Appl. Math. Inf. Sci. 6 No. 2S pp. 651S-659S (2012)

Applied Mathematics & Information Sciences
An International Journal

Probabilistic Graphical Model based Personal Route Prediction in Mobile Environment

Je-Min Kim[1], Haejung Baek[1] and Young-Tack Park[1]
[1] School of Computing, Soongsil University, 1-1, Sangdo-dong, Dongjak-Gu, Seoul, 156-743, Korea
Corresponding author: Email:kimjemins@hotmail.com, baekhj@gmail.com, park@ssu.ac.kr

Received July 15, 2011; Revised Aug. 15, 2011; Accepted Sep. 2, 2011
Published online: 1 January 2012

Abstract: Individuals tend to follow their own preferred paths when traveling to specific places. Information on these routes could be utilized to build various intelligent LBSs . In order to predict a current user's route, various approaches have been researched. In this paper, we suggest a practical approach to learning users' route patterns from their histories and using that information to predict specific routes. In cases where existing routes overlap, i.e., where parts of routes are the same, in a user's route model, it is difficult to identify the user's intended path. For more accurate prediction, firstly, we extract route patterns by adopting image processing. Secondly, we build a state-observation model reflecting users' intentions, based on route patterns, temporal features and weather information. Our approach consist of four steps: recognizing regions for splitting routes into trip segments, route pattern mining, learning users' route models and trip route prediction. Our method achieved a prediction accuracy of 96.4% in tests performed with 15 smartphone users.

Keywords: Data Mining, Temporal Probabilistic Model, Route Prediction, Location based Service.

1 Introduction

Smartphones incorporate many diverse and powerful sensors. In particular, the global positioning system (GPS) can easily observe and collect the trajectories of a person. When people travel in the real world, they leave record of their location histories in the form of GPS logs. The GPS logs provide a useful basis to learn the route patterns of people. Most people have personal route patterns, which are comprised of journeys of a repetitive nature. Personal routes would be extremely useful in the domain of location-aware intelligence services, particularly in applications such as group social networking based on the future location of users, prediction of personal behaviors, prediction of individuals' arrival times at specific places and so on.

These services have motivated researchers to explore the possibilities of route prediction. Currently, there are many different approaches to route prediction. A personal route prediction system proposed in [1] predicts the route that people will take, adapting a basic Markov model and 2nd Markov model. However, this system only use a GPS coordinates sequence to learn route patterns. In the same manner, most existing work does not incorporate all available facts that could provide clues to identifying a route..

In this paper, we propose a practical approach to predicting the current transit route of a user, using a probabilistic graphical model built from historical data. Our approach consists of four sequential parts: deciding upon RSPs (Route Separation Points), route pattern mining, personal route modeling and route prediction. In the RSP-decision step, we find significant places that separate a route into segments. This is done using a heuristic algorithm, adapting four filters to consider the velocity and density of a GPS sequence, WiFi access points, and user activity data. In the route pattern mining, using an image-processing algorithm, we abstract users' route patterns from personal GPS histories and detected RSPs. In the personal route modeling step,

