Predicting the Next Location Change and Time of Change for Mobile Phone Users

Mert Ozer, İlkcans, Ismail Hakki Toroslu, Pınar Karagoz
Computer Engineering Department
Middle East Technical University
Ankara, Turkey
{mert.ozer, ilkcans, toroslu, karagoz}@ceng.metu.edu.tr

Salih Ergut
Avealabs, Avea Technology Center
Umraniye, Istanbul, Turkey
salih.ergut@avea.com.tr

ABSTRACT
Predicting the next location of people from their mobile phone logs has become an active research area. Due to two main reasons this problem is very challenging: the log data is very large and there are variety of granularity levels for specifying the spatial and the temporal attributes. In this work, we focus on predicting the next location change of the user and when this change occurs. Our method has two steps, namely clustering the spatial data into larger regions and grouping temporal data into time intervals to get higher granularity levels, and then, applying sequential pattern mining technique to extract frequent movement patterns to predict the change of the region of the user and its time frame. We have validated our results with real data obtained from one of the largest mobile phone operators in Turkey. Our results are very encouraging, and we have obtained very high accuracy results.

Categories and Subject Descriptors
H.2.8 [Database Applications]: Data Mining

General Terms
Algorithms

Keywords
Sequential Pattern Mining, Location Prediction, Mobile Phone Users

1. INTRODUCTION
In today’s world, mobile phones have almost become indispensable part of daily life and almost everyone has one in average. Mobile phone operator companies log and use the basic usage information of their users such as their base station connections, call, SMS and GPS records. The collected data is further used for modelling the user’s movement profiles. We have focused on the location prediction problem. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

SIGSPATIAL ’14, November 04-07 2014, Dallas/Fort Worth, TX, USA
Copyright 2014 ACM 978-1-4503-3142-5/14/11 ...$15.00
http://dx.doi.org/10.1145/2675316.2675318

rather than focusing on a modelling human mobility which is a much broader problem.

In our recent work [10], we aimed to predict the user’s next location for two cases. The first one is predicting the next action’s (call, sms, gprs connection) location and time and the second one is predicting the next location of the user when he/she changes his/her current location by using only spatial data.

Since determining the exact coordinate of the next location of a user is almost impossible, we used the coordinates of the base station that user is connected to as the representative of the location of that user. Moreover in densely populated city centers, these base stations are very close to each other and they are far from each other in rural areas. To overcome the imbalance of the distances between base station coordinates, we clustered them into regions and we have obtained regions with similar sizes.

By applying sequence pattern mining techniques we have obtained 84% accuracy for the second problem, i.e., predicting the region of the next action of the user. A quick analysis showed that next action’s location is his/her current location for the 80% of the users’ actions. Therefore, we did not elaborate on the first prediction problem since a simple baseline of giving just the current location as next location prediction has already 80% accuracy.

In this work, similar to the second problem of [10], we again focus on the prediction of the user’s next location and time when he/she changes his/her current location. Rather than just using spatial information as in [10], by taking also temporal information into consideration, we get more precise prediction results.

In order to realize this prediction process, we follow four phases, which are preprocessing the data, clustering base stations, extracting frequent patterns by using sequence pattern mining methods and predicting the change of location and its time for mobile phone user. For the experimental evaluation, Call Detail Record (CDR) data obtained from one of the largest mobile phone operators in Turkey has been used.

The rest of this paper is organized as follows. Section 2
introduces previous work on location prediction problem. Section 3 gives the details of problem definition and the data used in this work and then introduces the proposed solution for the problem. Section 4 contains the experimental results of our prediction method. Section 5 concludes our work and points out possible further studies.

2. PREVIOUS WORK

In recent years, a variety of location prediction schemes have been presented in [11], [5], [12], [6], [3], [4], [2], [13], [7].

In [11] and [5], location prediction methods that do not make use of any sequential pattern mining techniques are proposed. In [11], Thanh et al. instead use Gaussian distribution and expectation maximization algorithm to learn the model parameters. Then, mobility patterns, where each is characterized by a combination of common trajectory and a cell residence time model, are used for making predictions. In [5], Gao et al. propose a method which uses Bayes’ rules for prediction models which use historical data while predicting the next location. For the best model named as HPY Prior Hour-Day Model, they managed to predict user’s next locations with an accuracy of 50%.

