Predicting Mobile Application Usage Using Contextual Information

Ke Huang
Department of Computer Science,
University of Massachusetts Lowell
khuang@cs.uml.edu

Xiaoxiao Ma
Department of Computer Science,
University of Massachusetts Lowell
xma@cs.uml.edu

Chunhui Zhang
Department of Computer Science,
University of Massachusetts Lowell
czhang@cs.uml.edu

Guanling Chen
Department of Computer Science,
University of Massachusetts Lowell
glchen@cs.uml.edu

ABSTRACT
As the mobile applications become increasing popular, people are installing more and more Apps on their smart phones. In this paper, we answer the question whether it is feasible to predict which App the user will open. The ability for such prediction can help pre-loading the right Apps to the memory for faster execution or help floating the desired Apps to the home screen for quicker launch. We explored a variety of contextual information, such as last used App, time, location, and the user profile, to predict the user's App usage using the MDC dataset. We present three findings from our studies. First, the contextual information can be used to learn the pattern of user's App usage and to predict App usage effectively. Second, for the MDC dataset, the correlation between sequentially used Apps has a strong contribution to the prediction accuracy. Lastly, the linear model is more effective than the Bayesian model to combine all contextual information and for such predictions.

Author Keywords
Mobile, Application, Prediction, Context

ACM Classification Keywords
H.5.2 Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms
Algorithms, Design, Experimentation, Performance.

INTRODUCTION
As the users install more Apps on their phones, it can become challenging to find the desired App to open given many icons across multiple screens. If there is an on-board service that can accurately predict which App the user is likely to open next, we could build an intelligent launcher that surfaces the icons of those Apps that are likely to be used to the front screen so the user can quickly find what she wants. In addition, the launcher can even pre-load the App into the memory for faster execution.

In this paper, we explore how to build an App prediction service. Intuitively, which App the user is likely to use depends on some contextual information. For example, the Apps used when the user is having a dinner with friends could be quite different than the Apps when she is working in the office. Thus we set out to answer following questions. First, what context information is relevant for App prediction? Second, how to extract and represent each context source? And lastly, how can we combine different context sources to predict App usage effectively?

We used the MDC dataset for this research. We started with extracting contextual information from the raw data. Then we used time, location, user profile, and the last-used App as the context to predict the next App to be opened. In order to represent the user's location, we proposed an algorithm to extract the user's significant places from the WLANs that were seen by the phones. Next, a Bayesian model and a linear model, which use different approaches to combine contextual information, were introduced for the prediction. Finally, the contribution of different contextual information and the results of two models are compared with the benchmark to show the effectiveness of the prediction. The study of this work provides evidence that the contextual information can help to learn and predict the App usage pattern effectively.

RELATED WORK
There has been a lot of work on context-aware mobile computing, such as Gellersen et al.[4] and Dix et al.[2], which study how to augment mobile devices with contextual information. Their mainly focus lies in the integration of generic context sensors, especially the location and visual context. On the other hand, Do[3] and Kang et al.[5] try to mine large-scale Apps usage data to better understand the user behaviors. They have not, however, integrated any contextual information. The AppJoy extends the use of these patterns

UbiComp '12, Sep 5-Sep 8, 2012, Pittsburgh, USA.

for App recommendation [8]. Overall, there has not been much work on leveraging context for usage prediction of mobile Apps.

PREDICTION APPROACH

In this section we discuss how to predict the usage of mobile Apps by using the Mobile Data Challenge (by Nokia) dataset. First, what and how the contextual information related to the prediction is extracted from the raw dataset. In Section Contextual Information, We represent an algorithm to find user's significant places, which is an important context for the prediction model. The procedure of how we process the App usage history is described in this section as well. Finally, we introduce two models to combine all the context and predict the App usage in Section Prediction Models.

Data Description

The MDC data available for the challenge contains 38 participants' data. All the data was collected from smart-phones in the course of more than one year [6]. The rich set of data related to location, social interactions (including social proximity), contextual attributes, media consumption/creation, application usage and device control is available. According to the available questionnaires submitted by participants, there are 20 male and 8 female participants. Most of (25 out of 29) participants' age range between 22 to 44 years old. And 13 participants are working full time, while the other 16 participants are working part time.