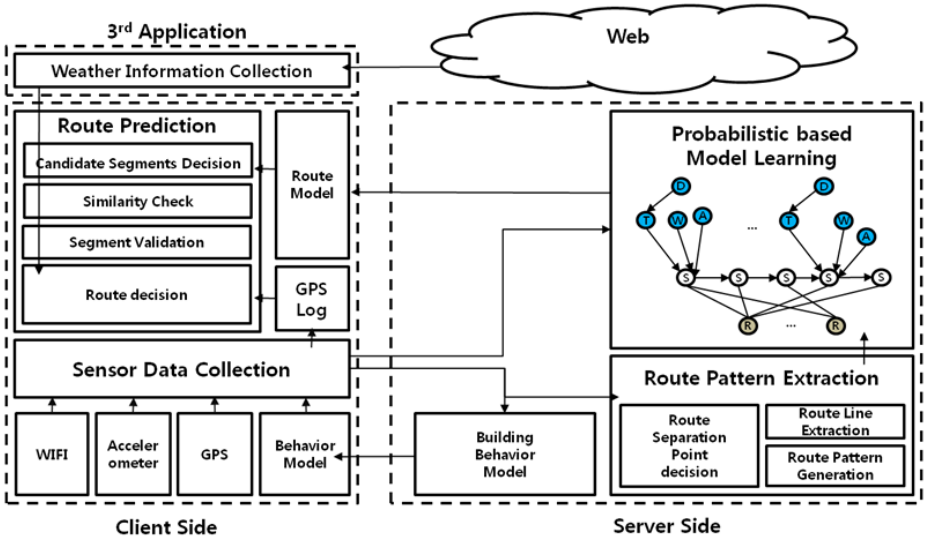

Figure 1: The personal route prediction architecture

we build a probabilistic model that is used in route prediction. In cases where segments overlap (i.e., some parts of routes are the same in a user's route patterns), it is difficult to identify the user's intended route. In general, when a person visits a specific place, routing is influenced by the user's environment. Therefore, we consider using not only coordinate sequences from users' GPS histories, but also aspects of their environment histories, including temporal information (day of the week, travel time), users' activities and weather information. In order to build a model, we consider the relationship among variables. There is a transition relation between values of segments because each route consists of an ordered segment sequence. On the other hand, there are conditional dependencies between segments and other environmental facts. In this paper, we adapt the State-Observation model [2]. In the personal route prediction step, we predict a user's current transit route, based on their route patterns and probabilistic models. This step includes four components: the candidate segment decision process, which identifies target segments to compare with a user's current location, the similarity calculation between candidate segments and a user location, the segment validation process, which decides whether the most similar segment is a user's current moving segment, and the route decision, based on current environmental facts.

In this paper, we make an important assumption . Many studies of users' activity recognition have been conducted [3-5]. Using diverse sensors offered by mobile devices, these studies infer users' activities, such as taking a bus, riding the subway, walking, running or standing. Therefore, the results of this prior work are incorporated into our approach to build a probabilistic model of users' route patterns.

2 Personal Route Modeling and Prediction

Our route prediction approach consists of four major steps: the RSP decision, which finds significant places that separate a route into segments; the route pattern mining step, which extracts the complete route patterns of user using an image processing algorithm [6,7]; the personal route modeling step, which entails learning the transition probability relationships among segments, as well as the dependencies between segments and environmental facts, using a State-Observation model; the personal route prediction step, which chooses the most suitable route based on a probability model.

To reduce the computational load on the smartphone, our approach has a client-server structure. Fig. 1 presents the client-server architecture for personal route prediction. The client side (smart phone) gathers recorded data in order to build a behavior model and route model. Then, the client sends the collected data to the server. The server learns personal behaviors, finds RSPs using four filters, extracts route patterns by employing an image-processing algorithm and builds a personal route model, based on a probabilistic analysis. The learned behaviors and personal route models are transmitted to the client. Thus, the client infers a

user's behavior and predicts their current transit route.

The route pattern extraction and model building process consumes a large amount of computing time and resources, and, further, it should be executed only rarely, thus these steps are performed on the server. On the other hand, the route prediction process should be executed in real-time, thus this step is performed on the client (smartphone).

3 Route Separation Points Recognition and Route Pattern Mining

In order to predict users' current transit routes, first, we perform the route recognition process to separate a personal route into segments. Adapting four filters, each region is separated by passing it through these filters. In this step, route separation points (RSPs) are decided upon by comparing a single parameter, associated with each filter, to velocity and density values, which are based on detected GPS coordinates, detected WiFi access points and a user's behavior during an arbitrary period of time, $t+\Delta$.

A RSP is categorized into a fixed area and a separation area. A fixed area is a region that a user remains in for a long time, such as a home or office. A separation area is a space within which a person's behavioral changes occur, such as a bus station or subway.

When a person reaches a separation area, there are differences in their velocity and activity. This occurs as they transition between route segments that are divided by this area because they may board public transit. When a person stays in a fixed area, this area shows a high density of GPS point records and, further, the same WiFi access points are detected repeatedly. Therefore, separation areas could be identified by the change in velocity and behavior, while fixed areas could be identified by the density of GPS points and detected WiFi access points in the area.

Velocity Filter: It is assumed that individuals move at a constant speed between two consecutive GPS points, and that there is a reasonable speed range for individuals. The parameter values of the velocity filter include Δvel_{non}, Δvel_{walk}, and $\Delta vel_{transit}$.

