# Next Place Prediction by Learning with Multiple Models

Zhongqi Lu, Yin Zhu, Vincent W. Zheng, Qiang Yang
Hong Kong University of Science and Technology, Hong Kong
{zluab, yinz, vincentz, qyang}@cse.ust.hk

## ABSTRACT

Location prediction has been an active research area for a long time. The Task 2 of Nokia Mobile Data Challenge (MDC) [6] provides plenty of data collected using mobile phone and the transitions from one place to another. Given a user currently at one place and the context associated with this visit, we are interested in the relationship between the current context and the next visit and the prediction of the next place the user is going to. We focused on using the transitions between places for each individual user, as well as the time context, to do prediction. We also tried to explore other context information such as call-logs and accelerometer data in the current place. By combining the location transition information and the context information, we provide an ensemble learning approach to pick the appropriate model for each user. Our experiments show that our solution is effective in solving the next place prediction task.

## 1. INTRODUCTION

The Task 2 of the Nokia MDC is to predict the next destination of a user given the current context. The challenge participants are free to estimate the context from all available data within certain time intervals corresponding to a visit in a place. Sensor-based next place prediction has been studied recently [4, 7]. However, we found it hard to apply the approaches in previous researches, because most of them have the knowledge of the exact GPS point and the road map. It is also hard to use the interaction among users [10], as the location history for each user is anonymized. Therefore, we are more interested in exploiting the location transition history, as well as the context information, to predict the next location.

Many people tend to have some routine activities in their life. For example, a working person may get up at home in the morning, and then move to his/her office in the weekdays. For the user with such a regular timetable, it would be reasonable to predict his/her next destination based on the time information, current place, etc. In order to capture this transition information, we are motivated to use historical trajectory data to build transition probability matrices among the places. Another interesting observation is that, people can have different routines in different time periods. For example, in the weekdays, working people may go to office; but in the weekends, they may go for shopping. Such routine difference motivates us to use the time as an important factor in computing the location transition probabilities. We have developed two location transition models to encode such knowledge in predicting a user's next destinations. The first one is simply using the transition probability computed from the (recent) history data to predict the next more probable location. The second one further differentiates the location transition probabilities at different time periods, and thus improves the performance.

Other than the location transition patterns, there is also some context information that can possibly useful in next place prediction. In particular, we are interested in the following question: given the current place ID as a prior knowledge, are the user activity data, say call-logs and accelerometer strength, good indicators for the next place? We have tried to answer this question by transforming the next place problem as a supervised learning problem. Specifically, we developed two models to solve the prediction problem. The first model is a classification model, where we take the sensor logs as input and use the next place ID as output. The other model is a ranking model, where we try to rank each pair of locations based on their transition counts. In this ranking model, we take as input the sensor data for each pair of location, as well as some feature denoting their transition. Then we rank different pairs of locations according to their transition counts. The basic idea is to train a model that can minimize the number of inversion in ranking. It is worth noting that, such a ranking model can possibly be more robust than a regression model because the transition counts may vary a lot as regression output [10].

Effectively, the above mentioned location transition matrix based models and the supervised learning models capture the different type of information. We have managed to combine them all through ensemble learning to get a better prediction for each user.

Finally, we feature our solution as follows:

- We provide two location transition matrix based mod-

This material was prepared for the Mobile Data Challenge 2012 (by Nokia) Workshop; June 18-19, 2012; Newcastle, UK. The copyright belongs to the authors of this paper.

els, which are able to use each individual user's location transition history for next place prediction.

- We also provide two supervised learning models, which are able to use the general context information from all the users' data together for next place prediction.

- We offer an ensemble learning solution to combine these different models, in order to assign appropriate models to each user in prediction.

- We have shown to achieve 55.3% accuracy in testing with the real data in Task 2.