Similar to our work, in [12], [6] and [3], they propose sequential pattern mining techniques for the location prediction problem. In [12], Yavas et al. propose an AprioriAll-based algorithm which is similar to our method. They extract frequent user trajectories which they name user mobility patterns (UMP) from a user movement database and predict the user’s next movement accordingly. However they do not use any spatial or temporal information while extracting UMPs or predicting. In [6], Giannotti et al. propose methods to solve different trajectory pattern mining problems. They define spatio-temporal sequences as the pairs of spatial attribute and the time that user has spent in there. They also try to detect the popular regions which is named as ROI. The difference with the conventional sequence pattern mining technique is the use of trajectories (T-patterns) rather than itemsets. In [3], Cao et al. introduce a method for discovering periodic patterns in spatio-temporal sequences. They also make use of an AprioriAll-based algorithm for extraction of periodic patterns. The distinctive feature of these periodic patterns is that they are not frequent in the whole time span but in some time interval, so they change their support definition accordingly.

In addition, in [8], we propose an Apriori-based method for location prediction which is named as Apriori-based Sequence Mining with Multiple Support Thresholds (ASMAMS). This method extract rules from data with respect to the multiple support parameters and predict accordingly.

3. PROPOSED METHOD

In this section we introduce our problem definition, data we used in this work and give the proposed method’s details for the given problem in the following subsections.

3.1 Problem Definition

The problem can be summarized as to predict the next location and to determine the time when the user changes his/her location. Predicting the time when change in location occurs is the basic novelty in the problem definition, which is not the case in our previous work [10]. Rather than predicting exact time of the action, it is aimed to find the time interval that action takes place in. Moreover, rather than predicting exact coordinate or base station, it is aimed to find the next region of the user. For this reason, base stations are grouped into the regions.

3.2 Data Set

In this work we utilized the CDR data of one of the largest mobile phone operators of Turkey. The data corresponds to an area of roughly 25000 square km with a population around 5 million. The CDR data contains roughly 1 million user’s log records for a period of 1 month. For each user there are 30 records per day on average. The whole area contains more than 13000 base stations. The records in CDR data contain anonymized phone numbers (of caller and callee or SMS sender and receiver), the base station id of the caller (sender), the time of the operation and some other additional information.

3.3 Preprocessing

Due to the high volume of the data and high number of irrelevant attributes such as city code, phone number etc., it is necessary to apply some basic preprocessing tasks on the data. First, we filter the unnecessary attributes. Date and time information are merged into a single column and, it is used for sorting records in temporal order. We further combine call data records of a user on the same day into a single record. By this way, each record, which is structured as a sequence of <base station id, time of the day> pairs, represents a user’s daily movement sequence. Time of the day attribute is formatted as ‘hh:mm’. An example preprocessing step can be seen in Table 1 and Table 2. For the sake of the table be fit into the document, we do not include unimportant attributes in Table 1.

<table>
<thead>
<tr>
<th>Reg#1</th>
<th>user#1</th>
<th>Reg#2</th>
<th>user#2</th>
<th>Day</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>R91</td>
<td>phone#1</td>
<td>R91</td>
<td>phone#2</td>
<td>20120907</td>
<td>010251</td>
</tr>
<tr>
<td>R91</td>
<td>phone#1</td>
<td>R21</td>
<td>phone#3</td>
<td>20120907</td>
<td>071008</td>
</tr>
<tr>
<td>R55</td>
<td>phone#1</td>
<td>R27</td>
<td>phone#4</td>
<td>20120907</td>
<td>072231</td>
</tr>
<tr>
<td>R55</td>
<td>phone#1</td>
<td>R27</td>
<td>phone#4</td>
<td>20120907</td>
<td>111540</td>
</tr>
<tr>
<td>R55</td>
<td>phone#1</td>
<td>R91</td>
<td>phone#5</td>
<td>20120907</td>
<td>144332</td>
</tr>
<tr>
<td>R55</td>
<td>phone#1</td>
<td>R3</td>
<td>phone#6</td>
<td>20120907</td>
<td>170304</td>
</tr>
</tbody>
</table>

Table 1: CDR data before preprocessing

| R91,01:02 | R55,07:22 |

Table 2: User sequence obtained from CDR data for phone#1 of Table 1

3.4 Extracting the Regions

In populated parts of the cities, such as downtowns, the base stations are placed very close to each other. Under high number of base stations, it is not practical to consider each station as the center of a movement to interpret the semantics of the movements. Therefore, in this work, we define regions by grouping the base stations. To this aim, we cluster base stations according to their location information (x and y coordinate attributes) using k-means algorithm [9]. There are 13281 base station ids in the original data. We
group them into clusters, which we call "regions". Then, base station ids in the preprocessed data are replaced with the corresponding region ids. For example for 100 clusters, at the end of this process, the largest cluster contains 656 base stations and the smallest cluster contains only 6 base stations. Graphical representation of the clusters for this case can be found in our previous work [10].

