On Mining Mobile Apps Usage Behavior for Predicting Apps Usage in Smartphones

Zhung-Xun Liao
Department of Computer Science, National Chiao Tung University
HsinChu, Taiwan
zxliao@cs.nctu.edu.tw

Yi-Chin Pan
Search Engineering, Yahoo! Inc.
Taipei, Taiwan
jeanpan@yahoo-inc.com

Wen-Chih Peng
Department of Computer Science, National Chiao Tung University
HsinChu, Taiwan
wcpeng@cs.nctu.edu.tw

Po-Ruey Lei
Department of Electrical Engineering, Chinese Naval Academy
Kaohsiung, Taiwan
kdboy1225@gmail.com

ABSTRACT

Predicting Apps usage has become an important task due to the proliferation of Apps, and the complex of Apps. However, the previous research works utilized a considerable number of different sensors as training data to infer Apps usage. To save the energy consumption for the task of predicting Apps usages, only the temporal information is considered in this paper. We propose a Temporal-based Apps Predictor (abbreviated as TAP) to dynamically predict the Apps which are most likely to be used. First, we extract three Apps usage features, global usage feature, temporal usage feature, and periodical usage feature from the Apps usage trace. Then, based on those explored features, we dynamically derive an Apps usage probability model to estimate the current usage probability of each App in each feature. Finally, we investigate the usage probability in each feature and select k Apps with highest usage probability from the probability model. In this paper, we propose two selection algorithms, MaxProb and MinEntropy. To evaluate the performance of TAP, we use two real mobile Apps usage traces and assess the accuracy and efficiency. The experimental results show that the proposed TAP with the MinEntropy selection algorithm could have shorter response time of Apps prediction. Moreover, the accuracy reaches to 80% when k is 5, and when k is 7, the accuracy achieves almost 100% in both of the two real datasets.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

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 the author(s) 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.
CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2263-8/13/10 ...$15.00.
http://dx.doi.org/10.1145/2505515.2505529.

Keywords

Mobile Apps, Behavior Prediction, Feature Extraction

1. INTRODUCTION

Mobile applications (Apps) have been developed rapidly to satisfy users' requirements with the considerable increase of the number of mobile devices. Users can easily download and install Apps in their smart devices via online Apps providers, such as Android Market (Google Play), Apple App Store, Nokia Ovi Store, Windows Phone Marketplace, and so on. In our collected dataset, the average number of Apps in a user's smartphone is around 53. For some users, the number could even reach to 150. Furthermore, the number is continuously growing up. Concurrently, the functionalities of Apps are becoming more extensive and powerful, so that users use mobile devices not only for calling but also for web browsing, shopping and socializing.

Regarding the complex functionalities and the increasing number of Apps installed in users' smartphones, predicting Apps usage facilitates fast Apps launching, intelligent user experience, and power management in smartphones. The authors in [20] proposed a system, called FALCON, to predict Apps usage so that they could preload those Apps which will be invoked and remedy the launch delay. The authors in [11] proposed an AppNow widget and the authors in [18] proposed a Naive Bayes classification method to predict Apps usage for helping users spend less time on searching Apps in their smartphones. However, all of the previous works collect a bunch of different types of sensor readings, such as, time, GPS, Wi-Fi, Apps usage, and so on. Obviously, the more sensor readings involved, the more energy and storage consumption and even the more noise data.

The authors in [2, 9] investigated the correlation between App usage and the usage time, and proved that time is highly related to the App usage behavior. For example, most users read news in the morning and play games at night [2]. Thus, in this paper, we only focus on predicting App usages via their temporal information to save both energy and storage consumption. The usage trace is formed by a time series data including the name of Apps and their launched time.

In this paper, we propose a Temporal-based Apps Predictor, called TAP, which dynamically lists k Apps which are most likely to be

used. In TAP, three Apps usage features are discovered from the temporal information of the collected usage trace. Then, given a query time and one user-defined threshold k, TAP dynamically calculates the usage probability from each feature, and finally k Apps are selected by assessing the usage probability in each feature.

By analyzing Apps usage traces, for each App, we observed three Apps usage features which are helpful for predicting the Apps usage. Here, different features are able to predict different types of usage behavior. The first feature is called global usage feature which records the global usage count of each App. Thus, Apps' global usage is captured. When we use global usage feature to predict Apps usage, it indicates that a user would more likely use the Apps which were used frequently.

The second feature is called temporal usage feature which records the usage counts regarding to a specific temporal bucket (e.g., at 9:00 a.m.). Therefore, the temporal usage feature models the usage behavior of being used at a specific time. As a result, given a query time q_t, it indicates that a user would more likely use the Apps which are used in the same temporal bucket as q_t. For example, suppose a user used Alarm at every 11:00 p.m., we could infer that when the time is 11:00 p.m in the future, the user will also use Alarm.

The third feature is called periodical usage feature which reveals the usage periods of each App. The Apps which are most likely to be used are those Apps which repeatedly used at query time. Thus, the periodical usage feature indicates that a user would more likely use the Apps whose usage period is matching the query time. For example, suppose the query time is 9:00 a.m. and a user played AngryBirds at 10:00 p.m. last night, if the usage period of AngryBirds is 11 hours for that user, we could infer that (s)he will play AngryBirds at the query time. However, to identify the usage periodicity is not as straightforward as identifying the global and temporal usage features. In this paper, we proposed a periodicity identification method to discover the period of each App.

Given a query time q_t, we can derive the usage probability based on each of the three App usage features for each App. In other words, each App has three usage probabilities derived from three different features respectively. Now, we have to select k Apps by integrating those probabilities. In this paper, we propose two selection algorithms: MaxProb and MinEntropy. The MaxProb iteratively selects one App with highest probability from all three features. The MinEntropy iteratively selects one App with highest probability from the feature with minimum entropy. The feature with lowest entropy is considered to be more discriminative.

To evaluate our proposed Apps usage features and the temporal-based Apps predictor, two real mobile Apps usage datasets are used. The experimental results show that our approach could accurately predict top-k Apps. In addition, the discovered features can well capture their corresponding usage behaviors.

The main contributions of this paper are summarized below.

