describes the details about the training and testing stage.

[4] http://factual.com/

3.4.1. Training Stage

The training stage consists of creating our predictive model for route and destination for each participant of the experiment. Therefore, the predictive model of a given user is personalized, that is, it will not be influenced by the trajectories performed by another user.

The procedure for training our predictive model is presented in Algorithm 3. The algorithm takes as input a list of users, which contains information about displacements, stay points visited by the users and user identification (line 2). The output of the algorithm is a list of users with their respectively trajectory models created (line 4). Regarding the execution of the algorithm, for each map matched route (at this moment a route is a list of road segments) from each user (lines 6 and 7), the exact location and road segments of origin and destination are gathered (line 8). Then, contextual information is retrieved from the route, which are the day of the week, the time interval of departure and the type of location of the origin and destination of stay points (line 9). Such route information is then used to create the PPM tree (line 10). The next step (line 11) consists of creating a trajectory model from all of these information captured between lines 8 and 10. If this trajectory model already exists (i.e., the model has already stored this trajectory), then a counter is incremented (lines 12 and 13). This can occur in case of a user has several equal displacements, such as home to work. Otherwise, the trajectory model is stored for the first time (lines 14 and 15).

Algorithm 3: Procedure for training stage

```
1   INPUT
2     users      // List of users for creating spatial clustering points
3   OUTPUT
4     users      // List of users updated, with their respectively trajectory models
5   METHOD
6     FOR EACH users as anUser DO
7       FOR EACH anUser.tripRoad as route DO
8         POIs   =  getOriginAndDestinationLocation(route);
9         contextualInfo = getContextualInformation(route);
10        ppm-tree = routeToPPMTree(route);
11        traject-model = createModel(POIs, contextualInfo, ppm-tree);
12        IF (anUser.existTrajectory(traject-model)) THEN
13            anUser.incrementCount(traject-model);
14        ELSE
15            anUser.store(traject-model);
16  // End of both FOR EACH
```

3.4.2. Testing Stage

The testing stage consists in obtaining the rates of correct predictions of the users destination and route, from the moment their trip starts. A test in the context of our work is to predict the geographic destination and route of a user ongoing displacement, and to predict the type of place that a user is going. The routes used in the training stage are not used in the testing stage. Therefore, we apply cross-validation in our tests, partitioning the corpus of routes for training to the corpus of routes for testing.

Algorithm 4 details the procedure for executing tests. The algorithm takes as input the object user, the list of GPS points along with timestamp of an ongoing route and contextual information, which are day of the week, type of stay point of the origin and origin, (lines 2-4). First, the algorithm retrieves trajectory models that have similar contextual information with the ongoing route, such as the day of the week, the time interval of departure, the stay point of departure and the type of the stay point of the origin (line 9). Then, the algorithm performs a map matching with the list of GPS points of trip (line 10). The route performed so far is compressed with all PPM trees of the retrieved trajectories model (line 12), in order to obtain the trajectory model with the best compression ratio (lines 13-15). The compression generates a Compression Rate (CR), which is the division of the clean file with the codified file. Nobre Neto et al. (2015) provides further details about this compression rate. The output of this algorithm is the best selected trajectory model for the ongoing trip, which contains information about the remaining path (road segments), the destination and the stay point of destination (line 6). Thus, with this information, we provide for the final user the stay point and the type of the stay point that he or she is going, besides the route that will be performed.

Algorithm 4: Procedure for testing stage

```
1    INPUT
2      user        // User that is an ongoing route
3      trip      // List of GPS points along with timestamp info of an ongoing trip
4      contextualInfo // Contextual information: day of week, type of place origin, origin
5    OUTPUT
6      selected-trajectory-model   // A trajectory-model predicted
7    METHOD
8      max-compression-rate = -1
9      trajectories-model = user.getTrajectoriesModel(trip, contextualInfo);
10     routeMapMatched = mapMatchPointsRoad(trip);
11     FOR EACH trajectories-model as aModel DO
12         curr-comp-rate = compress(aModel, routeMapMatched);
13         IF (cur-comp-rate > max-compression-rate) THEN
14              max-compression-rate = cur-compression-rate
15              selected-trajectory-model = aModel
16   // End of FOR EACH
```

4. Experimental Evaluation

This sections explains the data selected for the testing stage, and presents the results obtained from our model.

4.1. Data Selection

The data used in this work were obtained from people living in the cities of João Pessoa and Campina Grande, both in the State of Paraíba (Brazil). We selected eight participants for installing into their *smartphones* an application for capturing their position. The application can use both wireless network and GPS device of the *smartphone*. If a user is located in an indoor place, which possess Wi-Fi, then this type of resource is used for gathering the location. In an outdoor location, the 3G (if enabled) or GPS device of *smartphone* was used. The participants were oriented to let the application executing, since it can send data to the server automatically. More than 300.000 GPS points were collected from the *smartphones* of the participants, which represents a total of 156 routes. Thus, an average of 19.5 routes per user were generated. The data were collected for users that possess completely different habits, during one month.

4.2. Results

As mentioned in section 3.4.2, cross-validation (90% of data for training and 10% for testing) was performed in this work. From the route to be tested, our model derives three new ones, the first with 15% of the route, the second with 50% and the third with 85%. This is important for discovering if the prediction accuracy increases or remains the same when the route is getting near from destination.