3.5 Extracting Frequent Patterns

The following 3 parameters are used on the user sequences obtained from CDR data in order to extract frequent patterns: pattern length, minimum support and time interval value. Pattern length describes the length of the desired frequent patterns. Minimum support describes the candidate frequent patterns’ required proportion in data. Time interval length is used for discretizing time of the day. It indicates the span of discretized time interval. Rather than using exact time information of the call record, we prefer to use discretized format for time of the day to be able to augment frequent patterns in data. Applying discretization allows us to eliminate small time differences. Each day is divided into predefined number of equal length time intervals. A call record’s time of the day information is replaced with the starting time of its corresponding interval. After this change, a call record may have more than one base station id and time of the day pair having the same time interval value.

Our extraction method can be defined as a modified version of Apriori algorithm [1]. As given in the literature, Apriori algorithm consists of two main phases, namely; candidate generation and elimination. Difference is in the candidate generation step. Normally k-length candidates (k-sequences) are generated from (k-1)-length patterns. Since this operation is costly in time for big data and we are only interested in patterns of a given length, we have changed the candidate generation phase. Our algorithm generates all candidates while traversing the data. Two index pointers are used such that one of them points the start of candidate pattern while the other one points the end of it. Count of each candidate pattern observed is recorded for elimination phase. As in the conventional Apriori algorithm’s candidate elimination phase, candidate patterns whose support value falls below minimum support are eliminated, while the others constitute the frequent pattern set. The algorithmic representation of the extraction method is presented in Algorithm 1 where n represents the desired frequent patterns’ length.

![Algorithm 1 Extraction Algorithm](image)

Table 3 demonstrates 3 sample frequent patterns for pattern length 4. In the table, the pairs separated by paranthesis represent region id and discretized time of the day. Region id and discretized time of the day are separated by comma.

<table>
<thead>
<tr>
<th>Frequent Pattern</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;(R91, 10:00), (R95, 12:15), (R45, 16:15), (R48, 18:00)&gt;</td>
<td>4.02e-6</td>
</tr>
<tr>
<td>&lt;(R91, 10:00), (R95, 12:15), (R45, 16:15), (R70, 19:00)&gt;</td>
<td>3.69e-6</td>
</tr>
<tr>
<td>&lt;(R91, 10:00), (R95, 12:15), (R45, 16:15), (R55, 19:15)&gt;</td>
<td>2.54e-6</td>
</tr>
</tbody>
</table>

3.6 Prediction

In the prediction phase, initially, test sequence of the user for whom prediction will be performed is preprocessed and formatted in the same way as frequent patterns. Assume that the test sequence is of length (k-1) and we want to predict the next step, which is the kth element, for this user’s test sequence. In order to predict the next location of a test sequence, it has to be compared with the previously discovered frequent sequences, to find if there is a matching sequence. If there exists a matching sequence, the last element of the matching pattern with the maximum support is generated as prediction. The algorithmic representation of prediction method is given in Algorithm 2.

![Algorithm 2 Prediction Algorithm](image)

Although we take this approach as our base method, we added three toleration parameters to the prediction algorithm to improve our results. These are time tolerance, multi prediction limit and length tolerance allowed for one instance.

Under time tolerance, patterns are not fixed to some time interval value anymore. They are moved forward or backward in time with a tolerance value. If one pattern is not in frequent pattern set, then toleration mechanism runs and tries to find tolerated prediction value. Assume that we have a test sequence instance as following:

\[ S_{th} = <(R91, 10:15), (R95, 12:30), (R45, 16:30)> \]

and we want to predict next location time pair for that instance but our frequent pattern set does not have a pattern starting with it but has

\[ S_{fp} = <(R91, 10:00), (R95, 12:15), (R45, 16:15), (R52, 17:00)> \]

As it can be easily seen, \( S_{fp} \) is 15 minute tolerated version
of \( S_{th} \). If toleration value for time is greater than 15 minutes, then our method gives the result (R52, 17:00) as the prediction.