Density Filter: This is designed to check for the presence of redundant position data, which is recorded when users are inside specific areas, such as buildings. Given a window size, d, we first calculate the centroid for each d-sequential-position sequence in a region. Then, the maximum distance between these centroids is calculated to estimate whether the region contributes to a reasonable movement distance. The parameter values of the density filter include Δden_{non}, Δden_{sep}, and Δden_{fix}.

Behavior Filter: This filter assumes that individuals exhibit a single type of behavior at a given point in time. The parameter values of the behavior filter include Δact_{stay}, Δact_{walk}, Δact_{bus}, Δact_{car} and Δact_{subway}.

WiFi Filter: The value of the WiFi filter reflects the comparability of WiFi access points detected in a specific region. Therefore, this filter includes the $\Delta wifi_{mat}$ parameter value. As a result of testing, we concluded that the best precision for fixed area recognition is observed when we assign $\Delta wifi_{mat}$ a value of 0.85.

The route pattern extraction process is divided into two parts in which spatial information and temporal information are separately recognized. The first part entails representing locations on a grid space and learning only the geospatial information of GPS logs using image-processing. The second part entails making path graphs based on the learned spatial models and, further, learning route patterns using temporal factors of the GPS data.

3.1. Extracting Route Line

Grid mapping and line generation:

Consider Even though a user follows the same routes, the produced logs rarely show the same values because of GPS's inaccuracy (about 50m). In addition, GPS points in a real scale are sporadically distributed, which makes it more difficult to extract any pattern and also highly increases the complexity of doing so. To solve these problems and obtain GPS error tolerances, we generalize GPS measurements using a regular grid. The transformation function for GPS measurements from a single 2D Grid is defined as $T : H^2 \rightarrow N(R^2)$, where H^2 is the real world space using Haversine distance and R^2 is the whole 2D Grid using Euclidean distance.

We represent a space as a 400*400 two-dimensional, one-level grid, where each cell is less than 50 meters on a side. Deciding upon the dimensions of a grid space is a critical issue for the accuracy of trajectory representation. The basic idea is to cluster trajectories with similar start and end points, and to remove trajectory regions that are not clustered; those regions that are seldom visited by the user. We represent trajectories using simple lines, which connect start points with end points.

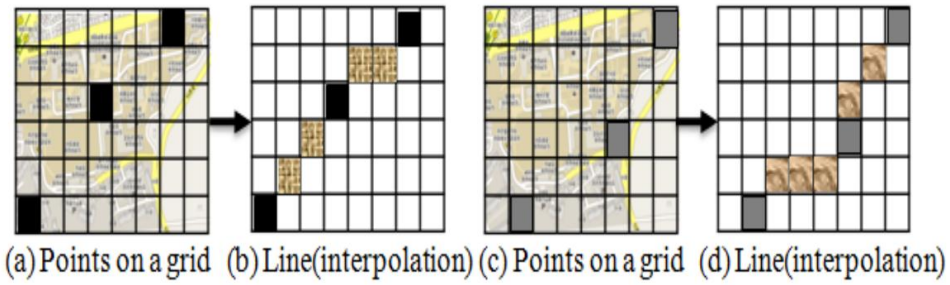

Figure 2: The process of line generation

Then, each trajectory is simply characterized by an angle, a start point, an end point, and the midpoint. If the angle and the three points (start, end and mid) between trajectories are similar, we consider the trajectories to be in similar regions. The function is designed to evaluate the closeness of four features when comparing two trajectories.

$$dist(I_i, I_j) = d_\theta(I_i, I_j) * w + dh_{center}(I_i.s_{im}, I_j.s_{jm'}) + dh_{start}(I_i.s_{i1}, I_j.s_{j1}) + dh_{end}(I_i.s_{ie}, I_j.s_{je'}) \quad (1)$$

Here, angular distance, d_θ, is the gap between the bearing angle of the starting and ending points of I_i and the bearing angle of I_j, while dh is the Harvesine distance. Even though the GPS points project onto a much smaller grid space, the grid points are not connected, which can be seen in panels (a) and (c) of Fig. 2. We make a connection by constructing intermediate points between the two initial points, following an interpolation process. For the connection, we adapt the Brensenham line algorithm, which is an efficient and fast algorithm, to draw a line on the grid space, graphically. Panels (b) and (c) of Fig. 2 depict the results of generating lines between points based on the Brensenham method [8]. These lines are called GPS lines, in order to differentiate them from route lines, which are learned from each GPS line.