- We propose a Temporal-based App Predictor (TAP) for Apps usage prediction according to only the temporal information of user's Apps usage behavior explored from the Apps usage history.
- By observing Apps usage traces, we define three Apps usage features which are global usage feature, temporal usage feature, and periodical usage feature.
- We formulate an Apps usage probability model to further dynamically model the usage probability of each App in each feature.
- In light of the Apps usage probability model, we develop two selection algorithms, MaxProb and MinEntropy, to determine top-k Apps with highest usage probability.
- Comprehensive experiments are conducted over two real datasets. The experimental results show that the proposed TAP is able to predict the Apps usage accurately and efficiently.

The remainder of this paper is organized as follows: Section 2 presents some related works. Section 3 describes the preliminaries and the overview of the proposed TAP. In Section 4, we introduce three proposed Apps usage features in detail and also describe the corresponding probability model. The top-k Apps selection algorithms are proposed in Section 5. Section 6 presents the comprehensive experimental results and Section 7 concludes this paper.

2. RELATED WORK

In this section, we introduce the state-of-the-art research works pertaining to predicting Apps usage. Concurrently, we investigate the current mobile data mining tasks and show the possibility of adopting those tasks to assist predicting Apps usage. In addition, we also survey the traditional prediction problems and differentiate the Apps usage prediction from other prediction problems.

2.1 Mobile Applications Usage

There are some research works analysing the usage of mobile Apps. In [9], the authors presented a probabilistic framework to mine usage patterns of mobile Apps by considering the temporal information in days. However, the temporal information is only quantized into 4 intervals: morning (6am-12am), afternoon (12am-6pm), evening (6pm-0am), and night (0am-6am). Besides, the number of usage is roughly categorized into no-use (0 times), low-use (1-2 times), middle-use (3-4 times), and high-use (more than 4 times). Such mechanism is heuristic and without rationales. The authors in [11] designed an Android widget to show k Apps which are most likely to be launched by constructing the temporal profile for each App. However, the temporal profile is based on the usage period of each App. Once the usage period does not exist, the prediction would be failed. In [20], the authors predicted the Apps usage by not only the temporal information but three kinds of mobile usage logs: locations, temporal bursts, and context signals. However, to collect accurate location information is energy-consuming. We need to turn GPS or Wi-Fi module on, but the satellite and Wi-Fi signals do not always exist. Besides, the location information from GSM signal is quite inaccurate. The inaccurate location information could be noisy. In [18], the authors investigated all possible sensors attached in a smartphone and adopted a Naive Bayes classification to predict the Apps usage.

However, collecting all possible sensors is inefficient and impractical. Moreover, the useful sensors for different users could be different according to their usage behavior. In this paper, we collect only the Apps usage traces for saving energy consumption, instead of logging every mobile usage information. Concurrently, from the temporal information, we extract three Apps usage features to enrich the usage information, and predict Apps usage by the three Apps usage features. As the different data input between previous works and this paper, we cannot compare with the previous works in our experimental study.

2.2 Mobile Data Mining

In addition to the mobile Apps usage, our daily locations, communication history, and movement trajectories are able to be captured with the popularity of mobile devices and smartphones. These

mobile data with users behavior and social interaction have attracted many researchers to pay attention on mining interesting and useful knowledge [13, 3, 12, 22, 21, 6, 23]. [13] uses an apriori-like approach to mine the associations between users interactions and contexts from mobile context logs, and further characterizes the habits of mobile users. [3] discovers the habits of mobile users to find out who are the similar mobile users. In [12, 22, 21], the authors focused on the mobile search problem which allows users to search, locate, and access web information on mobile devices. In addition, [23] uses a collaborative filtering technique to recommend locations where the mobile user might be interested, and [17] recommends Apps to users according to their usage patterns. Besides, the authors in [4] studied the privacy and risk security of mobile Apps. To analyze the smartphone usage behaviors, the authors in [8] presented two contextual variables, places and social context, and observed the relations between the smartphone usage and the two contextual variables.

2.3 Prediction Problems

The prediction problem has been studied extensively in recent years. There are many emerging studies and applications, including the buying trends, supply demands, future events, social influence, and location prediction [10, 14, 5, 12, 1, 7]. For location prediction [10, 14], Jeung et al. [10] explored the association rules to represent the trajectory patterns and proposed a hybrid prediction model that combined the trajectory patterns with a motion function to predict the future location of moving objects. [14] uses trajectory patterns, which are extracted from the previous movement patterns, to train and built a T-pattern Tree for predicting the next location. [12] presents a data-drive method to analyze the local search queries and then proposes location-aware features for local search click prediction. [1] proposes a hybrid method based on the probabilistic and the time-series model to predict the future events. [7] focuses on predicting social influence by formulating the prediction problem as a user-post matrix and proposing a hybrid algorithm, Hybrid Factor Non-negative Matrix Factorization (HF-NMF), to solve the problem.

For the mobile Apps prediction problem, we can not use the existing prediction methods, such as classification and regression, to solve the Apps usage prediction problem, since the Apps usage behavior of a user are different with time, and the features should be determined to consider individually or collectively at the query time. Therefore, we propose a temporal-based Apps prediction framework which discovers a user's Apps usage behavior by considering multiple features and utilize these features to predict which Apps are most likely to be used by the user at a given query time. In summary, the main theme of this paper is to extract Apps usage features and combine the usage probability derived by these three features to have the top-k Apps prediction.

3. OVERVIEW AND PRELIMINARY

Figure 1 shows the overall architecture of the proposed TAP which is composed of an off-line features extraction and an on-line Apps usage prediction. In the off-line features extraction, we extract three temporal-based features from the collected Apps usage trace which is defined in Definition 1. Then, the discovered temporal-based features are used in the on-line component to predict the Apps usage. Each feature contributes different prediction ability of different types of usage behavior. We will illustrate these three features in Section 4. In the on-line Apps usage prediction, given a query time, we dynamically build an Apps usage probability model to describe the usage probability of each App in each feature. For example, Figure 2 shows the Apps usage probability

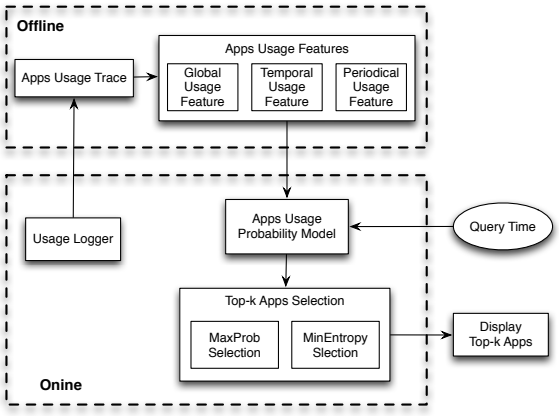


Figure 1: An overview of Temporal-based Apps Predictor (TAP).