Contextual Information

In this research, we hypothesize that the Apps a mobile phone user uses depend, to some degree, on the user's context. Hence, in order to predict App usage as accurate as possible, context needs to be extracted from raw dataset. The MDC data available for the challenge contains 38 participants' data. The rich set of data related to location, social interactions (including social proximity), contextual attributes, media consumption/creation, application usage and device control is available[6]. In this paper, we only focus on the context of time, location, user profile and the latest used App. The time context, latest used App context, and user profile context are extracted from the App usage history and mobile phone system logs. While the location context is extracted from the WLAN devices seen by the phone. We will describe how to extract these contextual information in following sections.

Time and Location

The most common and widely used contextual information is time and location. A smart phone user more or less tends to follow a pattern with time and location to launch an App. We propose there are two kinds of time patterns in two scales. The first time pattern scales in day of week. For example, a user is more likely to check emails, or receive calendar notifications during work days. While it is more probable to play some games during weekends. This pattern is applicable to almost all phone users to a certain extent, since we all have routine activities more or less. Though it is general to almost all smart phone users, it is not fine-grained. Hence, we introduce the second time pattern that scales in hour of day. This pattern is not general to all but varies between different users. For instance, one is tending to check the weather report, before she goes to work, between 6AM to 7AM. While other people are more likely to read newspapers. By combining and analyzing these two time patterns, user's routine activities of App usage can be learned.

The location context is strongly correlated to time that the user's position is determined by the time. Due to energy constraints, in general, the GPS reading of the MDC data are available during outdoor movements only. And a heuristic state machine has been used to determine when to activate GPS. Although there are some work in using GPS to predict user's next location[7] and learning user's significant location patterns for prediction[1], they are not fine-grained enough whether a user is in a meeting room or in an office. Moreover, a user seldom uses any App during outdoor movements. While when the GPS reading is not available indoors, the MDC data cannot guarantee that the last GPS reading is the place where the user is currently in, since when to activate the GPS is unknown to us. Hence, we cannot use the GPS data to identify user's significant places. An alternative is using WLAN devices. A good reason is that most of the time we stay indoors and it is not often to remove or replace a WLAN device, they are perfect symbols to indicate different places. And the MDC data collector periodically scans available WiFi access points. Therefore, there would have many access points in one significant place. Algorithm 1 shows how to find a user's significant places given all access points' MAC addresses.

Algorithm 1 Find Significant Places

Initialize L to be an empty list
for each timestamp in the raw input file **do**
  Add all scanned MAC addresses into new set S_t
  $Count(S_t) \leftarrow size(S_t)$
  for each set S_i in L **do**
    //Merge S_t and S_i if the overlap exceeds threshold
    if $size(S_t \cap S_i) \geq Threshold1$ **then**
      $S_t \leftarrow S_t \cup S_i$
      $Count(S_t) \leftarrow Count(S_t) + Count(S_i)$
      Delete S_i from list L
    end if
  end for
  Add set S_t into list L
end for
for each set S_i in list L **do**
  if $Count(S_i) \geq Threshold2$ **then**
    Mark MAC addresses in S_i as significant place i
  else
    Mark MAC addresses in S_i as insignificant place
  end if
end for

For each timestamp, all the MAC addresses scanned by the phone are used to identify a unique place or area. Then the algorithm compares this unique place with all other saved places, if there are enough overlap of MAC addresses, which is defined by a threshold, these two sets of MAC addresses are combined and marked as the same place or area. At the same time, the MAC addresses of a place that seen by the phone at this timestamp is counted. Since one MAC address can be scanned multiple times by the phone, the counting is not the size of the MAC address set of a place but an indicator how often the user goes to that place or area. This counting is used to filter insignificant places.

Two thresholds are used to control how many significant places we need to find. The first threshold $Threshold1$ is the number of overlapped WLAN devices between two scanning timestamps. It is used to control how to cluster different WLAN devices. The default value in this paper is set to 3. Different configurations of this threshold does not change the number of significant places that the variation is within 2 places. The second threshold $Threshold2$ is used to filter all insignificant places. We have set this threshold to 600 that if the total count of WLAN devices seen by the phone within one year is less than 600, then it should not be an significant place. This threshold can dramatically affect the number of significant places. By setting these two thresholds with default values, our results have shown that the number of significant places for most users is between 5 to 11 (28 out of 38 users). Table 1 shows the number of significant places that computed by Algorithm 1 with different configuration of two thresholds for all 38 users. As expected, the number of significant places increases if set $Threshold2$ to be a smaller value.

