On the Understanding of Interdependency of Mobile App Usage

Jiaxin Huang, Fengli Xu, Yujun Lin, and Yong Li.
Tsinghua National Laboratory for Information Science and Technology (TNLIST)
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
Email: liyong07@tsinghua.edu.cn

***Abstract*—With the rising popularity of smartphones and the rapid growth of mobile applications, understanding the app usage behavior of mobile users is of growing importance for both app designers and service providers. Different from previous studies mining the correlation between apps and physical world factors, e.g. location, time, etc., in this paper we focus on the interdependency among apps and try to address a series of research problems: what apps are frequently used together? What are the relations between apps (strengthen or undermine the use of each other) belonged to each category? To answer these questions, we employ the frequent pattern mining algorithm to a large-scale real-world dataset, which includes more than 1.7 million users and 5 billion app usage logs, and find out frequent app-sets and association rules with interesting insights. These results are usefulness in app marketing for service providers and in understanding different mobile users for app designers.**

***Index Terms*—Mobile big data, mobile application usage, frequent pattern mining**

I. Introduction

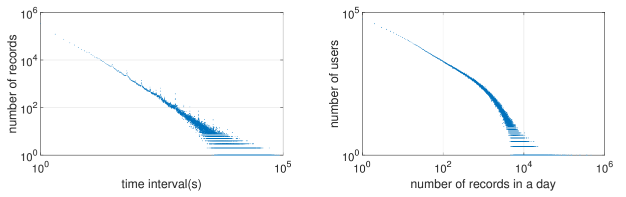
These days people are using apps on smartphones more and more frequently, and their app usage behaviors contain rich information. Mining the patterns in these behaviors is important and meaningful, which helps better profile mobile app users and customizing the offered services. Abundant previous researches focus on the correlations between contexts and apps usage[4][5], different patterns and regularity of app downloading, usages[6][7], and revisitations[8]. These former researches focus on either the app itself or the relations between apps and the physical world.

However, the inherent patterns of interdependency between different apps are also important in deepening our understanding of apps usages and users behaviors. In this paper, we focus on investigating the patterns and correlations among apps themselves, such as what apps are frequently used together, and what types of apps tend to promote or undermine the use of other apps in the same category. The answers to these questions are very useful to reveal the natural relations among apps, so that app designers can adjust the functions to meet the needs of mobile users and service providers can increase their profits by bundle sales of apps.

To answer the above questions, we look at a large-scale app usage dataset collected by Deep Packet Inspection(DPI) appliances from cellular network in Shanghai, one of the largest city in China. Each entry in the dataset contains the information of packet header of the HTTP request or response and the base station that the packet arrives at. Based on this dataset, we identify the apps responsible for each log by exploiting the features of packet headers. After that, we apply the frequent pattern mining algorithm to obtain the most frequent combinations of apps launched by users, and further analysis of insights can be carried out. However, there are two challenges to achieve these goals. First, identifying what app the user is using from the HTTP packet header is nontrivial. Especially when limited ground truth is available, we need to develop unsupervised methods to learn the mapping rules. Second, while applying the frequent pattern mining algorithm to our dataset seems simple, carrying out valid and meaningful analysis on the frequent itemsets requires a lot of effort, since the number of frequent patterns grows explosively as the threshold goes down. To solve these two challenges, in this paper we develop a method to map the HTTP header to its responsible app to carry out a frequent pattern mining analysis and reveal abundant analysis of insights behind these frequent patterns. What's more, different kinds of applications can make use of these frequent patterns to obtain performance gains. Our contribution can be summarized as the following two aspects.

- We develop a system to identify what app is used via HTTP header packet sent by the mobile phones of users. Specifically, we first use the field of user-agent to filter out the browsers, which are a great interfere to be removed. Then, we crawl the possible values of user-agents of the most popular apps in App Store and Google Play to extract the features of apps, and further use statistical information to classify the same apps with different user-agent names.
- We apply frequent pattern mining algorithm on the app usage dataset and discover various frequent patterns. Using different metrics, we find out app-sets with different interesting properties, association rules and different correlation patterns among apps.

The rest of this paper is structured as follows. Our dataset is introduced and visualized in Section II. The basic methods to mine frequent patterns and reveal physical insights are introduced in Section III and Section IV. In Section V we draw our conclusion and discuss related future work.