	Global	Temporal	Periodical
Gmail	0.27	0.12	0.24
Facebook	0.08	0.10	0.02
Alarm	0.07	0.18	0.02
Google+	0.06	0.17	0.15
Calendar	0.25	0.16	0.23
Line	0.25	0.17	0.16
Angry bird	0.02	0.10	0.18

Figure 2: An example of Apps usage probability model for seven Apps.

model of a user who used seven Apps in the usage trace. Each App has one usage probability in each feature, where the summation of the probabilities in each column is 1. We also show the definition of Apps usage probability model in definition 2. The Apps usage probability model abstracts the usage trace and could be placed on a local mobile device.

DEFINITION 1. **Apps usage trace:** *An Apps usage trace is a collection of tuples, (app, t), where app is a mobile application used at time t.*

DEFINITION 2. **Apps usage probability model:** *An Apps usage probability model is an N by M matrix, where N is the number of Apps and M is the number of features (i.e., $M = 3$, in this paper). In the matrix, each value describes the usage probability for each App estimated in each feature.*

Finally, two selection algorithms are introduced to select k Apps with highest probability in the Apps usage probability model. However, Apps usage probabilities are diverse in different features. The challenge is how to integrate the usage probability from all of the features and identify which feature is more confident to model the user's behavior. For example, as can be seen in Figure 2, Alarm and AngryBirds have the same probability, 0.18, in temporal usage feature and periodical usage feature, respectively. It is hard to tell which one is more likely to be used before we recognize the importance of each feature. To solve this problem, we propose two selection algorithms, MaxProb and MinEntropy, in this paper. The MaxProb selection measures the importance of features by the

Table 1: Notation used in this paper.

Symbol	Description
APP	the set of Apps
f_g	global usage feature
f_t	temporal usage feature
f_p	periodical usage feature
$P_{f_i}(app)$	the probability of app in feature f_i
$u_g(\cdot)$	usage count for f_g
$u_{t_i}(\cdot)$	usage count for the i-th temporal bucket of f_t
$u_{p_i}(\cdot)$	usage count for the i-th period bucket of f_p

Apps usage probability directly. By contrast, the MinEntropy selection views the importance of features by their Entropy. Here, we could have a formal definition of top-k Apps usage prediction as shown in Definition 3.

DEFINITION 3. **Top-k Apps usage prediction:** *Given the Apps usage trace of a user, a query time q_t, and a user-defined threshold k, the top-k Apps usage prediction is to list k Apps which are most likely to be used by the user at the query time, q_t.*

4. APPS USAGE FEATURES AND PROBABILITY MODEL

In this section, we state the property and characteristics of three Apps usage features: global usage feature, temporal usage feature, and periodical usage feature. Each feature captures different usage behavior of Apps. In addition, we describe how to build the Apps usage probability model when a query time is given. Table 1 lists the symbols used in this paper.

4.1 Global Usage Feature

The global usage feature is denoted as $f_g(app) =< u_g(app) >$, where $u_g(app)$ is app's usage count in the entire usage trace. When we use global usage feature to predict Apps, the usage probability of app in global usage feature is formulated as in Equation 1. Obviously, the global usage feature is useful for capturing the usage behavior of the Apps who are used frequently in the global view. The representative Apps are IM Apps and social networks Apps. For example, Figure 3 shows the usage history of Faceobook for one random user in the collected dataset. The usage is quit random and almost occurred in every hour.

$$P_{f_g}(app) = \frac{u_g(app)}{\sum_{app_i \in APP} u_g(app_i)} \quad (1)$$

4.2 Temporal Usage Feature

For each mobile App app, the feature of temporal usage is denoted as $f_t(app) =< u_{t_1}(app), u_{t_2}(app), \ldots, u_{t_{24}}(app) >$, where each element is mapped to one temporal buckets. For example, t_1 is the temporal bucket of 0:00 to 0:59, and t_2 is 1:00 to 1:59. Thus, $u_{t_1}(app)$ is the average number of usage in the temporal bucket t_1. In general, $u_{t_j}(app)$ could be calculated by Equation 2, where m is the total number of days in the usage trace, and $u^m_{t_j}(app)$ is the number of usage in the j-th temporal bucket of the m-th day.

$$u_{t_j}(app) = \frac{\sum_1^m u^{(m)}_{t_j}(app)}{m} \quad (2)$$

The corresponding probability model is depended on which temporal bucket the query time belongs to. Suppose the query time is belonging to the temporal bucket t_q, we thus have the temporal usage probability of app as in Equation 3, which is usage probability of app being used in time t_q. The temporal usage feature is useful for those Apps who are regularly used at a specific time. Those Apps, such as Alarm setting and calendar, could be considered as having this kind of usage behavor. Figure 4 shows that the user usually set alarm at around 23:00. Therefore, it would have a higher usage probability in temporal bucket t_24 than in other temporal buckets.

$$P_{f_t}(app) = \frac{u_{t_q}(app)}{\sum_{app_i \in APP} u_{t_q}(app_i)} \quad (3)$$

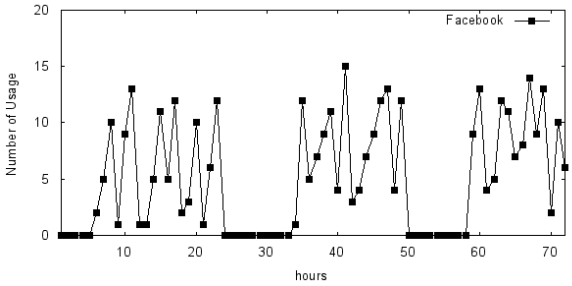


Figure 3: Example of Apps with frequent usage: usage history of Facebook.

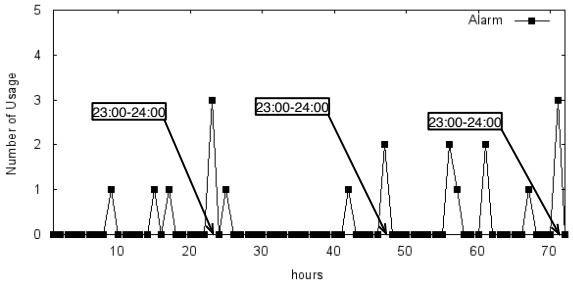


Figure 4: Example of Apps with specific usage time: usage history of Alarm setting.