# 2. RELATED WORK

Typical location prediction problems assume that some sensor observations such as WiFi [5], GSM [9] are already known at the time point of prediction. The task is then to use these sensor signals, in an either geometrical [2] or machine learning [1] way, to discover the location at the current moment (i.e. to predict the *current location*). However, the Task 2 of the Nokia mobile data challenge is interested in another type of location prediction problem, which aims to predict the next location without knowing the future sensor data (i.e. to predict the *future location*). This is essentially more challenging than the previous problem setting, since we do not have the observation in the future. We need to greatly rely on the historical data more than ever to solve this problem. Nevertheless, the previously proposed sequential learning models in current location prediction such Hidden Markov Model [5], Conditional Random Field [8], Particle Filter [1], cannot be simply applied anymore, because they require the (sensor) observations to be known for the time point of prediction. In our solution, we show to use the time-sensitive location transition patterns learned from historical data to predict future location, without requiring the future location's real-time observation data to be given beforehand.

In future location prediction, sometimes pure historical location data (and optionally the corresponding sensor data as well) may not be sufficient. Therefore, previous research has exploited some additional information to help the prediction by providing more constraints on the possible locations to go. For example, Krumm has shown to use the road map to help predict where drivers turn in the road [4]. Monreale et al. argue that if people use the movements of all objects in a certain area to learn a classifier, the next location prediction can be greatly improved [7]. However, such additional information is not always available in our task. For example, in our data, there is not road map since the exact GPS location has been anonymized and there is no way for us to uncover the true location for each user. Besides, because in our data each user has their unique location set and we cannot tell whether two users share the same location or not, it is impossible for us to use the multiple users' information together like [7] for prediction. Given such challenges, we chose to do the prediction in two ways. First, we focus on doing prediction for each user independently, and only consider their own location transition history. Second, we manage to extract features that are general for the users, and use them to train some unified supervised learning models for prediction on all the users.

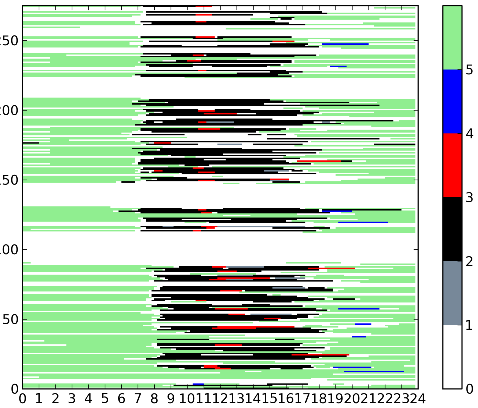


**Figure 1: The visiting log of user 189**

# 3. INDIVIDUAL USER STUDY

We notice that in the given data set, the user information can be hardly reused between different users because of the anonymization preprocess. Therefore, it is difficult to exploit those collaborative filtering idea [10] to help the prediction. In this task, we restrict ourselves on exhaustively mining the individual information. We are motivated to analyze the individual's behavior. In this section, we report several interesting findings on the individual user behaviors.

## 3.1 Most Users Have Regular Timetables

In order to better understand the user's behaviors, for each of the users, we retrieved the place record and visualize the user's visiting logs during the information collection period. Figures 1, 2 and 3 give the visualization of visiting logs for user 189, user 13, user 45 and user 110. In these figures, the Y-axis represents the day, where day 0 corresponds to the first day of the user's training record; And the X-axis is divided into 24 grids, which correspond to 24 hours of a day. We use different colors to represent different places and blank (white color) means no available visiting log. For the convenience of presentation, only top 5 most frequently visited places are visualized.

We find that for most of the users, like the user 189, they have a relatively stable timetable. As we can see, user 189 stays at a place represented by light green (maybe the home) in the evenings, and goes to the place represented by black in the working hours. By our study, there are about 60 out of 80 users can be classified as having a regular timetable. For those users who have a regular timetable, their next visits are highly predictable simply based on the historical trajectory data.

