Predicting Routes and Destinations of Urban Trips using PPM Method

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

[1]Departamento de Sistemas e Computação – Universidade Federal de Campina Grande (UFCG) – Caixa Postal 58.429-900 – Campina Grande – PB – Brazil

[2]Instituto Federal de Educação, Ciência e Tecnologia da Paraiba (IFPB) – Campina Grande – Brazil.

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

Abstract. *Information about destination and route that a person will take is important for various purposes, such as to prevent a user going through a congested route. However, an information system where users must explicitly input their intended destination seems not be useful for daily routines. Ideally, the system should be able to predict the destination and the route to be taken by a vehicle as soon as it starts to move. This paper presents a new technique to predict route and destination, based on Prediction by Partial Matching (PPM) compression method. By considering two important contextual information (day of week and time of departure), the results obtained by our approach were encouraging, reaching around 92% of accuracy rate.*

1. Introduction

Predicting the destination and route of a vehicle as soon as it starts to move is very useful in several contexts. Having this information, along with real-time traffic data, a computational system could suggest, for example, less jammed routes, which are faster and safer. Furthermore, it is also possible to suggest points of interest, such as a bakery or a drugstore located along the route for 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.

An important observation related to user displacement is that 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, one observes repetitions of the paths traveled. For example, people tend to take always 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. Besides the repetition of trips (i.e., origin, destination and route), it is observed a standard about the schedule and the days of week in which the trips occur. In this scenario, it is possible to infer that contextual information, such day of week and time, can be useful variables to improve the destination prediction.

An important decision for dealing with prediction of routes and destination is related to the scenario to be chosen. For example, is the prediction to be made in short

or long term? What kind of routes are to be used (eg. highways, urban roads)? Different approaches have been proposed to various contexts, since predictions in an expressway might require different procedures compared to predictions in a rural road.

Route, *partial route* and *remaining path* are terms frequently used in this paper. Therefore, formal definitions for these concepts are given in Section 3. At this moment, a high level definition of them is provided. We consider a route as a sequence of road segments. Road segment, or just segment, is an abstraction of a physical road segment, and is composed by two or more geographical points. A partial route is an ongoing route, i.e., when an object is moving and did not reach the destination yet. Finally, a remaining path is the route predicted while the object is moving.

Our work focus on the real-time prediction of routes and destination. Given a partial route, the systems predicts the destination and the remaining path. We propose a method based on the Prediction by Partial Matching (PPM) technique, which was originally conceived for the data compression context. 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 in most part real, but it contains simulated data too. In order to improve the real-time prediction, some information was obtained from the users, such as day of week and departure time.

The rest of this paper is organized as follows. Section 2 presents related works. Section 3 presents our developed approach, and in Section 3.2 we detail the working of the PPM. The collected data and experimental results are shown in section 4. Last section, we conclude the paper and discusses future work.

2. Related Work

Several works can be found in the literature concerning the problem of short-term prediction of destination and routes. These works use many different techniques: association rules, Morzy (2006); neural networks, Mark et al. (2004); Bayesian networks, Fei et al. (2011); and Markov models, Krumm (2008) and Simmons et al. (2006). Some works also considers contextual information such as time and day of week, that we take into account in this work to improve the prediction of routes and destination based on the departure location.

Simmons et al. (2006) used the Hidden Markov Model (HMM) and contextual information (day of week, time and speed of the vehicle) in a corpus of 46 trips in the Michigan area, in the United States. They applied cross validation, and used 90% of the trips for training, while the remaining ones were used for testing. 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% of the transitions did not occur in the crossings, 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 of 70%. The work by Simmons et al. differs from our work due to the creation of segments: we create segments that are delimited by the intersection between streets, which makes the classification task more difficult.

In Krumm (2008), in turn, segments must have approximately 237 meters in length. Therefore, segments are not necessarily delimited by road intersections. Differently, in our work the segment length is variable. Krum's work uses Markov

model as method to prediction. In his approach, after observing the last 10 segments traveled by a user, it is possible to predict the next one with 90% accuracy. For the prediction of the 10 next segments, it was achieved an accuracy rate of 50%.