4.3 Periodical Usage Feature

Finally, we introduce the periodical usage feature which captures the periodicity of each App. The periodical usage feature is denoted as $f_p(app) =< u_{p_1}(app), u_{p_2}(app), \ldots, u_{p_{\tau_{app}}}(app) >$, where τ_{app} is the period of app, to denote the periodical usage feature. For example, Figure 5 shows the usage of Gmail for a random user in our dataset. As can be seen in Figure 5, the user checks email around every 3 hours. Thus, we could have $\tau_{Gmail} = 3$ for the Gmail usage behavior of the user in this example. In addition, the probability model of app when the query time falls into the q-th bucket of its period (i.e., p_q) is listed in Equation 4, which is similar to the usage probability model of temporal usage feature.

$$P_{f_p}(app) = \frac{u_{p_q}(app)}{\sum_{app_i \in APP} u_{p_q}(app_i)} \quad (4)$$

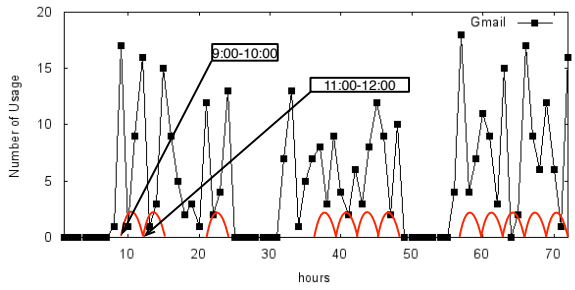


Figure 5: Example of Apps with significant period: usage history of Gmail.

The periodical usage feature is useful for those Apps who have significant period. However, to identify the significance of an App's period is challenging. Especially, the period is usually implicit. In this paper, we proposed an effective period detection method which can find the most significant period for each App.

4.3.1 Period Detection

To identify the usage period of an App, we apply Discrete Fourier Transformation (DFT) to transform the usage history in time domain to frequency domain. Then, a dynamic cut strategy [19] is adopted to filter out all the insignificant periods. The dynamic cut approach randomly shuffles the App's original usage history and applies DFT on the shuffled usage history. Consequently, the frequency with maximum power in the periodogram is considered as the significance threshold. Since the shuffled usage history is viewed as no significant period within, if the discovered frequency from the original usage history has lower power than the significant threshold, we believe that it is insignificant. Here, if there is no any significant period, we will only use the period with maximum power. Finally, the autocorrelation is conducted to prevent the problem that DFT cannot find the significant period in the low frequency region.

Figure 6(a) shows an example of an App usage histogram for 4 weeks, and Figure 6(b) is the periodogram after DFT. The red dashed line is derived by dynamic cut approach. Then, only those frequencies with higher power than the red dashed line are marked as the App's usage frequencies. In Figure 6(b), this App is recognized as having significant period and the frequencies are P_2 and P_3. Finally, in Figure 6(c), the frequency is mapped to the period on the autocorrelation curve, and we can see that the mapped significant period is corresponding to 20 hours which is marked as P_2 and 15 hours which is marked as P_3.

The time complexity of DFT is $O(NlogN)$, where N is the length of time series data. Since autocorrelation is a formal convolution which can also be solved by FFT, its complexity is $O(NlogN)$ as well. Thus, the overall time complexity of period detection is $O(NlogN)$. It is acceptable for executing in a smart device.

5. TOP-K APPS SELECTION

As we have all the usage probability lists from different features, to integrate the usage probability lists and output one single list is a challenge. Given an Apps usage probability model, the goal of top-k Apps usage selection is to list k Apps with highest probability over the Apps usage probability model. Those Apps are considered most likely to be used regarding the Apps usage probability model. We propose two selection algorithms, MaxProb and MinEntropy, to deal with this problem by different selection strategies.

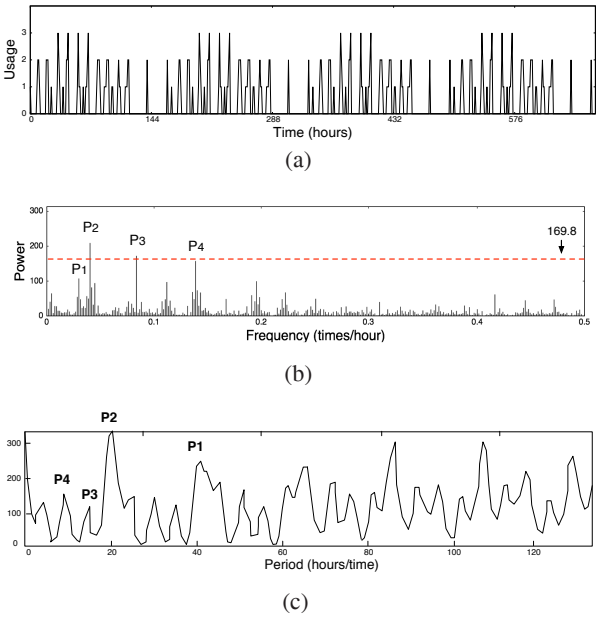


Figure 6: An example of periodicity detection

5.1 MaxProb Selection

The MaxProb method ignores the characteristics of different features. For each App, it uses the maximum probability over three features as the usage probability of the App. Thus, the k Apps with highest maximum usage probability is formed as the top-k Apps. In each iteration, the MaxProb method selects one App with the highest maximum probability, which satisfies Equation 5, where APP is the set of unselected Apps and f_i is the i-th feature. Intuitively, after k iterations, we could terminate the selection with k selected Apps.