Line integration and line thinning:

To abstract the accumulated GPS lines, we have adapted the thinning approach in computer vision. We restate our problem as a skeletonization of routes from an image, which is generated by integrating the GPS lines of trajectories on a 2D grid. When we integrate the lines, we use a thickening technique, which adds additional pixels to the original line. This simple technique allows our system to overcome the inaccuracy of GPS logs, stemming from GPS errors or reading intervals. From the integrated image, we extract pixel-wide route lines using a thinning algorithm. A simple example is shown in Fig. 3.

The accumulated image could be considered the result of summing images of the trajectories. The pixels, (x_i, y_j), in an integrated image, are counted using both those trajectories that have a pixel (x_i, y_j) that is taken as a GPS line as well as those trajectories that have a pixel (x_i, y_j) that is the neighbor of a GPS line, even though the latter pixels are not actually on GPS lines. Panel (a) of Fig. 3 simply shows this mechanism, in which the pixels surrounding the target pixel are on GPS lines. Even though the target pixel is not on the lines, we consider these pixels as components of user paths, check their validity and count them.

We use the Zhang-Suen thinning algorithm [7] for the skeletonization, to extract route lines from an integrated image. The Zhang-Suen algorithm has the advantage of processing speed as it uses a parallel method. However, this algorithm has some weaknesses, such as the fact that it produces that skeletons contain artifacts, like necking, tail and line fuzz. Further, the skeletons are not one pixel in width. To solve these problems, Parker has introduced a hybrid thinning algorithm [6]. The

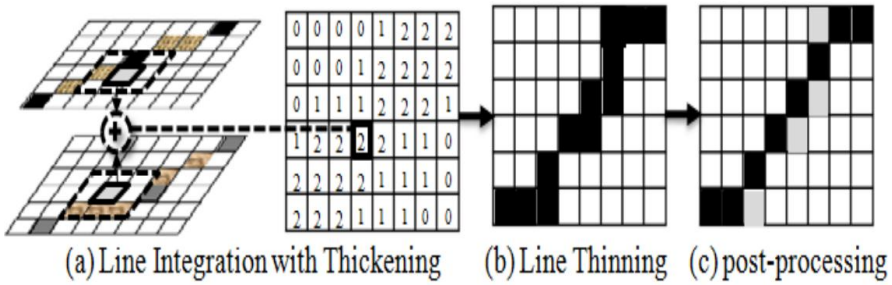

Figure 3: Pixel wide route line extraction example

algorithm merges three methods: Stentiford's preprocessing for reduction of defects, Zhang-Suen's thinning algorithm and Holt's staircase removal as a post-processing step, to produce a one-pixel-wide skeleton.

Panel (b) of Fig. 3 shows the result of thinning, which also reveals staircase problems. To extract one-pixel skeletons, we perform Holt's stair removal method. The one-pixel skeleton produced by Holt's algorithm is presented in panel (c) of Fig. 3. In route pattern learning, it is important to maintain topologies on the image, however, simplicity is more important. As such, we add the post processing algorithm to remove relatively short line fuzz as well as circulations that have a short distance, below a threshold.

3.2. Learning Route Pattern

We construct route patterns as graphs from the produced lines (skeletons) and RSPs. Firstly, structural features of skeletons become nodes, such as terminal points, turning points, which have less than two adjacent pixels or more, and RSPs. The connected pixels between nodes become edges. A line graph, LG, is a undirected graph that is a pair, (V, E), where $V = \{v_i: v_i \in P\}$ and $E \subseteq \{\{i, j\}: i, j \in V$ and $\{i,j\} \subseteq$ route lines$\}$, which is a set of unordered pairs. Fig. 4 shows a graph generated from a thinned image. We adapt a breadth-first search algorithm to make a line graph. The basic idea is to recognize nodes, choose a start node, *s*, explore every edge of the nodes, put the encountered nodes into a queue, Q, and to then repeat this routine until every node and edge has been checked.