In Chen et al. (2011), the prediction of the path and destination is done according to the individual's movements, instead of being done based on the path traveled by the vehicle. In that work, in the training stage, a route is created with each node representing a region of interest. As input for the testing, it is passed a set of routes already traveled. The method obtained a rate of 71% of the trips predicted correctly. In our work, we consider the prediction of movements of vehicles, instead of persons.

Xue et al. (2013) use Markov chains and Bayesian inference to model the route prediction algorithm. Their approach supports the prediction of trips which did not occur yet, that is, the predictor is not just modeled based on the historical data. With the Bayesian inference, the future path to be taken is inferred based on a path history, in which the probability for choosing a certain route after the prediction point is obtained according to the amount of the path that has already been traversed. The corpus used by Xue et al. (2013) comprises the displacements of taxi drivers, rather than displacements of people who use their cars for personal transportation, which is the focus of our work.

Knowledge discovery techniques, such as association rules, have already been used as an approach for 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. One of the approaches takes into account contextual information, such as day of week, time of day, number of passengers, weather conditions and baggage weight.

The PPM algorithm has already presented encouraging results in a work of Burbey and Martin (2008), which is also concerned with prediction of future location. Their 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. Despite of the similarity to our work regarding the technique used, our method consists in the prediction of destination and route. In addition, we consider different input information.

3. Predicting Route and Destination using PPM

This section describes our approach for destination and route prediction. First, we formally introduce important concepts used along this paper: *route*, *partial route* and *remaining route*. A route R comprises a sequence of segments $(S_1,S_2,S_3,\ldots,S_n,\ n > 0)$. Each segment has two or more points $(P_{i1},P_{i2},P_{i3},\ldots,P_{ik},\ k > 1$ and $1 \leq i \leq n)$, where the first and the last one are located at the intersecting between streets. Each point P represents a geographic coordinate (latitude and longitude). A partial route T represents a subset of segments of a route R $(S_1,S_2,S_3,\ldots,S_m,\ 1 \leq m < n)$. The segments $(S_{m+1},S_{m+2},\ldots S_{m+p},S_n,\ m + p + 1 \leq n$ and $0 \leq p < n\text{-}m)$ represent the remaining route (or path) F, to a certain destination. Formally, we have:

- $R = (S_1,S_2,S_3,\ldots,S_n)$, with $n > 0$ and S_i representing the i^{th} road segment of a route;

- $S_i = (P_{i1},P_{i2},P_{i3},...P_{ik})$, with $k > 1$, and P_{ik} representing the k^{th} point i^{th} road segment. A point (x, y) represents a geographic coordinate (latitude, longitude);
- $T = (S_1,S_2,S_3,...,S_m)$, with $1 \leq m < n$;
- $F = (S_{m+1},S_{m+2},...,S_{m+p}S_n)$, with $m + p + 1 \leq n$, and $0 \leq p < n\text{-}m$.

In the next subsections we explain our prediction technique, based on data compression PPM method, and our training and testing stage.

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 update for each new symbol codified.

When a character is read from the source to be codified, it is sought in the higher-order context, that is, the one which groups the higher number of characters. If it is found, it will be coded with the probability of the symbol in the context identified, and, after the codification, the probability will be updated. In the case it is not found, a special character, called *escape*, is coded and then the algorithm tries to code the symbol in the lower-order context. This procedure of reduction of context repeats until the character of the source is coded. The zero-sized context references the context of smallest order, which contains all the symbols obtained from the information source. The symbol probability is computed through the frequency at which it appears in a context (the last *n* characters) divided by the sum of the probabilities of other symbols in that context, including the frequency of the escape. The frequency of the escape in a context is just the number of different symbols in that context. For example, if in the context "ab" the symbols "a" and "b" have already been codified, the escape frequency in this context is two. The variant of PPM used in this work was PPMC, which the count of *escape* is equal to the different symbols appeared in the context.