Algorithm 1 shows the detailed procedure of the MaxProb selection. For example, Figure 7 shows the process of selecting top-k ($k = 6$) Apps by the MaxProb selection. The result of k Apps is <Gmail, Calendar, Line, Alarm, Angry bird, Google+>.

$$SelectedApp = \underset{app \in APP}{\operatorname{argmax}} \max_{i} P_{f_i}(app) \tag{5}$$

Algorithm 1: MaxProb Selection

Input: Apps usage probability model: M, number of prediction: k
Output: A top-k Apps list: L

```
Define L = ∅;
while |L| < k do
    app ← GetHighestProbability(M, APP) ;
    Append app to L ;
    APP ← APP − {app} ;
end
return L
```

Ignoring the property of features could generate two problems: 1) cross-feature comparison and 2) dominated by single feature. First, the maximum probability of an App is drawn from comparing the probabilities cross all features for the App. The comparison of the probabilities in different features could be meaningless. For

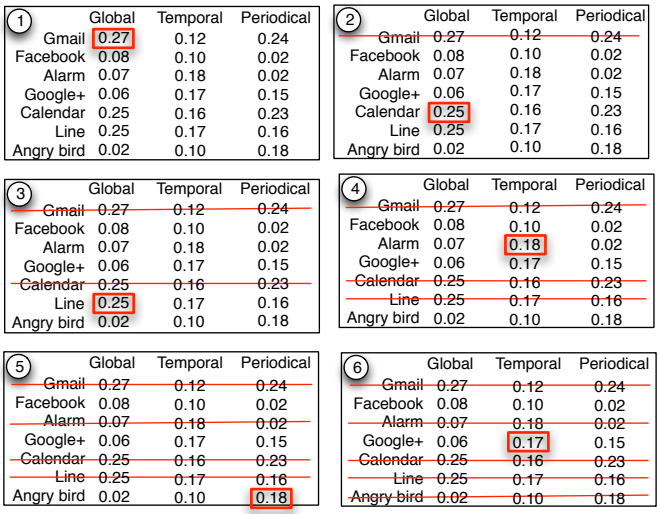


Figure 7: An example of MaxProb selection for top-k Apps.

example, the probability of 0.9 in the global usage feature and the probability of 0.5 in the temporal usage feature are not drawn from the same base of source. Second, we observed the result of the MaxProb method could be dominated by a single feature. From Figure 7, the first three Apps are selected from global usage feature. As described in Section 4, each feature handles a particular type of usage behavior. Therefore, the result of MaxProb selection would miss the Apps with other types of usage behavior. Concurrently, the result could be biased by the dominated feature.

Therefore, we propose an entropy-based method, called MinEntropy selection, which iteratively selects one App with the highest probability in the feature with minimum entropy. Theoretically, the feature with minimum entropy is considered as with lowest uncertainty. Therefore, we can guarantee the selected Apps are from a more discriminative feature which is easier to predict.

5.2 MinEntropy Selection

The MinEntropy selection focuses on not only the usage probability of Apps but also the entropy of each feature. Since entropy measures the uncertainty of a random variable [16]. When a random variable is of less uncertainty, it's entropy value will be smaller. As a result, the concept of entropy implies the degree of certainty and can be adopted to improve the prediction accuracy. In this paper, when the entropy of a feature is smaller, we argue that the feature is more helpful for Apps usage prediction. To calculate the entropy value of feature f_i, we apply Equation 6, where $\sum_{app\ in APP} P_{f_i}(app) = 1$. For example, the entropy value of the global usage feature shown in Figure 2 is $-(0.27 \times \log 0.27 + 0.08 \times \log 0.08 + 0.07 \times \log 0.07 + 0.06 \times \log 0.06 + 0.25 \times \log 0.25 + 0.25 \times \log 0.25 + 0.02 \times \log 0.02) = 0.74$. Thus, in each iteration, we select the App with maximum probability from the feature with minimum entropy.

However, when one App is selected, we have to normalize the probabilities of unselected Apps and recalculate the entropy for next iteration. For example, Figure 8 shows a running example with the prediction result of <Gmail, Calendar, Line, Angry bird, Google+, Alarm>. After the first iteration, since Gmail is selected, the probabilities for the remaining Apps should be normalized such that the summation of their probabilities would equal to 1. Then, the entropy is recalculated according to the normalized probability. To avoid the normalization of each probability for updating the entropy, we propose an incremental entropy update scheme. It only needs the entropy in the previous iteration and the probability of selected App, which is shown as in Equation 7, where $H_{f_i}^{(k)}$ is the entropy of feature f_i in the k-th iteration, $\hat{p}$ is the probability of the selected App. Algorithm 2 shows the minimum entropy selection, where we only need k iterations to collect top-k Apps.

$$H_{f_i} = -\sum_{app \in APP} P_{f_i}(app) \log P_{f_i}(app) \tag{6}$$

$$\begin{aligned}
H_{f_i}^{(k)} &= -\sum_{app \in APP} \frac{P_{f_i}(app)}{1-\hat{p}} \log \frac{P_{f_i}(app)}{1-\hat{p}} \\
&= -\sum_{app \in APP} \frac{P_{f_i}(app)}{1-\hat{p}} (\log P_{f_i}(app) - \log(1-\hat{p})) \\
&= -\frac{1}{1-\hat{p}} \sum_{app \in APP} P_{f_i}(app)(\log P_{f_i}(app) - \log(1-\hat{p})) \\
&= -\frac{\sum_{app \in APP} P_{f_i}(app) \log P_{f_i}(app)}{1-\hat{p}} \\
&\quad + \frac{\log(1-\hat{p}) \sum_i P_{f_i}(app)}{1-\hat{p}} \\
&= \frac{H_{f_i}^{(k-1)} + \hat{p} \log \hat{p}}{1-\hat{p}} + \frac{(1-\hat{p}) \log(1-\hat{p})}{1-\hat{p}} \\
&= \frac{H_{f_i}^{(k-1)} + \hat{p} \log \hat{p}}{1-\hat{p}} + \log(1-\hat{p})
\end{aligned} \tag{7}$$

Algorithm 2: MinEntropy Selection

Input: Apps usage probability model: M, number of prediction: k
Output: A top-k Apps list: L

```
Define L = ∅;
while |L| < k do
    f_i ← GetLowestEntropy(M)
    app ← GetHighestProbability(f_i, APP) ;
    Append app to L ;
    APP ← APP − {app} ;
end
return L
```

6. EXPERIMENTS

In this section, we evaluate the performance, including accuracy and efficiency, of the proposed prediction methods and two baseline methods. We also test the impact of the discovered three Apps usage features to show their prediction ability. Finally, the parameters analysis is delivered. All algorithms are implemented in Java and executed on a GNU/Linux PC with an Intel Xeon Core 4 CPU (2.66GHz) and 8 GB memory.

6.1 Experimental Environment

First of all, we introduce the environment of our experiments, including the characteristics of two datasets we used, the measurements to evaluate the prediction accuracy, and the methods we implemented to compare with TAP.