The Latest Used App

The latest used App is an important hint to the next App that will be using by the phone user. The idea behind this is that there is always correlation between two sequential used Apps. A user who has received a text message tends to make a phone call or reply the text message. Another example is that it is more likely to open Google Map for navigation after the user checked Yelp reviews. We trace user's App usage history, then build up a first-order Markov chain of the App usage. The transition function of an App is learned by the statistic of frequency that an App followed to be a foreground App. During the learning procedure, we need to ignore two processes that one is the idle process (UID 101fd64c) and the other is the screensaver process (UID 100056cf). There are few reasons for doing this. First, these two processes are not launched by the user that we do not need to predict. Second, these two processes dominate nearly half of the total App usage. Last, a very strong correlation between these two has been observed and have less correlation to other Apps.

When building up the first-order Markov chain of the App usage, App usage session is introduced in the analysis. Each App usage session is divided by the screensaver and/or idle process. Even though we do not know the real functionality of these two processes, by analyzing the dataset, we find that there is always a large time interval before or after the launch of these two processes. The strong correlation between these two supports our analysis. Hence, we use these two processes as markers of the starting and ending of App usage sessions. Then the transition function of App usage is learned according to each individual App usage session. The advantage of doing this is that the last used App few hours ago will not falsely consider to be correlated to the next App that launched in the next session. The App correlation in the same session is meaningful for us and we only focus on this kind of correlations in this paper. Although sometimes the inter-session correlation is also meaningful.

User Profile Configuration

The last useful context is the user profile. The configuration of user profile reveals user's mood or some special customization of the phone. People may argue that the profile is dependent on time and location as well. We cannot fully agree with that. Indeed, phone users change their profile at a special time or in a special place, but people seldom configure the profile very often. It is usually to be set for some special needs but not special time or location. By setting the phone to be a silent mode, for example, it indicates that the user do not want to be interrupted. It could be set at any time in any place which only depends on user's requirements. Hence, by setting to a special user profile, only a small subset of Apps would be using with this profile configuration. The user profile context is just used to capture these details of the App usage.

The probability of App usage under a certain user profile configuration is learned during the training process. One may argue that most of people just configure the phone to be a general mode at most of time, and this user profile context may have little contribution to the prediction. It is true that users do not change the phone mode very often, but we are introducing this context not on the aspect of what Apps the user is going to launch but taking the advantage of what Apps the user will not use. For instance, when the phone is set to the flight mode, it is not possible for the user to make a phone call or send any text message. For most of the time, this context may cannot increase the accuracy of the prediction, but it can help us to filter the Apps to be launched next. This is the huge difference between the profile context and other contextual information.

Prediction Models

Base on the context described in above sections, we can learn the App usage pattern and then predict the App that will be using given the context. The prediction can be made by integrating the influences of all contextual information in a Bayesian network that shown in Figure 1. In this model, the location (L) is correlated to the day of week (D) and the hour of day (H). Since the last used App (C) is also depend on the time, location and etc., in order to simplify our model, the correlation between two sequentially used Apps is calculated separately by the Conditional Probability Distribution (CPD) $P(A \mid C)$. And the user profile (U) as a context is independent to other random variables. Having these assumptions, we can define the prediction score as a product of the correlation between Apps and the CPDs of an App given time, location and profile configuration.

As shown in Equation 1, we first calculate each App's prediction score given the context by multiplying the CPDs of

Table 1. Number of Significant Places with Different Threshold Configurations

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
`T1=2,T2=600`	9	5	11	10	6	5	11	8	11	11	11	9	9	21	3	5	5	7	6
`T1=2,T2=300`	11	8	19	21	8	8	16	11	17	15	16	22	11	42	4	7	8	13	13
`T1=3,T2=600`	10	5	10	10	7	5	11	8	11	11	11	9	10	21	2	5	5	6	6
`T1=3,T2=300`	12	8	20	21	9	7	16	12	19	15	16	22	12	44	4	7	8	12	13

	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
`T1=2,T2=600`	5	8	11	8	3	4	10	3	9	10	5	4	9	7	3	5	20	4	4
`T1=2,T2=300`	12	14	18	10	5	12	12	4	16	15	9	7	12	8	6	13	25	9	11
`T1=3,T2=600`	5	8	12	8	4	4	10	3	10	11	5	4	8	7	4	5	20	5	4
`T1=3,T2=300`	11	13	20	10	6	10	12	4	17	16	7	7	12	8	7	13	25	9	11