## 3.2 Recent Data Is More Important

We also notice that during the data collection period, some of the users may change their regular visited places due to various reasons. For example, people may move to another apartment, or get another job, etc. Take user 13 as an example. As shown in Figure 2, this user seems to change the living places since the place in the evening is different in the last 100 days of the training record. Among those 60 regular users, there are about 20 users who changed their regular

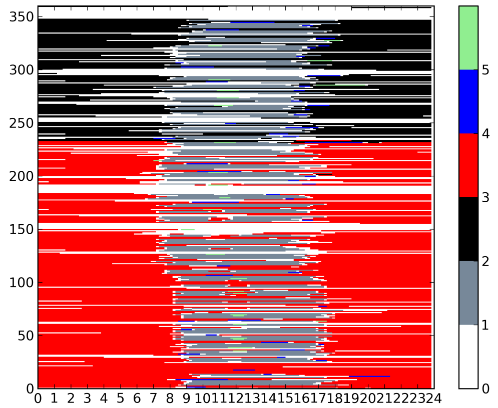

**Figure 2: Users, such as user 13, may change their regular visited places**

visited places during the data collection period.

Given that the users can change their regular visited places, the information before changing the regular visited places would become less useful in predicting the next visits. Therefore, in our approach, we use only use the most recent data for these 20 users who changed the regular visited places. In Section 5.1, we study the impact of using different amount of most recent data for training the transition matrix. We also show that the experimental results justify our assumption that recent data is more important for the users who change their regular visited places.

## 3.3 Some Users Have Irregular Timetables

It is worth noting that there are around 20 users having relatively irregular timetables. As shown in Figure 3, most of these users either have very limited logs to identify the patterns or do not keep a regular timetable. The first case corresponds to the cold start problem, that is, the user has very few records such that we cannot build effective models to predict the next visit. The second case is for those users, like taxi drivers, who are always moving around and the destination is not determined. These users are generally hard to deal with in prediction. We follow the intuitive solution and suggest the most frequently visited places as their destinations.

# 4. MODELS

## 4.1 Transition Matrix Models

As most users have regular timetables, we propose to formulate the location transition probability to help predict the next location. In this section, we first introduce the transition matrix, and then show several extensions of the transition matrix.

### *4.1.1 Simple Transition Matrix Model*

The transition matrix is built purely based on the visiting logs on the whole training data set. We count 1 for the matrix entry $(i, j)$ if there is a transition from the place $i$ to the place $j$. The transition matrix captures the probability of the transition from current place $i$ to the destination $j$. Given the user to be at place $i$, in order to predict the next visit, we retrieve the $i$-th row of the transition matrix and obtain the next place by finding the column with maximum (probability) value in the row.

### *4.1.2 Enhanced Transition Matrix Model using Time as a Prior*

One problem for the previous simple transition matrix model is that, it does not consider the mobile users' behavior difference in different time periods. We find in the data that, in the weekdays, most people have regular timetable in the weekdays; but in the weekends, people tend to have more choices to visit different locations. Therefore, we are motivated to separate the data from weekdays and weekends in constructing the transition matrix. For a next place prediction on weekdays, we use the transitions obtained on weekdays. For weekends, we only use the transition matrix during weekends.

In addition to weekday/weekend, we also find it useful to study the behavior different in different time periods in a day. Specifically, we further divide a day into several time spans and use the data accordingly to build the transition matrices. Our experimental results show that by dividing the day into three time spans, i.e. 0:00-8:00, 8:00-16:00 and 16:00-24:00, we can further improve the performance. The reason for choosing such time spans is that, by dividing a day into such three time spans, we can consider the time before going to work, the working hours and the time after work separately. Because most people do not change their residence place and working place, given such a time interval as the prior, the search space for next visit can be greatly reduced. This leads to a better performance.

## 4.2 Supervised Learning Models

While being efficient and capable to make good next place predictions, the transition matrix based models have obvious drawbacks. First, they take only the visit logs as input and can hardly use the rich context information. Second, it cannot make reasonable predictions for the cold start users. Therefore, we further exploit the use of context features.

### *4.2.1 Context Features*

We extract a set of useful features, which are associated to a place and can be applied to our classification model and ranking model.

- **Time features.** We compute the mean and variance of stay durations at a place.
- **Accelerometer features.** We use the mean and variance of the accelerometer strength in a minute as features.
- **Application features.** We count the frequency of each application status, such as "close", "started", "view" and "foreground".
- **Bluetooth and WLAN features.** The statistics including mean, variance, minimum and maximum of the signal strength are used.