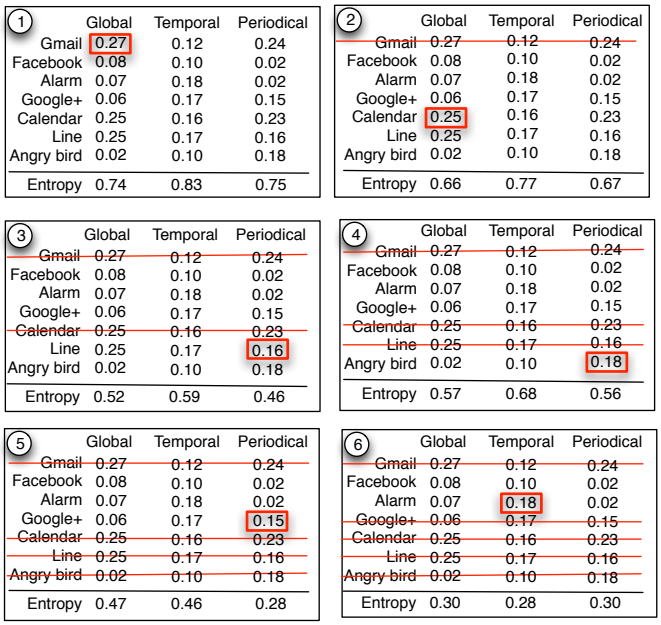


Figure 8: An example of MinEntropy selection for predicting top-k Apps.

Table 3: Statistics information of our real dataset

Abbreviation	Method
Freq	Frequency-based method
MRU	Most recently used method
TAP-MaxProb	TAP method with MaxProb selection
TAP-MinEntropy	TAP method with MinEntropy selection

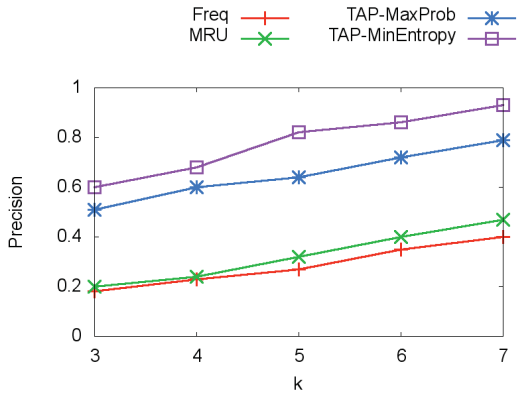


Figure 9: Prediction recall comparison on the small dataset.

Table 2: Statistics information of the real datasets

	Small	Large
Average usage count of a user	3492	13717
Average number of Apps of a user	53	41
Average daily usage count of a user	21	70

6.1.1 *Dataset Description*

We conduct extensive experiments on two real datasets from a mobile phone company. We implement one monitoring program on the Android 2.2 platform to record the usage trace and stores it in the local storage. For the small dataset, we collected 15 participants' usage traces from July to December in 2010. For the large dataset, the data was collected by 80 participants from October 2009 to February 2011. Each dataset is separated as 60% for training and 40% for testing. Table 2 shows the statistical information of those two datasets.

6.1.2 *Accuracy Measurement*

To evaluate the accuracy and effectiveness of our proposed method, we use $recall$ and Mean Reciprocal Rank, MRR [15], as our evaluation measures. Here, $recall$ calculates only the hit ratio of the top-k Apps without considering the order of the clicked App. As long as the user clicks anyone of the top-k Apps, we count it as one hit. By contrast, MRR score is sensitive to the order of the top-k Apps. Therefore, we could use MRR score to measure the effectiveness of the prediction. Equation 8 shows how to calculate MRR score, where Q is the number of query and $rank_i$ is the rank of the clicked App by the i-th query. Note that if the user did not click any Apps in the top-k list, we will set $rank_i = \infty$ and thus, $\frac{1}{rank_i} = 0$. According to the characteristics of MRR score, if a user always uses the Apps in the prediction list (i.e. $recall = 1$), the MRR score will be larger than or equal to $\frac{1}{k}$.

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{8}$$

6.1.3 *Compared Methods*

Although in Section 2, we describe three existing works which predict Apps by collecting most mobile sensors' readings, the input data are very different from ours in this paper. We only have temporal information through the entire proposed prediction process. Therefore, we also conduct 2 most famous and well-known methods, Freq and MRU, as the baseline methods, which are listed in Table 3. The Freq method ignores the query time and selects k most frequently used Apps as the result. The MRU method returns k most recently used Apps regarding the query time. Note that the MRU method is currently used in most mobile OS including Android and iOS. For the proposed TAP method, we also evaluate the performance of different selection algorithms, MaxProb and MinEntropy, respectively.

6.2 Experimental Results

In this paper, we conduct 4 experiments: 1) accuracy evaluation, 2) efficiency evaluation, 3) impact of Apps usage features, and 4) parameter studies.

6.2.1 *Accuracy Comparison*

We first show the experimental results for the small dataset. Figure 9 depicts the $recall$ and MRR scores with varied k. In general, the prediction accuracy increases as k growing up. Concurrently, the experimental results show that Freq and MRU methods cannot predict anything, since their recalls are lower than 50%, even when $k = 7$. By contrast, the TAP-MinEntropy method outperforms all other methods. As can be seen in Figure 9, the accuracy even reaches to 100% when $k = 7$. In addition, it has 70% accuracy for only predict 3 Apps (i.e., $k = 3$). On the other hand, the MRR scores for the small dataset are shown in Figure 10, where

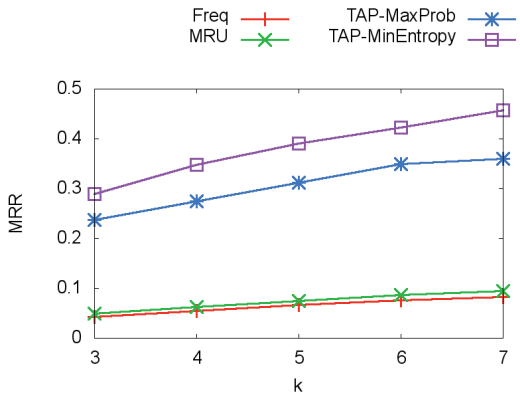


Figure 10: MRR scores comparison on the small dataset.