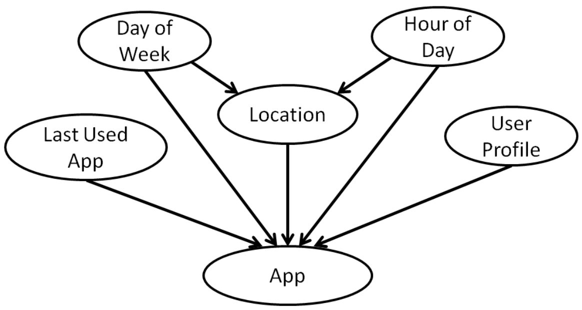

Figure 1. The Influence Bayesian Network for App Usage Prediction

the App and each context. Then according to how many Apps we need to predict, the top scored Apps are the final predictions.

$$Score(A) = P(A \mid C) \cdot P(A \mid D, H, L) \cdot P(A \mid U) \quad (1)$$

Another approach to calculate the prediction score is using a linear model. We first calculate the three CPDs individually, then for each training entity we can use these three CPDs as input and mark the score of the real used App with value 1, while adding auxiliary entities for each other Apps with score value -1. The coefficient β_i for each CPD can be learn by applying least square linear regression. During the testing process, the prediction score of each App can be calculated by Equation 2. And the top scored Apps are predictions as well.

$$Score(A) = \beta_0 + \beta_1 P(A \mid C) + \beta_2 P(A \mid D, H, L) + \beta_3 P(A \mid U) \quad (2)$$

We will show the detailed results of these two models in the next section.

EVALUATION

In this section, we conduct evaluations to verify the effectiveness of our prediction models. The 10-fold cross-validation approach is applied to the MDC dataset of each user for learning and testing.

Process Raw Data

Since our prediction model dedicates to learn the mobile user's App usage pattern and predict App usage given the context, we apply the 10-fold cross-validate approach on each user of the MDC dataset. The raw data cannot be used by the model directly, the context information need to be extracted. Instead of learning all significant places from each training fold, we learn them by using user's whole dataset. The idea behind this is that when we apply 10-fold cross-validation, 9 slices of the dataset should be enough to learn all significant places. In addition, this approach can reduce the computation dramatically. Then we combine the App usage history with user's significant places, and user profile configuration according to the timestamps to generate a new data file. This new data file is then used for the 10-fold cross-validation.

Benchmark

Before we evaluate the effectiveness of the prediction model, a benchmark of the prediction is computed by a statistic of the frequency that the Apps have been used based on the whole dataset. The benchmark of predicting N Apps is the accumulative frequency of the top N most used Apps.

Prediction Results

There are three individual conditional probabilities during the training procedure, we can treat them as three different kinds of influence to the prediction. The combined influences are then used by two models described in Section Prediction Models. Figure 2 shows the prediction results of one user. In this figure, the prediction hit-rates of three conditional probabilities and the results of two prediction models are presented. During the corss-validation, if the real used App is within the predicted Apps, we count it as a hit. The hit-rate defines the effectiveness of the prediction model. We can see that the hit-rates of both prediction models exceed the benchmark. Among three influences, the correlation between Apps has the highest hit-rate that it even exceeds the combination of three. While the influence of time and location has the lowest hit-rate.

Since different context has different contribution and holds different effectiveness to the final prediction, the linear model introduced weights for each context and thus it can coordinate the contributions between them. Within our expectation, the hit-rate of the linear model is better than the Bayesian model which treats these contributions equally. A surprising result is that the influence of correlation between Apps has the highest hit-rate. We can infer from the results that the correlation between Apps have a very strong contribution