The second parameter, namely the \textit{multi prediction limit} allowed for one instance, is introduced to utilize the cases in which there are more than one frequent pattern starting with test sequence instance. By adding multi prediction limit parameter, more than one prediction value are generated. This parameter puts a limit to the proportion of the total support of the patterns in the prediction set, to the total support of all patterns that start with given test sequence. All frequent patterns starting with test sequence instance are sorted in decreasing support value order and prediction set is populated by adding \( k^{th} \) elements of frequent patterns until the multi prediction limit is satisfied.

For example, \(<(R91, 10:00), (R95, 12:15), (R45, 16:15)>\) is the test sequence instance and there are frequent patterns with length 4 as given in Table 3. For this user sequence, in the single prediction method, among the matching patterns, it chooses only \((R48, 18:00)\), which has the maximum support. If the multi prediction limit is 0.5, it only gives one prediction value which is \((R48, 18:00)\). However if the limit is 0.8, then it gives two predictions which are \((R48, 18:00)\) and \((R70, 19:00)\).

The last one, namely, \textit{length tolerance} is a simple form of alignment. Since alignment does not produce successful results for location and time prediction, we decide to use a simple alignment parameter. Length tolerance can be applied in two ways. In the first way, toleration in pattern length gives us opportunity to predict one test sequence instance’s next region id by examining the shorter frequent patterns. This can be possible when the shorter frequent pattern is a subsequence of the exact test sequence instance. Assume that, we have the test sequence as following:

\[
S_{tk} = < (R77, 09:00), (R91, 12:15), (R95, 15:00), (R16, 15:00), (R22, 16:00), (R41, 16:15) >
\]

however, there is no exactly matching frequent pattern. Instead, we have the frequent pattern as following,

\[
S_{fpk} = < (R77, 09:00), (R95, 15:00), (R16, 15:00), (R22, 16:00), (R41, 16:15) >
\]

Since the sequence \( S_{fpk} \) is a subsequence of \( S_{tk} \), in which the second element of the \( S_{fpk} \) is missing, the last element of frequent pattern that starts with \( S_{fpk} \) can be given as the prediction result for input sequence \( S_{tk} \) if the length tolerance is greater than 0.

Second way of tolerating the pattern length gives us opportunity to predict the next region by examining the longer frequent patterns. This is possible when the longer frequent pattern contains the exact test sequence instance (order of region ids are important) but also contains some additional region ids. Assume that, we have the following user sequence:

\[
S_{tk} = < (R77, 09:00), (R91, 12:15), (R95, 15:00), (R16, 15:00), (R22, 16:00), (R41, 16:15) >
\]

however, there is no exactly matching pattern. Instead, we have the frequent pattern that starts with the following,

\[
S_{fpk} = < (R77, 09:00), (R91, 12:15), (R95, 15:00), (R18, 15:00), (R16, 15:00), (R22, 16:00), (R41, 16:15) >
\]

Since the sequence \( S_{fpk} \) contains \( S_{th} \) in the same order, which also has the fourth element \((R18, 15:00)\) as the difference from the test sequence instance, last element of frequent pattern that starts with \( S_{fpk} \) is given as the prediction result for test sequence instance \( S_{tk} \) if the length tolerance is greater than 0.

4. EVALUATION

In this section, first we introduce our evaluation method and evaluation metrics, and then we give the experimental results for the method explained in the previous section.

4.1 Evaluation Method and Metrics

In order to assess the quality of the predictions, we used 5-fold cross validation on a real CDR data set that has been introduced earlier. In the training phase, proposed frequent pattern extraction method is applied on the training data. In the test phase, firstly, the test data is processed as in the training phase to extract all sequential patterns, except this time with no minimum support, in order to generate all test sequence patterns. For each one of the test sequences, prediction algorithm introduced in the previous section is applied to predict the last elements of these sequences. The result of the prediction is compared against the actual last element of the test sequence.

We used three different metrics, namely \( g \)-accuracy, \( p \)-accuracy and prediction count to evaluate our prediction method’s quality.