Table 1 summarizes the results obtained. There are two results considered in this work: one about route and destination prediction (RDP), which considers only geographic movements; and the other that is type of place prediction (TPP), which considers semantic information. For each result, there are three columns, representing the progress of the route to be tested. With 15% of the route performed, the accuracy rate for RDP was 39.2%, while TPP have 60.7% of correct rate. Testing 50% of the route, the accuracy rate of RDP increases to 45.96%, while TPP reached 62.9%. When the route has 85% of the segments travelled, RDP has an accuracy rate of 46.02%, and TPP reaches 62.9%.

Table 1: Accuracy rate according to the percentage of an ongoing partial route

	Route and Destination Prediction (RDP)			Type of Place Prediction (TPP)		
	15%	50%	85%	15%	50%	85%
Accuracy Rate	39.2%	45.96%	46.02%	60.7%	62.9%	62.9%

Our tests were performed on a computer equipped with a Core i7-4500 CPU, 16GB of RAM and 1TB of Hard Disk, and about one second have been spent for predicting route, destination and the type of place.

5. Conclusions and Future Work

The model proposed by this work is for predicting both destination and routes, apart from the type of location. In the tests performed, where our algorithm uses cross-validation, it was possible to obtain that the model has a better accuracy rate for predicting the type of place of the destination compared to the route and destination prediction, which considers only geographic displacements. Thus, even that the algorithm predicted wrong geographic destinations, it was possible that the type of place predicted might be correct. Differently from many works, we incorporate semantic information in our predictive model. The daily use of our model might be really useful, because it is not necessary an active user interaction and a good performance was obtained of the execution.

For further work, we intend to predict if a person is getting away from a destination that we initially predicted, that is, instead of predicting a new destination based on historical displacement, we will try to discover if the user is going to a place that he had never visited before. This will be possible because our model is considering semantic information. Another planned improvement is to expand the type of places that we consider, and develop an *Application* for implementing the model proposed by us.

References

Aggarwal, C. C. and Reddy, C. K. (2013), Data Clustering: Algorithms and Applications. Chapman and Hall/CRC. Boca Raton, FL.

Alvares, L. O., Bogorny, V., Palma, A., Kuijpers, B., Moelanes, B., and Macedo, J. A. F. (2007) “Towards Semantic Trajectory Knowledge Discovery”, In: Technical Report, Hassel University.

Burbey, I. and Martin, T. L. (2008) “Predicting Future Locations Using Prediction-by-Partial-Match”, In: Proc. 1st ACM MELT, pages 1-6.

Cao, X., Cong, G., and Jensen, C. S. (2010) “Mining Significant Locations From GPS Data”, In: Proceeding of VLDB, Vol. 3, No. 1, pages 1009-1020.

Ester, M., Kriegel, H., Jorg, Xu, C. (1996). “A density-based algorithm for discovering clusters in large spatial databases with noise”, In: Proc. 2nd Int. Conf. on KDD.

Froehlich, J. And Krumm, J. (2008) “Route Prediction From Trip Observations”, In: Society of Automotive Engineers (SAE).

Krumm, J. (2008) “A Markov Model for Driver Turn Prediction”, In: SAE.

Lung H-Y., Chung C-H., and Dai, B-R. (2014) “Predicting Locations of Mobile Users Based on Behavior Semantic Mining”, In: Trends and Applications in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science, Vol. 8643.

Morzy, M. (2006) “Prediction of moving object location based on frequent trajectories”, In: ISCIS, p. 583-592, Springer.

Nobre Neto, F. D., Baptista, C. S., and Campelo, C. E. C. (2015) “Predicting Routes and Destinations of Urban Trips Using PPM Method”, In: 7th SBCUP, Brazil.

Quddus, M. A., and Noland R. B. (2006) “A High Accuracy Fuzzy Logic Based Map Matching Algorithm for Road Transportation”, In: Journal of Intelligent Transportation Systems.

Salomon, D. (2004), Data Compression: The Complete Reference. Springer, 3rd Edition, New York, NY.

Simmons, R., Browing, B., Yilu, Z. and Sadekar, V. (2006) “Learning to Predict Driver Route and Destination Intent”, In: Intelligent Transportation Systems Conference.

Tanaka, K., Kihino, Y., Terada, T., and Nishio, S. (2009) “A Destination Prediction Method Using Driving Contexts and Trajectory for Car Navigation Systems”, In: ACM symposium on Applied Computing, pages 190-195.

Tiwari, V. S., Chaturvedi, S., and Arya, A. (2013) “Route Prediction Using Trip Observations and Map Matching”, In: IEEE IACC.

Tork, H. F. (2012) “Spatio-temporal Clustering Methods Classification”, In: DSIE – Doctoral Symposium in Informatics Engineering.

Parent, C., Spaccatietra, S., Renso, C., Andrienko, G., Andrienko, N., Bogorny, V., Damiana, M. L., Gkoulalas-Divanis, A., Macedo, J., Peledis, N., Theodoridis, Y. and Yan, Z. (2013) “Semantic Trajectories Modeling and Analysis”, In: ACM Computing Surveys, Vol. 45, No. 4, Article 42.

Ying, J. J-C., Lee, W-C., Weng, T-C. and Tseng V. S. (2011) “Semantic Trajectory Mining for Location Prediction”, In: Proceedings of the 19th ACM SIGSPATIAL GIS.

Ying, J. J-C., Chen H-C., Lin, K. W., Lu, E. H-C., Tseng V. S., Tsai, H-W., Cheng, K. H. and Lin, S-C. (2014) “Semantic trajectory-based high utility item recommendation system”, In: Expert Syst. Appl. 41(10): 4762-4776.