Table 1 shows a symbols tree at the end of the codification of the word "abraabra", for a second order context (K = 2). Each column of the table represents an order. The first column is order 2 (K = 2), where the coding of a symbol of the information source must the done considering the last two characters read. For example, for the codification of the symbol "r" and if the last two symbols (context) coded were "ab", the "r" would be coded with probability (p) of 66% (p = ⅔), since the symbol "r" appeared twice before the present coding (c = 2). However, if instead a "b" were to be codified also in the "ab" context, it would be codified the escape with probability 33% (p = ⅓), and switched to the smallest order (K = 1). Now, there would be an attempt to codify the symbol "b" in the order K = 1, in which only the last symbol would be retrieved, the "b" context. Again, we notice the absence of the symbol "b" in the "b" context, and the escape will be codified. In the order K = 0, we notice that the symbol "b" was retrieved twice from the source (c = 2) and its probability to be codified will be 18% (p = 2/11). After that, the contexts "ab" and "b" must be updated, because they now have a new symbol, "b". Also, the values of the escape will be updated to 2 in both contexts, since there will be two different symbols.

Table 1: PPM tree modeled for the word "abraabra"

Order $k = 2$			Order $k = 1$			Order $k = 0$			Order $k = -1$		
Predictions	c	p	Predictions	c	p	Predictions	c	p	Predictions	c	p
ab → r	2	$\frac{2}{3}$	a → a	1	$\frac{1}{5}$	→ a	4	$\frac{4}{11}$	→ A	1	$\frac{1}{\lvert A\rvert}$
→*Esc*	1	$\frac{1}{3}$	→ b	2	$\frac{2}{5}$	→ b	2	$\frac{2}{11}$			
br → a	2	$\frac{2}{3}$	→ *Esc*	2	$\frac{2}{5}$	→ r	2	$\frac{2}{11}$			
→ *Esc*	1	$\frac{1}{3}$	b → r	2	$\frac{2}{3}$	→ *Esc*	3	$\frac{3}{11}$			
ra → a	1	$\frac{1}{2}$	→ *Esc*	1	$\frac{1}{3}$						
→ *Esc*	1	$\frac{1}{2}$	r → a	1	$\frac{1}{2}$						
aa → b	1	$\frac{1}{2}$	→ *Esc*	1	$\frac{1}{2}$						
→ *Esc*	1	$\frac{1}{2}$									

A feature of PPM, which can be useful in classification and prediction tasks, is the capability of rapidly elaborating a symbols tree, adapted to the information source. This symbols tree, shown in Table 1, is called PPM symbols tree, or simply PPM tree.

3.2. Training Stage

The training stage consists in the individual modeling of the routes and destinations of the users of the experiment. For each user, there is a 7x24 matrix of trajectory models, where the rows represent the seven days of the week, and the columns represent the 24 1-hour interval throughout a day. That is, the first column comprises all the records with time between 0 and 1 a.m., the second one comprises the records between 1 a.m. and 2 a.m., and so on. Each cell of the matrix stores trajectory models that represent the different displacements of a user. A trajectory model is represented by a 6-tuple <*route, origin, destination, day-week, time, route-type*>, where:

- *route* is the trajectory performed between origin and destination points;
- *origin* is the place where the displacement begins;
- *destination* is the reached place, when the displacement finish;
- *day-week* is the day of week that the route was performed;
- *time* is the interval time that the route was performed;
- *route-type* is a label for the route according to the type of origin and destination (e.g., home-to-work, home-to-leisure). The possible values of types for origin and destination are: home, work, leisure, education and sports.

For a user who starts a trip on Mondays from home to work, between 8 a.m. and 09 a.m., always following the same route, there will be one single trajectory model in the cell of the matrix corresponding to this day and time. Nevertheless, if on a given Monday between 8 and 9 the user goes to the same destination (work), but through a different route, another trajectory model will be added to the cell corresponding to the respective day of the week and time interval.