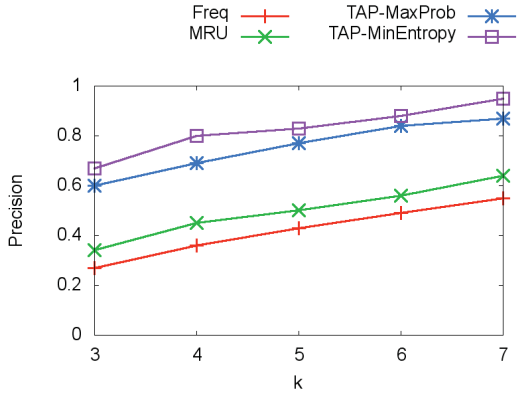


Figure 11: recall comparison on the large dataset.

the proposed TAP-MinEntropy also outperforms the other methods. It shows that MinEntropy selection correctly selects not only the Apps which are used by users but the raking of top-k list. Note that when $k = 7$, the MRR score is almost 0.5 for the proposed TAP-MinEntropy method, which means that users click the second App of the prediction list on average.

Then, we evaluate the accuracy on the large dataset. Figure 11 is the result of $recall$ evaluation. As can be seen in Figure 11, the proposed TAP-MinEntropy method obviously outperforms other methods. Especially, when $k = 7$, their $recall$ is around 100%. For smaller k (i.e., $k = 3$), the $recall$ is almost 70%. Figure 12 shows the similar result that TAP-MinEntropy is much better than the Freq and MRU methods.

6.2.2 *Efficiency Evaluation*

To evaluate the efficiency of the proposed methods and the baseline methods, we conduct experiments to compare the execution time among different methods. The efficiency evaluation of the execution time is performed on both the small and the large datasets when we predict 5 Apps ($k = 5$). Table 4 shows the results which imply that all selection algorithms need almost twice of execution time on the large dataset. Although the Freq and MRU methods outperform the other methods in the efficiency evaluation, their accuracy are only 40%. By contrast, the proposed TAP method with the MinEntropy selection could not only maintain high accuracy but acceptable efficiency.

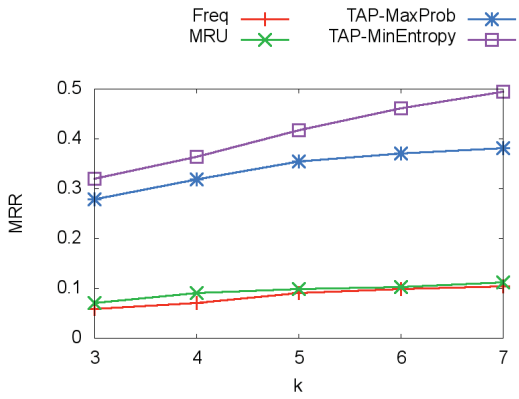


Figure 12: MRR scores comparison on the large dataset.

Table 4: The execution time in seconds.

Methods	Small	Large
Freq	0.020	0.060
MRU	0.025	0.063
TAP-MaxProb	0.041	0.131
TAP-MinEntropy	0.043	0.136

6.2.3 *Impact of Different Apps Usage Features*

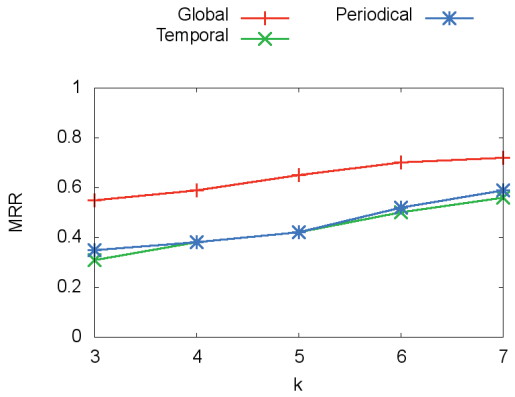
As each feature is good for predicting different Apps of different usage behavior, we try to understand the cross impact of prediction accuracy by different features. For example, we want to know the accuracy of predicting periodically used Apps by only global usage feature or temporal usage feature. First of all, we synthesis three datasets from the large dataset. Each dataset is consisted of only one usage behavior (i.e., frequently used, with specific usage time, or having significant usage period). Then, we predict k Apps by only the global, temporal, and periodical usage features respectively on all of the three datasets. Figure 13 shows the results where only one particular feature performs well on its corresponding dataset. It shows that each feature is isolated from others and deals with its corresponding behavior. In addition, the periodical usage feature is the most difficult one to be substituted by other features, since the $recall$ of periodical usage feature is much higher than the others in Figure 13(c). By contrast, the global usage feature is relatively easy to be replaced by the other two features as shown in Figure 13(a).

6.2.4 *Parameter Studies*

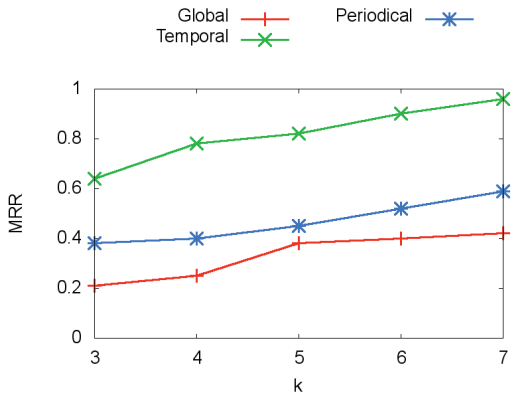
In this paper, the size of temporal bucket is the only parameter used in the proposed algorithms. We study the effect of different temporal bucket size, bks, for the temporal usage feature and periodical usage feature. The value of bks should not be too high because 1) there would be too many Apps in one temporal bucket, and 2) using higher bks cannot catch the user's usage infinities instantly. Here, we evaluate the accuracy by varying the range of bks from 10 minutes to 180 minutes. Figure 14 shows that both TAP-MaxProb and TAP-MinEntropy have highest recall when bks is set to 60 minutes. Accordingly, we suggest setting that bks equals to 60 minutes.