Using the produced line graph, we learn the trajectories of a user with temporal information. We project each trajectory with a timestamp into a line graph, LG, using the similarities between the pixels of each trajectory and the pixels of the LG. This method of comparing pixels is relatively easy, but also produces a local minimum problem; a few pixels of a given trajectory may be similar. To avoid the local minimum problem, we perform a hybrid approach. In cases where the pixels of trajectories correspond to nodes of a LG that has at least three alternatives, we use a more sophisticated similarity function, which is explained in chapter 5. In the other case, where the pixels are not near nodes and have less than two alternatives, we just use the similarity between pixels.

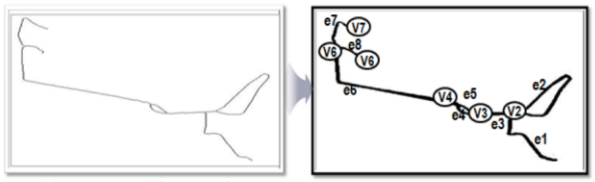

Figure 4: An example of a line graph

4 Building Personal Route Model

In cases where overlapping segments exist (i.e., where parts of routes are the same in a user's route patterns), it makes difficult to identify the user's intended route. In order to solve this problem, we adapt a probabilistic analysis to consider facts reflecting a user's intention. Considering two relations, we build the personal route model based on a State-Observation model.

First, we build a transition model for the transition probability between segments. Generally, the probability of a person travels in a segment is influenced by the previous segment because each route consists of an ordered segment sequence. Therefore, the transition model is defined by the following equation

$$P(s) = P(s \mid s_i) = \prod_{i=1}^{n} P(s_i \mid \Pr ev\, s_i) \quad (2)$$

Given a user's current segment, *s*, or a series of segments, (s_1, s_2 , s_3, ... , s_k), the prediction of the next segment that the user will visit, s_0, is determined by this joint probability. P(s, s_i) is the probability that s and si occur. This is the probability that a person visits s and s_i.

Second, we build an observation model for the conditional probability between a segment and set of environment facts. Generally, the probability that a person travels in a segment is influenced by the particular time, day of the week and weather. Furthermore, users' behaviors are different in different segments. Therefore, these facts could provide a basis for understanding a user's intended route. The observation model is defined by the following equation.

$$P(s, o_1, ..., o_n) = P(s) \prod_{i=1}^{n} P(o_i \mid s) \quad (3)$$

Given a user's current segment, *s*, and a series of observed values reflecting environmental facts, (o_1, o_2, o_3, o_4), the prediction of the next segment, s_0, is determined by the probability of each *o* and the conditional probability, P($s \mid o_i$), for each *o*. P($s \mid o_i$) is the probability that *s* and o_i occur. This refers to the chances of it being a particular time, particular

day, the weather being in a particular state and the users performing a particular behavior when they visit *s*. The following table shows the values of observed variables.

Table 1: The values of each environmental variable

Environment variable	*Value*
time of day	morning, noon, night
day of week	weekday, holiday
weather	sunny, rainy, snowy
behavior	stay, walk, inBus, inSubway, inCar

5 Personal Route Prediction

The route prediction process is composed of candidate segments selection from a set of target segments, a similarity calculation, which finds the best connected segment (BCS), a segment validation process, which decides whether the best similarity segment is a predictable segment, and a route decision based on current environmental facts. In route decision step, considering the user's intentions, we adapt a probabilistic method to resolve the overlapping routes problem. We use both a segment of GPS logs as well as the average travel time, traveling days, user activity and weather information, in order to determine the user's intention.

Because recording a segment of GPS logs for use in route prediction consumes significant smartphone battery life, we set a time interval of ten seconds for the recoding of GPS points. A segment of GPS logs, *G'*, is a sequence of locations, from g_1 to g_m.

Finding a current route within a route model consumes much computation time in the similarity calculation. Therefore, it is important to reduce the number of segments that are to be compared, which we call target segments. The candidate segment decision process finds target segments that exist within a bounded segment of GPS logs.

The boundary threshold, which we call the MB (Minimum Bound), is a bounding box that reduces the search space of the route model. Pivoting on each point in a segment of GPS logs, MBs are calculated by λ. Considering our context of location-aware mobile services, we determine λ using GPS error rates and the walking speed of each user. Therefore, λ is determined by the maximum deviation between each trajectory that is used in building a route model, a learned route pattern and the average distance per period (10 seconds).