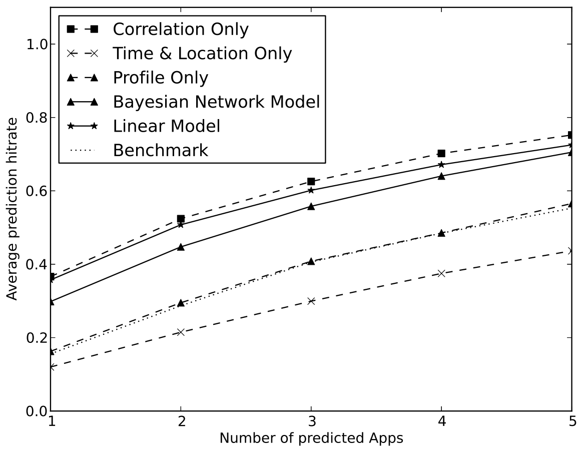

Figure 2. Hit-rate of Predictions Based on Separate Influences for One User

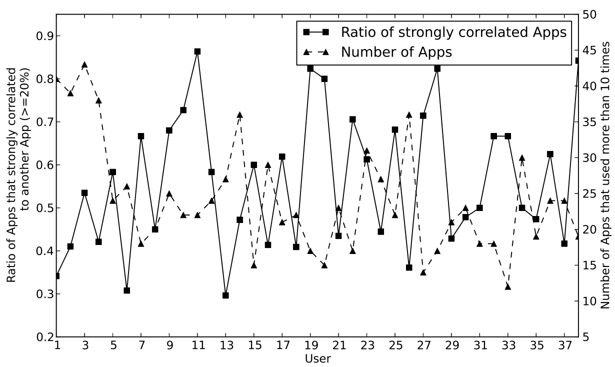

Figure 3. Apps that Strongly Correlated to Other Apps with a Correlation Rate Higher Than 20%

to the prediction. Figure 3 shows that about 50% of Apps, which have been used more than 10 times, are strongly correlated to another App with a correlation rate higher than 20%. We can find a lot of evidences from the dataset. For example, text message editor process is strongly correlated to the text message viewer process or phone call process. And the email process is strongly correlated to the calendar process. It verifies the dependencies between Apps. And it can also explain why the contribution of App correlation can dominate the prediction. Since we only have the App usage information at the process level, and the content or information presented within the App is not available, a deeper analysis of App correlations is not conducted. For the MDC dataset, it is so strong for the prediction that adding the influence of time and location, and user profile, the prediction hit-rate cannot be improved. Another possible reason is that adding more influence factors makes the model sensitive to the noises which will introduce false predictions and pull down the prediction hit-rate.

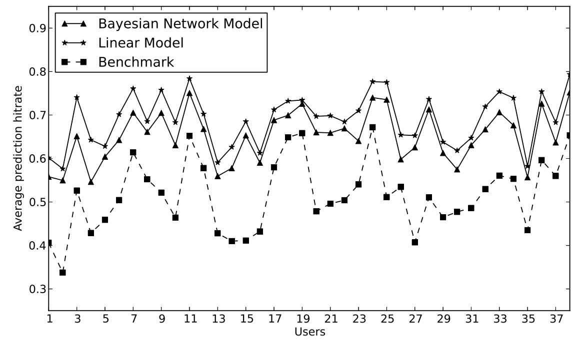

Figure 4. Predicting 3 Apps for All 38 Users

Table 2. Summary Statistic of Hit-rate by Predicting 3 Apps and the Number of Often Used Apps Across All 38 Users

Model	Max	Min	Mean	STD
Bayesian	75.20%	54.63%	65.13%	0.0596
Linear	79.37%	57.64%	69.14%	0.0598
Benchmark	67.20%	33.77%	51.54%	0.0813
Apps	43	12	24.18	7.7011

Effectiveness of Prediction

The results showed in Figure 2 are typical to all 38 users of the MDC dataset. A comparison of results between two prediction models and the benchmark are shown in Figure 4. It shows that the hit-rate of both models are much better than the defined benchmark. Furthermore, the results of the linear model exceeds the Bayesian model a little bit. Table 2 shows a summary hit-rate statistic of the Bayesian Network model, Linear model and the benchmark when predicting 3 Apps across all 38 users. This statistic indicates that the Linear model is about 18% more accuracy than the benchmark in average. And the hit-rate of Bayesian model is nearly 14% higher than the benchmark as well. A statistic of often used Apps (excluding the idle and screensaver processes), which is presented in Figure 3, is also shown in this table. This result presents the effectiveness of two prediction models.

Thus, we can infer from the results that user's App usage pattern can be learn, if there is a pattern, from contextual information and it can be used to predict App usage effectively. A prediction accuracy of nearly 70% in average can be reached by using the Linear model when predicting 3 Apps. A deeper analysis of the influence of these contextual information is conducted in the following sections.

Influence of App Correlation

The hit-rate of predictions based on the App correlation, which is shown in Figure 2, presents a sign that the App correlation makes a major contribution to the final prediction. And the prediction results can support this conclusion. Refering to Figure 3, it clearly shows that users 11, 19, 20, 28 and 38 have a very high ratio of App correlation, all of them have a very high prediction hit-rate in Figure 4 as well.