7. CONCLUSION

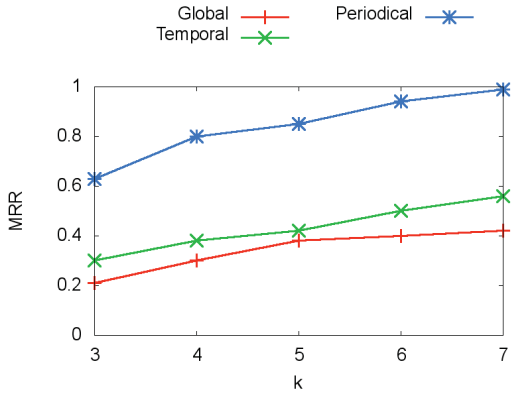
In this paper, we propose a Temporal-based Apps predictor (abbreviated as TAP) to predict k Apps which are most likely to be



(a) Global usage feature only



(b) Temporal usage feature only



(c) Periodical usage feature only

Figure 13: Prediction accuracy comparison under synthetic dataset with different usage features

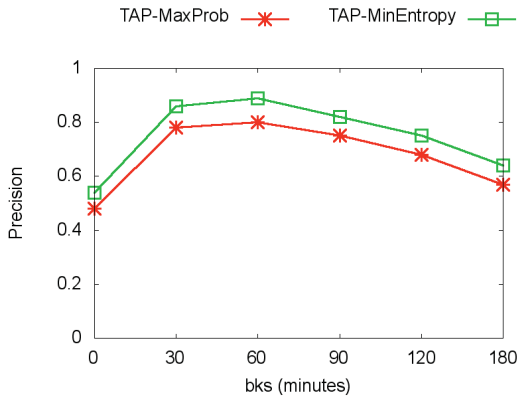


Figure 14: Impact of temporal bucket size.

used at a given query time. TAP consists of an off-line features extraction component and an on-line Apps prediction component. In the off-line component, we discover three temporal-based features, global usage feature, temporal usage feature, and periodical usage feature, from the collected Apps usage trace. In the on-line component, we build an Apps usage probability model through the discovered temporal-based features and the query time. A selection algorithm is introduced to investigate the three features and select k Apps with highest overall usage probability. In this paper, we propose two selection algorithms, MaxProb and MinEntropy, which are based on the concepts of selecting the maximum probability in all features and in the feature with minimum entropy respectively. In our experimental study, two real datasets are involved to test the effectiveness and efficiency of the proposed TAP algorithm. The results show that TAP with MinEntropy selection outperforms other methods.

8. ACKNOWLEDGEMENT

Wen-Chih Peng was supported in part by hTC, by the National Science Council, Project No. 102-2221-E-009-171-MY3 and 100-2218-E-009-016-MY3, by Academic Sinica Theme project, Project No. AS-102-TPA06, by Taiwan MoE ATU Program, by D-Link and by Microsoft. Po-Ruey Lei was supported in part by the National Science Council, Project No. 102-2221-E-012-002.

9. REFERENCES

[1] G. Amodeo, R. Blanco, and U. Brefeld. Hybrid models for future event prediction. In *Proc. of CIKM*, pages 1981–1984, 2011.

[2] M. Böhmer, B. Hecht, J. Schöning, A. Krüger, and G. Bauer. Falling asleep with angry birds, facebook and kindle: a large scale study on mobile application usage. In *Mobile HCI*, pages 47–56, 2011.

[3] H. Cao, T. Bao, Q. Yang, E. Chen, and J. Tian. An effective approach for mining mobile user habits. In *Proc. of CIKM*, pages 1677–1680, 2010.

[4] P. H. Chia, Y. Yamamoto, and N. Asokan. Is this app safe?: A large scale study on application permissions and risk signals. In *Proc. of WWW*, pages 311–320, 2012.

[5] D.-A. Chiang, Y.-H. Wang, and S.-P. Chen. Analysis on repeat-buying patterns. *Knowledge-Based Systems*, 23(8):757–768, 2010.

[6] D. Choujaa and N. Dulay. Predicting human behavior from selected mobile phone data points. In *Proc. of UbiComp*, pages 105–108, 2010.

[7] P. Cui, F. Wang, S. Liu, M. Ou, S. Yang, and L. Sun. Who should share what?: Item-level social influence prediction for users and posts ranking. In *Proc. of SIGIR*, pages 185–194, 2011.

[8] T. M. T. Do, J. Blom, and D. Gatica-Perez. Smartphone usage in the wild: A large-scale analysis of applications and context. In *Proc. of ICMI*, pages 353–360, 2011.

[9] T. M. T. Do and D. Gatica-Perez. By their apps you shall understand them: Mining large-scale patterns of mobile phone usage. In *Proc. of MUM*, page 27, 2010.

[10] H. Jeung, Q. Liu, H. T. Shen, and X. Zhou. A hybrid prediction model for moving objects. In *Proc. of ICDE*, pages 70–79, 2008.

[11] Z.-X. Liao, P.-R. Lei, T.-J. Shen, S.-C. Li, and W.-C. Peng. Mining temporal profiles of mobile applications for usage prediction. In *12th IEEE International Conference on Data Mining Workshops, ICDM Workshops, Brussels, Belgium, December 10, 2012*, pages 890–893, 2012.

[12] D. Lymberopoulos, P. Zhao, A. C. König, K. Berberich, and J. Liu. Location-aware click prediction in mobile local search. In *Proc. of CIKM*, pages 413–422, 2011.

[13] H. Ma, H. Cao, Q. Yang, E. Chen, and J. Tian. A habit mining approach for discovering similar mobile users. In *Proc. of WWW*, pages 231–240, 2012.

[14] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti. Wherenext: A location predictor on trajectory pattern mining. In *Proc. of KDD*, pages 637–646, 2009.

[15] D. R. Radev, H. Qi, H. Wu, and W. Fan. Evaluating web-based question answering systems. In *Proc. of LREC*, 2002.

[16] C. E. Shannon. A mathematical theory of communication. *Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[17] K. Shi and K. Ali. Getjar mobile application recommendations with very sparse datasets. In *Proc. of SIGKDD*, pages 204–212, 2012.

[18] C. Shin, J.-H. Hong, and A. K. Dey. Understanding and prediction of mobile application usage for smart phones. In *Proc. of UbiComp*, pages 173–182, 2012.

[19] M. Vlachos, P. Yu, and V. Castelli. On periodicity detection and structural periodic similarity. In *Proc. of SDM*, 2005.

[20] T. Yan, D. Chu, D. Ganesan, A. Kansal, and J. Liu. Fast app launching for mobile devices using predictive user context. In *Proc. of MobiSys*, 2012.

[21] J. Yi and F. Maghoul. Mobile search pattern evolution: The trend and the impact of voice queries. In *Proc. of WWW*, pages 165–166, 2011.

[22] J. Yi, F. Maghoul, and J. O. Pedersen. Deciphering mobile search patterns: A study of yahoo! mobile search queries. In *Proc. of WWW*, pages 257–266, 2008.

[23] V. W. Zheng, B. Cao, Y. Zheng, X. Xie, and Q. Yang. Collaborative filtering meets mobile recommendation: A user-centered approach. In *Proc. of AAAI*, 2010.