For the 6-tuple <*route, origin, destination, day-week, time, route-type*>, the computational representation for each one of the values is as follows: the origin and the destination are represented by geographic coordinates (latitude and longitude); the day of week is represented by an integer (0 = Sunday, 1 = Monday, ..., 6 = Saturday); the time range is also represented by an integer which corresponds to an interval *i* between two times (0 for $0 < i \leq 1$; 1 for $1 < i \leq 2$; ...; 23 for $23 < i \leq 24$). The route is computationally represented by a tree, in which each node (intermediate or leaf) corresponds to a segment. Figure 1 (a) illustrates the status of a trajectory models matrix. In the first cell (Sunday, between 00:00 and 1:00), there are two models (represented by two filled circles). This means that there are two different 6-tuples <route, origin, destination, day-week, time, route-type>. An empty cell represent the absence of trips in the corresponding day of week and time interval. Figure 1 (b) represents a trajectory model, with a PPM tree representing the path traveled and the textual description below it, containing information about origin and destination of the route, day of week and departure time, besides the route type. A node in the tree represents a segment (except the root node). This route is assigned to a particular user.

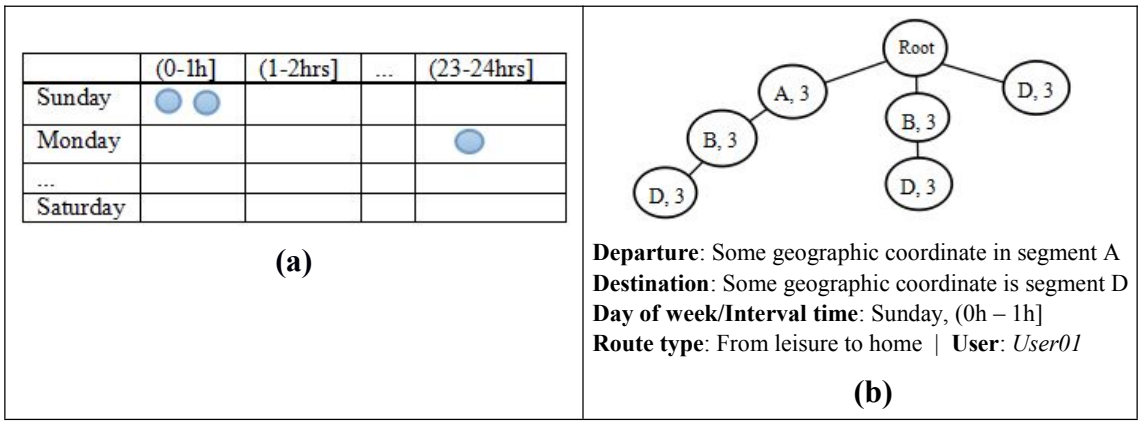

Figure 1: (a) Trajectory models matrix; (b) Content of a trajectory model

The computational modeling of the routes and destinations for each user (construction of the trajectory models matrix) is based on historical information about trips traveled by the user (origins, destinations, route types, day of week and departure time). In the Algorithm 1, it is possible to visualize the pseudo-code of the procedure used to generate the trajectory models matrix. The algorithm receives as input the information about origin and destination (geographic coordinates and type), route information (segments of the path, day of week and departure time) and a 7x24 trajectory models matrix. Each cell of the matrix has a list of trajectory models. In the body of the algorithm (starting at line 8), it obtains the day of week and time interval of the beginning of the trip (lines 9 and 10) based on the route information. These data will be used later to identify the correct cell to which the trajectory model will be added. In line 11, a word is created based on the segments that form a trajectory. In line 12, the word created is compressed in order to generate the PPM tree. Then, in line 13, a trajectory model is created based on the information obtained (including the PPM tree for the route traveled). This model is added to the models matrix (lines 14).

Algorithm 1: Parameters, output and procedure for the update of the models matrix

```
1    INPUT
2      origin-info       // Coordinate and type of origin
3      dest-info         // Coordinate and type of destination
4      route-info        // segments, day of week and time interval
5      matrix-of-models  // Matrix of PPM models - specific user
6    OUTPUT
7      matrix-of-models  // Updated Matrix of PPM Models
8    METHOD
9      day-of-week  = route-info.day
10     time-interval  = timeToTimeInterval(route-info.time)
11     word       = segmentsToChars(route-info.segments)
12     ppm-tree     = createPPMTree(word)
13     trajectory-model = (origin-info, dest-info, day-of-week, time-interval, ppm-tree)
14     updateMatrix(matrix-of-models, day-of-week, time-interval,  trajectory-model)
```

3.3. 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. The routes used in the training stage serve as basis for the generation of the corpus in the testing stage. Each type of route gives origin to three new paths, after their partitioning in three percentages: 15%, 50% and 85%. Thus, for the composition of the test corpus, we discard the route used in the training stage, but take into consideration the three new routes resulting from the partitioning of the original route (partial routes).