\textbf{Accuracy} measures how much of our predictions match with exact next region id of the test sequence. It simply can be defined as the ratio of true predictions to the all predictions. In our case, we have two types of accuracy. The first one, which is the \( g \)-accuracy (general accuracy), is the ratio of number of true predictions to the number of all patterns with the same length in the test set. The second one, which is the \( p \)-accuracy (predictions’ accuracy), is the ratio of the number of true predictions to the number of all predictions we are able to make. The reason for using two different accuracy calculation is due to the fact that the proposed algorithm may not be able to generate prediction for each of the test instances, if there is no matching frequent pattern found for the test sequence instance. In the first form of accuracy calculation, the accuracy result superficially drops for such cases.

\textbf{Prediction count} metric is required because of the multi prediction limit parameter. It quantifies the size of the prediction set when correct prediction result is in the prediction set.

4.2 Experimental Results

In this section, the results of the experiments of the proposed method under different parameters are given.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline
\textbf{Pattern Length} & \textbf{$\bar{S}_{tk}$} & \textbf{prediction} & \textbf{prediction} & \textbf{prediction} & \textbf{prediction} & \textbf{prediction} & \textbf{prediction} \\
\hline
\textbf{15:00} & (R52, 17:00) & (R52, 17:00) & (R52, 17:00) & (R52, 17:00) & (R52, 17:00) & (R52, 17:00) & (R52, 17:00) \\
\hline
\end{tabular}
\end{table}
For this set of experiments, length tolerance is 2, time interval length is 60, time tolerance is 120, multi prediction limit is 0.8, cluster count is 100, and minimum support is 4e-7.

Figure 1: Pattern Length vs g-Accuracy

As it can be seen from Figure 1, when pattern length increases, g-accuracy decreases. Since it is much harder to find frequent patterns for longer patterns, g-accuracy eventually decreases. However as can be seen in Figure 2 p-accuracy do not have a continuous increase as expected when pattern length increases. The reason for this different behaviour is the sudden decrease of prediction count where pattern length increases from 5 to 6 which can be seen in Figure 3. Since prediction count for pattern length 5 is way larger than the one for pattern length 6, its p-accuracy is also greater than the one for pattern length 6.

Figure 2: Pattern Length vs p-Accuracy

Minimum Support

For this set of experiments, pattern length is 5, length tolerance is 2, time interval length is 60, time tolerance is 120, cluster count is 100, and multi prediction limit is 0.8.

As it can be seen from Figure 4, when minimum support value increases, g-accuracy decreases. It is an expected behaviour to have smaller g-accuracy values for the greater minimum support values since the greater minimum support value means the fewer number of frequent patterns, which eventually causes the number of predicted sequence to drop superficially. Consequently the g-accuracy decreases.

Figure 3: Pattern Length vs Prediction Count

As it can be seen from Figure 5, when minimum support value increases, p-accuracy decreases. When compared to the effect on g-accuracy, decrease in p-accuracy is much less than that of g-accuracy. The reason is related with the definition of p-accuracy. P-accuracy does not take unpredicted sequences into consideration. However still there is a small decline in the graph, since the number of extracted frequent patterns is much lower for the greater minimum support values. As it can be seen in Figure 6, when minimum support value increases, prediction count decreases. The reason is the same with the previous two graphs; due to the lower number of extracted frequent patterns.

Figure 4: Minimum Support vs g-Accuracy

Figure 5: Minimum Support vs p-Accuracy

Figure 6: Minimum Support vs Prediction Count
Figure 6: Minimum Support vs Prediction Count

*Length Tolerance*

For this set of experiments, pattern length is 5, time interval length is 60, time tolerance is 120, multi prediction limit is 0.8, cluster count is 100, and minimum support is 4e-7.

As it can be seen in Figure 7 and 8, when length tolerance increases g-accuracy and p-accuracy increases. Increasing length tolerance makes some unpredicted test sequences predictable which increases the g-accuracy. This also increase p-accuracy since true predicted sequences increase although its increase is much lower than the g-accuracy. Moreover, as it can be seen in Figure 9 higher length tolerance makes prediction sets larger.

Figure 7: Length Tolerance vs g-Accuracy

Figure 8: Length Tolerance vs p-Accuracy

Figure 9: Length Tolerance vs Prediction Count

*Multi Prediction Limit*

For this set of experiments, pattern length is 5, length tolerance is 2, time interval length is 60, time tolerance is 120, cluster count is 100, and minimum support is 4e-7.