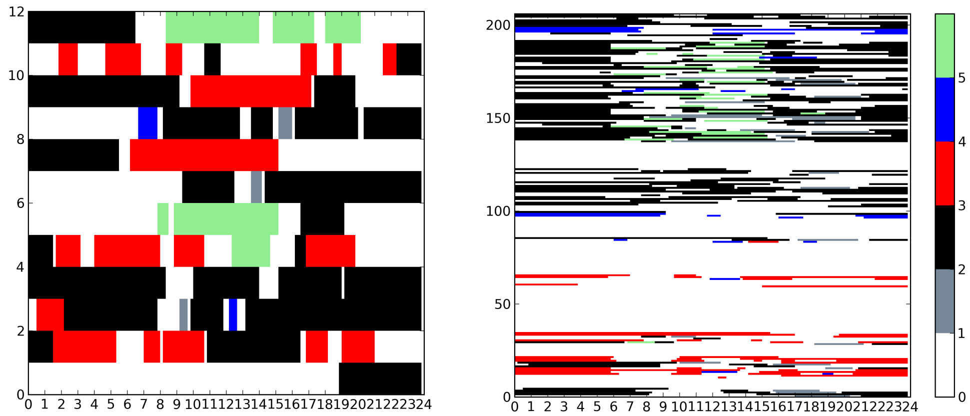


**Figure 3: Irregular users**

- **Call-log features.** We used the features including ratio between incoming and outgoing and ratio of missing calls.
- **System features.** We count the frequency of the phone status, such as "charging" and "silence".

The above features are extracted and associated with a place ID. We used these features in both our classification model and ranking model.

### *4.2.2 Classification Model*

The next place prediction can be viewed as a classification problem. The input is the context features for current location, and the output is the next location ID. Due to the anonymization process, one user's records cannot help to predict another's next visit. Therefore, we train a support vector machine (SVM) for each user. One challenge to directly use such a classification model is that, there can be too many labels, which make the classification difficult. In order to address this challenge, we leverage our observation that some places are visited for quite few times and they are less likely to be visited again. Therefore, we propose to group all these rarely visited places as a dummy place to reduce the number of classes for prediction.

### *4.2.3 Ranking Model*

The next place prediction can also be seen as a ranking problem. In the classification model, we take the information at current place as input and the next place as output. Effectively, we miss the links between the current place and the next potential visit. In order to use such links, we can take every possible location as a potential next place to visit for a user, then we construct a set of location pairs with the current place and each of these possible locations. Each pair of such location denotes a possible transition, and our goal is to rank these pairs and find the one ranked highest as the most probable transition for prediction. In training, the input data is a rank list. Each position in the list is a pair of locations, and it has a feature vector composed of both locations' context features and their location transition features. Some typical location transition features include:

- Frequency of next place being visited.
- Frequency of visits in current time interval.
- Start time of next place, given the end time of current place.
- Frequency of next place being visited, given the end time of current place.
- Frequency of next place being visited, given the current place.

The rank list is ordered by considering the transition counts of each location. Then, we adopt the idea in [3] to implement the ranking model. In testing, given a current place, we can still get the context feature for current place and the transtion features w.r.t. each possible next location. However, we do not have the real-time context features for these possible next locations. We propose to re-use the context features for these possible next locations from their training data. Though it may not be perfect, we empirically show in the experiments the effectiveness of using such features to deliver a ranking model. Finally, with the complete feature vector for each location pair, we predict the rank and find the most probable transition from the current place.

## 4.3 Ensemble Learning

To solve the next place prediction problem, we have proposed four models, which use different information in different ways. A reasonable choice to combine these different models is ensemble learning. In this task, we adopt an embarrassingly simple, yet effective ensemble learning solution. Specifically, as different users tend to have different behavior patterns, we may not expect a single model can outperform the others on all users. Therefore, we propose to use the most suitable model (out of the four proposed models) for each user individually. We empirically show in the experiments that, the best model chosen in the validation data also works well on the test data. This ensures us to use such a simple ensemble learning solution to guarantee the best performance from a bunch of base models. We will discuss more on ensemble learning in the experiment section.