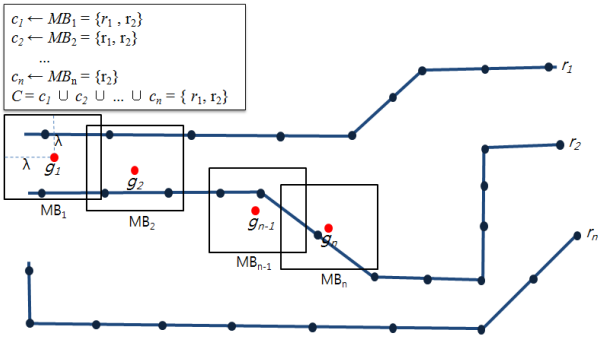

Figure 5: An example of candidate segments selection

$$\lambda = \sqrt{(\max\_deviation)^2 + (Ave\_dist/time)^2} \quad (4)$$

The similarity calculation process decides the BCS that has the highest similarity within a candidate set, for a segment of GPS logs. The similarity is calculated by a function that considers the distance from the points of each segment in the route model to each point in the segment of GPS logs. We define *dist*, the distance between a point of GPS log, g_i, and a segment, $s_m = \{s_1, s_2, \ldots, s_j\}$, as

$$dist(g_i, s_m) = \min_{s_j \in T} (dist(g_i, s_j)) \quad (5)$$

where $dist(g_i, s_j)$ is the Haversine formula based on the distance between g_i and s_j. So, in actuality, $dist(g_i, s_m)$ is the shortest distance from g_i to any point on s_m. We define the similarity function, $Sim(G', s_m)$, between G' and s_m, based on the distance of each matched pair. The matched pair, $<g_i, s_j>$, consists of g_i and the nearest point to g_i, s_j. We use the exponential function, *e*, to measure the contribution of each matched pair to $Sim(G', s_m)$. This is because we would like to assign a larger contribution to a closer matched pair of points, while giving a much lower value to those pairs that are distant. This results in an exponentially decreasing contribution, as $dist(g_i, s_m)$ linearly increases.

$$Sim(G', s_m) = \sum_{i=1}^{m} e^{-dist(g_i, s_m)} \quad (6)$$

The segment validation process decides whether the BCS is the segment that should be used in a prediction. To maintain efficiency, we adapt a lower bound (LB). The LB is a standard value that accumulates the allowable error distance for each s_j. In order to select a proper LB, we consider the maximum deviation and average deviation between each trajectory used in building a route model, a learned route pattern, as well as the margin of GPS error (20, 30, 50 meter). According to the results of a test, using the average deviation in the validation

process was found to produce the best accuracy for route prediction. If the similarity of the BCS is greater than the LB, the BCS is a predictable segment.

$$LB = \sum_{i=1}^{m} e^{r_{s_i}} \tag{7}$$

The route decision process infers a user's current transit route based on a predictable segment and a route probability model. As Fig. 6 shows, routes r_1 and r_2 have a common segment, which is the nearest path (from v_1 to v_2) to a segment of GPS logs. Thus, r_1 and r_2 have equal similarity to a segment of GPS logs. In order to resolve this problem, we adapt a State-Observation model to consider other facts that reflect a user's intention.

In order to decide a user's intended route, we use environmental information, such as the variables in table 1. For instance, our method calculates the probability of segment v_2~v_4 and the probability of segment v_2~v_3 for a given State-Observation model, in order to determine whether the predictable segment is r_1 or r_2. Therefore, a route including a segment that has the highest probability is determined to be the user's current transit route.

$$\begin{aligned} &current\,route \equiv (\arg\max p(seg_i)) \in r_i \\ &p(seg_i) \equiv \prod_{i=1}^{n} p(seg_i | \Pr ev\, seg_i) * p(o | \Pr ev\, seg_i) \\ &p(o | \Pr ev\, seg) \equiv p(\Pr eg\, seg) \prod_{i=1}^{n} p(o_i | \Pr eg\, seg) \end{aligned} \tag{8}$$