While users 1, 2, 4, 13 and 26 who have low ratio of App correlation, their prediction hit-rate are comparatively lower than other users. People may argue that the hit-rate has nothing to do with the App correlation but the benchmark that a user who has a high benchmark also has a high prediction hit-rate, and vice versa. Actually it is not the case. Take user 15 for example, this user's benchmark is low but the hit-rate is comparatively high. Another counter-example is user 27 who has a very high benchmark, while the corresponding hit-rate is low. There are also a lot of examples showing that users who have very close benchmark but at different hit-rate levels. Evidences can be found that the benchmark is not correlated to the ratio of inter-App correlation either. Users 18, 19, 24 and 38 have a high benchmark but very different ratios of App correlation. And users 2, 13, 14, 15, 16 and 27 whose benchmarks are low also have different App correlation ratios. In a conclusion, in general the App correlation is positively correlated to the prediction hit-rate, and the benchmark is independent of the ratio of App correlation and the prediction hit-rate.

Another interesting finding is that there is a negative correlation between the App correlation ratio and the number of often used Apps. As it is shown in Figure 3, a user who often uses a smaller number of Apps has a higher ratio of strong correlated Apps, and vice versa. It makes sense that given a same number of correlated App pairs, the ratio is negatively correlated to the number of often used Apps. But it does not mean that a user who uses less number of Apps also presents a higher hit-rate. For instance, both user 3 and user 34 use comparatively a large number of Apps, but the hit-rate of these two users are high as well. While users 10 and 27 who use less Apps have a lower hit-rate. The reason behind this is simple that the absolute number of correlated App pairs contributes to the hit-rate, while the number of often used Apps cannot truly indicate the number of correlated App pairs. This is the collateral evidence that mobile users have certain App usage patterns, and these Apps are correlated with each other to some extent.

Finding that the App correlation contributes the hit-rate significantly does not mean we can ignore other contextual information. A little math work shows that both user 1 and user 28 have about 13 strongly correlated Apps, but the hit-rates of these two users have a big difference. Another example is user 2 and user 38 whose number of strongly correlated Apps is 16 also presents a large gap between the hit-rates. The only reasonable explanation is that other contextual information also makes a significant contribution to the prediction, even though it seems that the App correlation can dominate the hit-rate.

Influence of Significant Places

As it is shown in Table 1, the number of significant places can be dramatically affected by the second threshold. Then how could the prediction hit-rate be influenced by the number of significant places? Figure 5 shows the hit-rate of predicting 3 Apps for all 38 users by fixing $Threshold1$ with value 3 and varying $Threshold2$. The results indicate that the prediction hit-rates are almost identical with three different configurations of $Threshold2$. It could be inferred from the results that the influence of significant places, which calculated by Algorithm 1, to the prediction hit-rate is trivial for the MDC dataset.

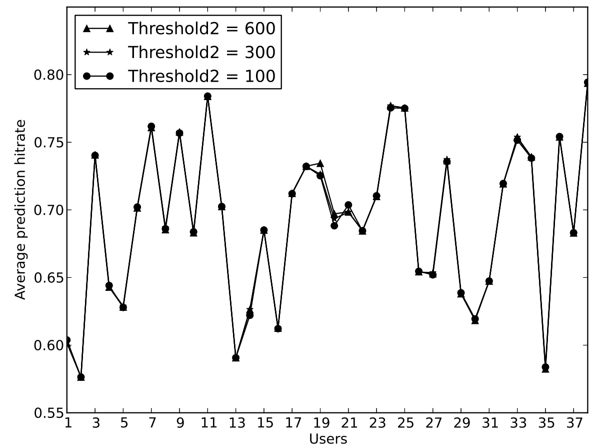

Figure 5. Prediction Hit-rate With Different Threshold Configurations

One explanation to the results is that the location is highly depends on time. Since we have already taken the time contextual information into the consideration in our models, the influence of location context becomes trivial to the final results. In Figure 3, it also shows that most of the users are only using about 25 Apps and the top three most used Apps dominate nearly 45% of the total usage in average which shown in Figure 4. Hence, the correlation between the location and the App usage is not so significant. Another possible reason could be that Algorithm 1 is too simple to infer user's significant places by only using the MAC addresses that scanned by the phone.