As it can be seen in Figures 10 and 11, when multi prediction limit increases g-accuracy and p-accuracy increase. The higher multi prediction limit means the larger prediction set for one test sequence. Therefore it causes the both of the accuracy metrics to get higher values. In addition to it, by definition, prediction count also increases when multi prediction limit increases which can be seen in Figure 12.

Figure 10: Multi Prediction Limit vs g-Accuracy

Figure 11: Multi Prediction Limit vs p-Accuracy

Figure 12: Multi Prediction Limit vs Prediction Count
For this set of experiments, pattern length is 5, length tolerance is 2, time interval length is 60, time tolerance is 120, multi prediction limit is 0.5 and minimum support is 4e-7.

**Cluster Count**

As it can be seen from Figure 13, when cluster count increases g-accuracy decreases slightly. It is due to the increase in the number of the unpredicted test sequences since increasing cluster count makes frequent patterns harder to find but more accurate. Therefore, they usually give correct predictions when compared to ones with the fewer number of clusters. It also eventually decreases the size of prediction set i.e., prediction count as seen in Figure 15.

**Time Interval Length**

As it can be seen in the Figure 16 and 17, when time interval length increases, g-accuracy and p-accuracy increase. Since the larger time interval means more similar daily sequences and eventually higher number of frequent patterns, increase in the values of the both accuracy metrics is an expected behaviour. Although it decreases for time interval length 360, we can say that prediction count increases in general since it is a negligible tiny decrease. As can be seen in Figure 18 the reason for increase in the size of the prediction set is the same reason for the increase in g-accuracy and p-accuracy; due to the higher number of frequent patterns.

**Time Tolerance**

As it can be in from Figure 19 and 20, when time toler-
Figure 17: Time Interval Length vs p-Accuracy

Figure 18: Time Interval Length vs Prediction Count

Figure 19: Time Tolerance vs g-Accuracy

Figure 20: Time Tolerance vs p-Accuracy

Figure 21: Time Tolerance vs Prediction Count

5. CONCLUSION

In this work, we focused on the problem of predicting the next location change of the mobile phone users' and the time these changes take places. The main novelties of this work are predicting the location and the time of the change together and introduction of the spatio temporal alignment of user logs. In experiments, we have evaluated our model’s prediction quality with respect to g-accuracy, p-accuracy and prediction count and further analyzed the effects of several parameters on the accuracy. Some basic findings of our previous work in [10] and most important results for new method are presented in Table 4. Prediction counts (P. Count) are presented as the ratio of the actual prediction count to the maximum size of the possible prediction set for the given parameters. Maximum size of the possible prediction set is calculated with the formula of [Number of Regions] x [Number of Time Intervals] in one day. Since no temporal information is used in our previous method in [10], for its entries possible prediction set represents only the cluster count. For the third entry 6400 clusters are used, and 60 minute time intervals are preferred which makes maximum size of the possible prediction set 6400 x 24, in other words 153600. For the fourth entry 100 clusters are used, and 15 minute time intervals are preferred which makes maximum size of the possible prediction set 100 x 96, in other words 9600.

As it can be seen from the Table 4, while g-accuracies superficially drops for the new method, p-accuracies increase...
Table 4: Results for Previous and the New Proposed Methods

<table>
<thead>
<tr>
<th>Method Name</th>
<th>G-accuracy</th>
<th>P-accuracy</th>
<th>P. Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method in [10]</td>
<td>58.48%</td>
<td>73.96%</td>
<td>2.4/100</td>
</tr>
<tr>
<td>Method in [10]</td>
<td>84.79%</td>
<td>92.48%</td>
<td>14/100</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>41.64%</td>
<td>86.94%</td>
<td>3.44/153600</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>48.28%</td>
<td>92.8%</td>
<td>14.76/9600</td>
</tr>
</tbody>
</table>

for similar prediction counts. Since including temporal information in daily sequences makes finding matching frequent patterns for test sequences harder, the number of non-predicted test sequences increases, eventually g-accuracy decreases. However since our frequent patterns are more precise compared to the previous method’s, accuracy of the next location of the predicted test sequences increases, eventually p-accuracy increases as well.

As a future work, we plan to enlarge our problem space towards the following sub-areas; predicting the time of the next action or predicting both the time and the location of the next action using spatio-temporal data.

6. ACKNOWLEDGEMENT

This research was supported by Ministry of Science, Industry and Technology of Turkey with project number 01256.STZ.2012-1 and title "Predicting Mobile Phone Users' Movement Profiles".

7. REFERENCES