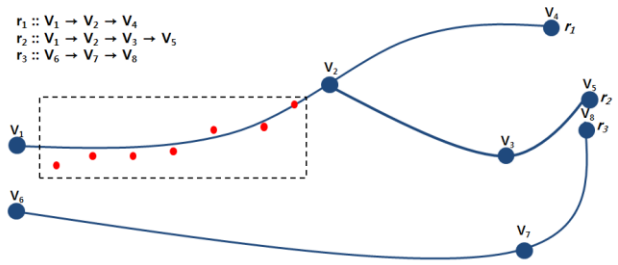

Figure 6: An example of an overlapping route problem

6 Experiment

We describe the results of tests for our approach. For the tests, we have collected real data sets from 15 users for roughly 60 days, in Korean cities. We performed three tests in order to prove the practicality of our approach for route prediction. Our approach is implemented in Java and examined on a android phone with QSD 8250 and 512MB Memory. In order to detect users' behavior, we use an activity recognition module, which was developed by Kyunghee Univ. This module could distinguish 5 behaviors (stay, walk, run, bus, subway) based on the smartphone's accelerometer and GPS. The test data consist of 46 routes of 15 users (tester), generated by route model learning.

In Korean cities, in order to get on a bus or a subway, people commonly walk (about 5~10minites) from specific places (home, office and so on) to bus stops or subway stations. Therefore, it is proper that a segment of GPS logs has a length of 1~5 minutes, to maintain the practicality of prediction results. Through the first test, we decided the appropriate length of a segment of GPS logs. Each tester identified whether a predicted route is a current route on which he/she is moving. This was done using our pilot application, which predicted the user's current route. This work is performed over one month without the trajectory validation step.

For the detailed analysis of test results, the fifteen testers were divided into five groups and each tester identified whether a predicted route was their current transit route. As shown in Fig. 7, the accuracies of route predictions, based on 3-minute segments of GPS logs, are 92~94%. On the other hand, the accuracies for 1 or 2-minute segments of GPS logs are low, overall. The accuracies for 4 or 5-minute segments of GPS logs are no better than those obtained with 3-minute segments of GPS logs.

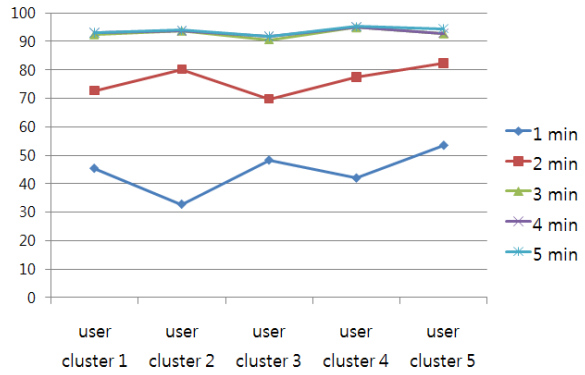

Figure 7: The accuracy by the length of a GPS log

In order to decide a threshold (for the LB) to validate whether a BCS is a predictable segment, we performed a second test (using 3 minute segments of GPS logs). We consider the maximum deviation and average deviation between the GPS points of a route and GPS points of each trajectory used in building the route, in addition to GPS error margins of 20, 30, and 50m (because detected GPS points have an error rate of about 50m, at most)

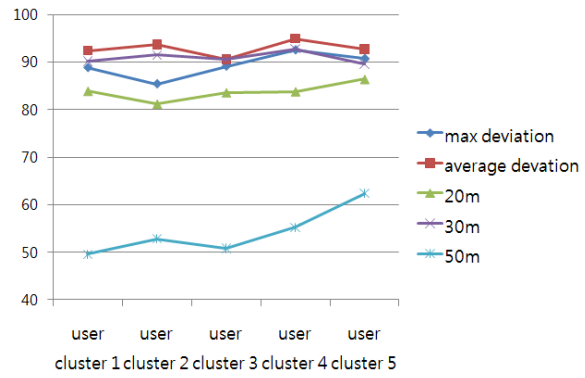

Figure 8: The validation accuracies for different threshold factors of the LB

As shown in Fig. 8, the accuracies of the validation results are the highest when the average deviation is applied to a threshold to calculate the LB. Because the overall validation accuracies are 92~95%, the average deviation is a good factor in determining the threshold for validation.

In the final test, we measured the accuracy of our route prediction approach, adapting a probabilistic approach with a segment of GPS logs (3 minutes in length - test 1) and the average deviation (test 2). We consider sensitivity and specificity in computing accuracy. Sensitivity refers to whether a predicted route correctly identifies the user's current intended route and specificity refers to whether non-current routes are correctly filtered. Table 2 shows the result for our final test.