Contribution Weights in Linear Model

Different users have different mobile App usage patterns. The contribution of different contextual information may varies between different users as well. In the Linear model, there are four coefficients in Equation 2. And Figure 6 represents all 38 users' mean contribution weights of 10 folds after training. The weights of App correlation (β_1) and time-location context (β_2) are positive, and the weight of the user profile context (β_3) is negative for all users. It verifies our analysis that both the App correlation and time-location context contribute to what Apps users are more likely to use, while the user profile context contributes to what Apps users are not likely to use.

For all 38 users, the weight of App correlation has mean 2.014, and the standard deviation is 0.1433. It proves that the App correlation commonly exists within all 38 users. The mean weight of time-location context is 2.956 but the standard deviation is 0.997. This reflects the difference of usage patterns between users. And it is the most unpredictable as-

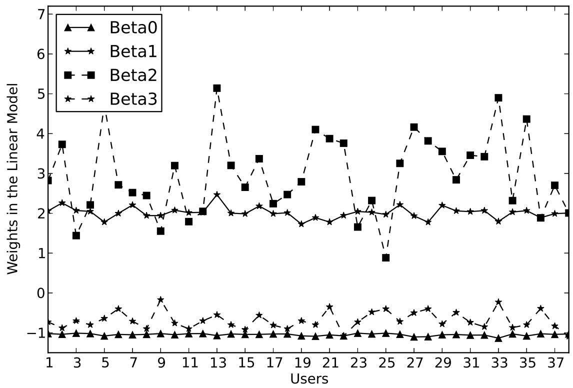

Figure 6. Weights in the Linear Model of All 38 Users

pect when predicting App usage. It also explains why combining all contextual information produces a lower hit-rate than only using the App correlation. Contrast to these two context, the weight of user profile context negatively influences the prediction. We can also see a small diversity, that its standard deviation is 0.217 and the mean is -0.686, between 38 users.

CONCLUSIONS

We have presented our exploration of how to use the contextual information, such as time, location, user profile, and latest used App, to predict the mobile App usage. Focusing on the domain of context awareness for App usage prediction, we found that context can really affect the behavior of user's App usage. And by analyzing contextual information, we can learn the pattern, to some degree, of mobile user's App usage. This work suggests new opportunities for research in personal mobile system that leverage contextual information to dynamically represent information, like Apps to be using, to user and improve the interaction experience between user and mobile phone.

ACKNOWLEDGEMENT

The work described in this paper was conducted by using the Nokia MDC (2012) dataset. We extend our thanks to Nokia and especially the MDC research team for making available their mobile user study dataset.

REFERENCES

1. D. Ashbrook and T. Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, 7(5):275–286, Oct. 2003.

2. A. Dix, T. Rodden, N. Davies, J. Trevor, A. Friday, and K. Palfreyman. Exploiting space and location as a design framework for interactive mobile systems. *ACM Trans. Comput.-Hum. Interact.*, 7(3):285–321, Sept. 2000.

3. T.-M.-T. Do and D. Gatica-Perez. By their apps you shall understand them: mining large-scale patterns of mobile phone usage. In *Proceedings of the 9th International Conference on Mobile and Ubiquitous Multimedia*, MUM '10, pages 27:1–27:10, New York, NY, USA, 2010. ACM.

4. H. W. Gellersen, A. Schmidt, and M. Beigl. Multi-sensor context-awareness in mobile devices and smart artifacts. *Mob. Netw. Appl.*, 7(5):341–351, Oct. 2002.

5. J.-M. Kang, S. seok Seo, and J.-K. Hong. Usage pattern analysis of smartphones. In *Network Operations and Management Symposium (APNOMS), 2011 13th Asia-Pacific*, pages 1 –8, sept. 2011.

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 *Mobile Data Challenge by Nokia Workshop, in conjunction with Int. Conf. on Pervasive Computing*, June 2012.

7. S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. Campbell. Nextplace: A spatio-temporal prediction framework for pervasive systems. In K. Lyons, J. Hightower, and E. Huang, editors, *Pervasive Computing*, volume 6696 of *Lecture Notes in Computer Science*, pages 152–169. Springer Berlin / Heidelberg, 2011.

8. B. Yan and G. Chen. Appjoy: personalized mobile application discovery. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, MobiSys '11, pages 113–126, New York, NY, USA, 2011. ACM.