Table 1: Consensus of prediction accuracy on validation set and testing set

| User | | 001 | 003 | 008 | 014 | 015 | 020 | 021 | 022 | 025 | ··· |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Methods on Validation Set | transSimple | 0.1538 | 0.2857 | 0.3684 | 0.3214 | 0.3667 | 0.4500 | 0.3913 | **0.7059** | 0.2308 | |
| | transTime | 0.3077 | **0.3571** | **0.5263** | 0.3929 | 0.0667 | **0.7110** | **0.4348** | 0.5882 | 0.4782 | |
| | Classification | **0.3462** | 0.2755 | 0.4210 | 0.3571 | **0.4194** | 0.6667 | 0.4310 | 0.5790 | 0.4118 | |
| | Ranking | 0.3249 | 0.3073 | 0.4198 | **0.4053** | 0.3995 | 0.6833 | 0.4326 | 0.5638 | **0.5384** | |
| Methods on Testing Set | transSimple | 0.2258 | 0.4000 | 0.4623 | 0.3077 | 0.5333 | 0.4737 | **0.3478** | **0.7632** | 0.1539 | |
| | transTime | 0.5161 | **0.5333** | **0.4959** | 0.2308 | 0.5333 | **0.6316** | 0.2609 | 0.5790 | 0.2500 | |
| | Classification | **0.5484** | 0.4667 | 0.4737 | 0.2508 | **0.5667** | 0.6153 | 0.3044 | 0.5020 | 0.3077 | |
| | Ranking | 0.5042 | 0.4789 | 0.4901 | **0.3211** | 0.5121 | 0.6229 | 0.3321 | 0.5598 | **0.3597** | |

**Table 2: Percentage of Heuristic Data in Use. We obtain best prediction accuracy when using 70% data.**

| Percentage (%): | 30 | 50 | 70* | 90 | 100 |
|---|---|---|---|---|---|
| Accuracy (%): | 46.23 | 48.68 | **51.21** | 49.73 | 49.55 |

# 5. EXPERIMENTS

In order to test our methods, we constructed the testing set by splitting out the latest 10% of each user's training data and retaining those with trusted transitions, which we believed simulated the test data set held by the MDC organizer. And we also constructed the validation set by retrieving another latest 10% of each user's training data. The rest 80% of the data are used as the new training data set. In the followings sections, we adjust the essential parameters for our models based on the prediction accuracy on the validation set. And we report the performance of the models based on the prediction accuracy on the testing set.

## 5.1 Using Different Amount of Training Data For Transition Matrix

As mentioned in Section 5.1, since the transition matrix model is very sensitive to the data quality, we study the impact of using different amount of most recent training data to build the model for those 20 users who changed their regular visited places. The overall prediction accuracies are shown in Table 2. We find that, on one hand, when only very few recent data are used in training, the performance is not very good due to the lack of information. On the other hand, when the entire training data set is used in training, the noise mentioned in can make it hard to make proper predictions. Based on the experimental results, we finally use the most recent 70% of the training data for our further experiments and submissions. This is also consistent with our investigation on each individual user's data. For example, as shown in Figure 1 and Figure 2, which visualize the location history of two typical users with regular timetables, we find that using the most recent 70% of the training data is also a reasonable choice for these types of users.

## 5.2 Ensemble

### *5.2.1 Consensus of Prediction Accuracy on Validation Set and Testing Set*

**Table 3: Models and Their Descriptions**

| Model | Description |
|---|---|
| transSimple | The Naive transition matrix model |
| transTime | Transition matrix with time as a prior |
| Classification | Classification model. Here, we adopt SVM |
| Ranking | Ranking model implemented with learning to rank algorithm |

**Table 4: Overall accuracy of single models and the ensemble. The ensemble of single models outperforms single models.**

| Models | Accuracy (%) |
|---|---|
| transSimple | 41.9 |
| transTime | 51.2 |
| Classification | 49.6 |
| Ranking | 52.7 |
| **Ensemble⋆** | **55.3** |

Before we ensemble the proposed models to give the final predictions on testing set, we would like to verify that our validation set and testing set have consensus on the prediction accuracy for each user, so that we can choose a proper model for each user based on the prediction results on validation set. Table 1 lists the prediction accuracy of the four of our prime models on both validation set and the testing set. Notice that in most cases, the model which did the best on the validation set also achieves the best performance on the testing set. This gives us the reason to believe that such a simple ensemble learning solution is likely to work well.