Table 2: The test results of route prediction

	Predictable	*Unpredictable*	*Total*
Current transit route	536	14	550
No current transit route	22	428	450
Total	541	459	1000
Sensitivity	0.975		
Specificity	0.951		
Accuracy	0.964		

7 Conclusion

In this paper, we propose a practical approach for personal route modeling and prediction. For efficiency and performance modeling, we suggest a learning approach using image-processing and a probabilistic method. In order to perform efficient prediction, we focus on three parts - the appropriate length of a segment of GPS logs, the decision of users' intended route, adapting a probabilistic method, and the threshold for validation. In particular, the proposed probabilistic approach, which is based on a State-Observation model, helps to solve problems with overlapping routes that reflect a user's intentions. Then, we suggest the average deviation as a useful threshold for validation tests. Based on experiments conducted with 15 smartphone users, our approach shows 96.4% accuracy. In route pattern learning, we have found several points that can be improved, such as parameter -sensitivity or troubles with routes over short distances. In the future, we will focus on these improvements.

Acknowledgements

This work was supported by the Industrial Strategic Technology Development Program (10035348, Development of a Cognitive Planning and Learning Model for Mobile Platforms) funded by the Ministry of Knowledge Economy(MKE, Korea).

References

[1] L. Chen, M. Lv, Q. Ye, G. Chen, and J. Woodward. A personal route prediction system based on trajectory data mining. *Information Science*, Vol.181, No.7, (2011), 1264-1284.

[2] D. Koller and N. Friedman. Probabilistic Graphical Models: Principles and Techniques (The MIT press, Cambridge, USA, 2009).

[3] E.M. Tapia, S.S. Intille, W. Haskell, K. Larson, J. Wright, A. King, and R. Friedman. Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. *Proceedings of the International Symposium on Wearable Computers*, (2007), 1-4.

[4] J. Lester, T. Choudhury and G. Borriello. A practical approach to recognizing physical activity. *Proceedings of the International Conference on Pervasive Computing*, Vol.3968, (2006), 1-16.

[5] Y. Zheng, Y.K. Chen, Q. Li, X. Xie and W.Y. Ma. Understanding transportation modes based on GPS data for web applications, *ACM Trans on the Web*, Vol.4, No.1, (2010), 1-36.

[6] J.R. Parker. Algorithm for image processing and computer Vision (Wiley, Hoboken, USA, 2010).

[7] S. Zhang and K.S. Fu. A thinning algorithm for discrete binary image, *Computer Graphics and Image Processing*, Vol.13, No.2 , (1980), 142-157.

[8] J.E. Bresenham, Algorithm for computer control of a digital plotter, *IBM systems journal*, Vol.4, No.1, (1965), 25-30.

Je-Min Kim is a Ph.D. student at School of computing , Soongsil University. In 2004 he received his M.Sc. degree in computer science. He worked previously at the Kangnam University as a teacher in the fie ld of compute r science. He is the author and co-author of several scientific papers, and participates in semantic web, ubiquitous computing and mobile computing project. He has strong scientific and development expertise in ontology modeling, ontology reasoning and machine learning.

Haejung Baek is a researcher at School of computing, Soongsil University. She received her M.Sc. degree in computer science in 1998. In 2004, she received her Ph.D degree in Artificial Intelligence; with thesis focused on active learning of robots using symbolic and non-symbolic learning. She studied on robots at KIST(Korea Institute Science Technology) and on object recognition for robots at CMU(Carnegi Mellon University in USA). She is also a adjunct professor in Kijeon University in Jeonju. She is interested in machine learning, robot(vision and cognitive learning) and mobile computing.

Young-Tack Park is a professor at School of computing, Soongsil University. He received his M.Sc. degree in computer science in 1980. In 1992, he received his Ph.D. degree in Artificial Intelligence; with thesis focused on black board scheduler control knowledge for heuristic classification. He teaches A.I at the Faculty of Computer Science. He is also a supervisor and consultant for Ph.D. , master and bachelor studies. He has authored and coauthored multiple research papers and participated in national research projects. His research interests in Semantic Web, Ontology Reasoning and Machine Learning. He has strong scientific and development expertise in ontology reasoning, agent system and mobile computing.