For tests, which will predict the destination and the remaining path, the following information must be taken as input: user id, day of the week, departure time, and the path already traveled. The user id is used to retrieve the appropriate models matrix, while the day of week and the departure time are used to retrieve the trajectory models concerning that specific moment of the movement (i.e., the PPM matrix cell). Concerning the route traveled up to a given moment, it will be used to compute the probability values of the possible destinations to be reached and trajectories to be traveled from the user's present location. For example, if the partial route to be tested started on a Monday, at 8:30 a.m., only the trajectory models related to Mondays and in the time interval (8-9] will be retrieved. The several segments that form the partial route are converted into a word which, in turn, is compressed with basis on the PPM trees of the trajectory models retrieved. After that, a compression ratio is obtained (which is the quotient between the plain text and the compressed word) for the tree of each model. The PPM tree with the highest compression ratio will have its trajectory model assigned to that partial route. Thus, from the selected model, information about destination, remaining path and route type are retrieved.

Algorithm 2 details the procedure for executing tests. Input information consists of a partial route (percentage of segments traveled - 15%, 50% or 85%), the day of week and the departure time, besides the user to whom the prediction is to be made. The output is a remaining trajectory object, which groups the information about destination and the remaining route. In this procedure, the first step is to retrieve the models matrix of the user (line 9) and the information concerning the partial route (lines 10-11), both passed as input. The partial route data are used to retrieve the suitable trajectory models

(line 12). Then, the segments that form the partial route are converted into a word (line 13), which is then compressed with all the retrieved models (line 15), in order to obtain the trajectory model with the best compression ratio (lines 17-18). The remaining trajectory object is retrieved (line 20). This object contains data about the predicted destination, the remaining path and the destination route type. Cross validation was not carried out, since we have predicted routes only for previously traveled trajectories.

Algorithm 2: Procedure for destination and remaining path prediction of a partial route

```
1   INPUT
2     partial-route-info  // route info (segs, day and time int.)
3     user // A specific user
4   OUTPUT
5     T // Remainder Trajectory
6   METHOD
7     max-compression-rate = -1
8     selected-model = NULL
9     matrix-model = getMatrixModelFromUser(user)
10    day-of-week = partial-route-info.day
11    time-int = timeToTimeInterval(partial-route-info.time)
12    trajectory-models = getTrajectoryModel(M, day-of-week, time-int)
13    p-route-to-word = segsToChars(partial-route-info.segs)
14    FOR EACH trajectory-models as model
15       cur-comp-rate = Compress(p-route-to-word, model)
16       IF (cur-comp-rate > max-compression-rate) THEN
17            max-compression-rate = cur-compression-rate
18            selected-model = model
19     // End of for each
20    T = ObtainRemainderTrajFromModel(selected-model)
```

4. Experimental Evaluation

4.1. Data Selection

The data used in this work was obtained from people living in the city of João Pessoa, State of Paraíba (Brazil). We selected six participants who were asked to give details about their daily routes during the last week and label the origin and destination types. Based on this information, we defined different routes types, such as home-to-work. Also, the participants were instructed to inform the day of week and the time interval that they started their trips. The trajectory mapping was done with help of Google Maps[1].

We obtained a total of 119 real routes from the six participants. Among these routes, we noticed repeated ones, such as the same path from home to work. Grouping the repetitions according to the route type, we obtained 44 different types of routes. However, since the sample under analysis refers to a single week, the obtained data do not cover the case where there are multiple routes in the same day and time interval, which may hinder the prediction of destination and routes. For this reason, eight new route types were artificially added to the routes corpus (based on the real routes informed by the users), in order to simulate this test case. The routes created are as follows: four of them achieved the same destination of four original routes, but with no

[1] https://maps.google.com/

segment intersection; two of them covered 60% of segments equivalent to an original route; and the two remaining ones took 95% of equivalent segments into consideration, and only 5% of the remaining path was traveled through by other segments. The addition of the 8 new routes led to a total of 52 different route types.