### *5.2.2 Result of Ensemble*

By selecting the best model for each user according to the prediction on the validation set, most users are assigned the most suitable model. As shown in Table 3, the overall prediction accuracy after ensemble is improved over each single model. On our test data set, the best overall prediction accuracy can reach 55.3%.

# 6. TOOLS AND COMPUTATION

In Task 2, we have encountered little problem in computation. Although the training dataset occupies 48GB disk space, the data that our models rely on are small. We choose F# and Python as our data processing languages. We use

Python first and all our visualization is done using Python's `matlibplot`[1] package. But later, we have to coordinate the programming with that in our Task 1 [11], which uses F# extensively for the feature extraction. Under F#'s strictness and expressive power in types, we write programs with few bugs.

# 7. CONCLUSION

Next place (or future location) prediction is an important task. It is different from the traditional location prediction problem, which aims to predict the current location with (sensor) observation input. This makes the previously proposed sequential model such as Bayesian filtering and Conditional Random Fields not applicable to this task. To address this issue, we chose to build two time-sensitive location transition models, by formulating how likely a user will move from one place to another place at different time periods. Besides, in Task 2, as each user's data are anonymized, it is difficult for us to further exploit the correlation among the users by just looking at their location data. Therefore, we compromise our way of using multiple user's data by extracting the general context features. These general features, independent of users, are further consumed by two supervised learning models, in order to formulate the different ways of modeling. Finally, given these individually learned models, we adopt some ensemble learning model to further assign each user with the suitable model. Overall, we show to achieve 55.3% accuracy for our best model in the test data.


# 8. ACKNOWLEDGMENT

We thank the support of Hong Kong RGC GRF projects 621010 and 621211. We thank Xing Xie and Nathan N. Liu for discussions.


# 9. REFERENCES


[1] D. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello. Bayesian filtering for location estimation. In *Proc. of IEEE Pervasive Computing and Communication*, volume 02 of *PerCom '03*, pages 24–33, 2003.

[2] A. Goldsmith. *Wireless Communications*. Cambridge: Cambridge University Press, 2005.

[3] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142, New York, NY, USA, 2002. ACM.

[4] J. Krumm. Where will they turn: predicting turn proportions at intersections. *Personal and Ubiquitous Computing*, 14(7):591–599, 2010.

[5] A. M. Ladd, K. E. Bekris, A. Rudys, L. E. Kavraki, D. S. Wallach, and G. Marceau. Robotics-based location sensing using wireless ethernet. In *Proc. of the 8th Annual International Conference on Mobile Computing and Networking*, MobiCom '02, pages 227–238, New York, NY, USA, 2002.

[6] J. K. Laurila, D. Gatica-Perez, I. Aad, J. Blom, O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Proc. Mobile Data Challenge by Nokia Workshop, in conjunction with Int. Conf. on Pervasive Computing*, Newcastle, UK, 2012.

[7] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti. Wherenext: a location predictor on trajectory pattern mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 637–646, New York, NY, USA, 2009. ACM.

[8] R. Pan, J. Zhao, V. W. Zheng, J. J. Pan, D. Shen, S. J. Pan, and Q. Yang. Domain-constrained semi-supervised mining of tracking models in sensor networks. In *Proc. of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 1023–1027, New York, NY, USA, 2007.

[9] O. Veljo, V. Alex, L. Anthony, and de Lara Eyal. Accurate gsm indoor localization. In *Proc. of the 7th International Conference on Ubiquitous Computing*, UbiComp '05, pages 141–158, 2005.

[10] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Towards mobile intelligence: Learning from gps history data for collaborative recommendation. *Artif. Intell.*, 184-185:17–37, 2012.

[11] Y. Zhu, E. Zhong, Z. Lu, and Q. Yang. Feature engineering for place category classification. In *Proc. Mobile Data Challenge by Nokia Workshop, in conjunction with Int. Conf. on Pervasive Computing*, Newcastle, UK, 2012.


[1] `http://matplotlib.sourceforge.net/`