4.2. Results

As presented in section 3.3, from each real route, we derived three new ones, the first with 15% of the original route, the second with 50% and the third with 85%. So, since there are 52 different types of routes, we tested a total of 156 routes.

Table 2: Accuracy rate according to the percentage of an ongoing partial route

	Percentual of progress of a partial route		
	15% performed	**50% performed**	**85% performed**
Accuracy Rate (%)	84%	84%	92%

Table 2 summarizes the accuracy rate of the tests performed by our predictive modeling. Considering only the 52 routes tested with 15% of the segments traveled, the algorithm managed to predict correctly the remaining path of 44 trajectories. The eight failures occurred because eight routes had more than 15% of equality in the segments, so it was not possible to distinguish the path to be predicted. Thus, the accuracy rate was of approximately 84%. This same rate was found with respect to the 52 routes with 50% of the trajectory traveled, since eight routes had more than 50% of equality in their segments, and it was not possible to tell which remaining path should be traveled. For the test with the 52 routes with 85% of the segments, 48 of them had the remaining route correctly predicted, which represents an accuracy rate of approximately 92%. The four failures were due to the existence of four routes with 95% of the trajectory similar, and just the destination being different. Considering 156 routes tested, 136 of them were correctly predicted, which represents an accuracy rate of approximately 87%. Figure 2 shows a screenshot of the prototype developed for the simulation. The continuous line represents the segments of the streets already traveled, whereas the dotted line represents the predicted segments. The car icon indicates the current position of the vehicle. The tool icon informs the user's destination type is the work. As the user moves, the prediction can be updated. Our tests were performed on a computer with a Core i7-4500 processor, where each route took approximately one second to be predicted.

Figure 2: Real-time simulation of a trip, with prediction of destination and remaining route

5. Conclusion and Future Work

The PPM algorithm proved to be efficient, quickly adapting itself to the information source. We observed that, with data concerning just one week, it was possible to build a considerably accurate model (though a larger volume of data would have made the

study even more reliable). In approximately 84% of the cases, with just 15% of a (partial) route, it was possible to obtain the destination and the trajectory to be traveled. Moreover, with 85% of a route, 92% of the remaining paths were correctly obtained. The main difference between our model and others in the literature is related to our predictive modeling. Our model can infer the destination just after a few movements, then it is updated in real time, while the route is performed.

We developed a mobile application to automatically collect the users' locations. This app, which is being used by several volunteers, will provide more reliable data for future experiments. Using data concerning a longer period of time, even more representative matrices can be built, with more robust and confident models. As an additional improvement, we intend to consider other relevant environmental variables, such as speed, current weather conditions, as well as information about the user's profile.

A limitation of this research is that the prediction of routes requires the use of at least one of the segments (among those already traveled) for training. Thus, in the case a user takes a completely different route, the system will not make any prediction. As future work, we intend to overcome this limitation by considering historical information from other users with similar profiles.

References

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

Chen, L., Lv, M., Ye, Q., Chen, G. and Woodward, J. (2011) "A personal route prediction system based on trajectory data mining", In: Information Science, pages 1264-1284, Elsevier.

Fei, X., Lu, C. and Liu, K. (2011) "A bayesian dynamic linear model approach for real-time short-term freeway travel time prediction", In: Transport. Res. Part C, pages 1306-1318, Elsevier.

Krumm, J. (2008) "A Markov Model for Driver Turn Prediction", In: Society of Automotive Engineers.

Mark, C. D., Sadek, A. W. and Rizzo, D. (2004) "Predicting Experienced Travel Time with Neural Networks: A PARAMICS Simulation Study", In: Proceeding of 7th International IEEE Conference on Intelligent Transportation Systems.

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

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.

Xue, A. Y., Zhang, R., Zheng, Y., Xie, X., Huang, J. and Xu, Z. (2013) "Destination Prediction by Sub-Trajectory Synthesis and Privacy Protection Against Such Prediction", In: International Conference on Data Engineering, pages 254-265.