(a) Intervals between two records (b) Number of records of each user

Fig. 1. Dataset characteristics to show their fine-grain property.

TABLE I
EXAMPLES FOR THE COLLECTION PROCESS OF APP UNIQUE IDENTIFIER.

Source	Identifier Type	App Name
<id im:id="299853944" im:bundleId="com.sina.sinanews">	bundleId	Sina News
https://itunes.apple.com/us/ app/weibo/id350962117?mt=8	url	Weibo
<title id="main-title">Alipay - Android Apps on Google Play</title>	main-title	Alipay
https://play.google.com/store/ apps/details?id=com.dianping.v1	id	DianPing

II. Dataset and Visualization

A. Dataset Description

Our dataset is collected by one of the major mobile operators in Shanghai, one of the largest cities in China using Deep Packet Inspection (DPI) appliances. The traces last from April 20th to April 26th in 2016 and cover the whole metropolitan area of Shanghai, including urban as well as rural areas. Each entry contains an anonymized User Identification, timestamps of HTTP request or response, the length of the packet, the domain visited and the user-agent field.

The dataset has a total volume of more than 800 gigabytes, covering more than 1,700,000 users and 5 billion logs. The average time interval between two records is 222 seconds, which indicates that it is fine-grained. To show the richness of our data, in Fig.1(a), we plot the interval between two records. The power law can be found in the figure, and as the interval becomes longer, the number of records drops quickly. Most of the intervals between two records are less than 1000 seconds. The number of records of a single user in a day is plotted in Fig.1(b). From the results, we can observe that the number of records generated by a user each day goes down smoothly between the range of 1 to 1000, but drops drastically at points larger than 1000. While the most active mobile user can generate up to hundreds of thousands of records in a day. All these results demonstrate the fine-grain property of our dataset, which guarantee the credibility of our study.

It is worth pointing out that privacy issues of this dataset are seriously considered and measures are taken to protect the privacy of these mobile users. Our dataset is collected via our collaboration with the ISP network, and the data does not contain personally identifiable information. The "user ID" field has been anonymized (as a bit string) and does not contain any user meta data. All the researchers are regulated by strict non-disclosure agreement and the dataset is located in a secure off-line server. In our dataset, more than 80% traffic uses HTTP at the time of data collection, and there is little affection by HTTPS protocol.

B. Preprocessing

Our study need to first identify the app corresponding to each HTTP flow. The reason that we are able to identify this is that in the HTTP header, fields are utilized as the identifiers of the apps to communicate with their host servers or third party services. The hosting servers need to tell different apps apart in order to provide proper contents. Therefore, we are able to identify the app if we dedicated distinguish the identifiers in each trace. We design the app identification system by the following four steps.

1) Removing browsers' records from traces: Firstly, HTTP packets sent by browsers must be filtered out since they are beyond our scope. We go through all the mobile browsers in the main-stream app market, and find that they can be removed by identifying the keywords in user-agent field, such as "Mozilla", "Opera" and "MQQBrowser". It is important to point out that the user-agents of most browsers begin with "Mozilla", which reduces a great number of keywords to compare and facilitate the filtering. Specifically, by comparing the user-agent field with the pattern of $\hat{}Mozilla.+$, we can filter out the records generated by Mozilla. Similar, other browsers like Opera and QQBrowser for IOS can also be filtered out by replacing the "Mozilla" in the above pattern by "Opera" and "MQQBrowser".

2) Collecting dictionaries for IOS and android apps: In the second step, we collect the possible patterns of user-agents of the most frequently used apps. We crawl the top 50 apps in each categories in App Store(IOS apps) and Google Play(Android apps). Since we only focus on the frequent app usage pattern, only the most frequently used apps are needed, and therefore 50 apps for each category is more than enough. For IOS apps, we use the API provided by App Store and get the *bundleId* and *url* of downloading page of each app. For Android apps, we crawl the *id* for each app and *main-title* on the downloading page. Four dictionaries are created to map the *bundleId/url/id/main-title* to a unique app, which are the unique identifier for each app to map the user-agents to apps. A sample of the collection process is shown in Table I.

3) Mapping user-agents to apps: In this step, we deal with the user-agents in the traces and match them to the keys of four dictionaries. Usually, one app will have different versions, while we want to treat them as the same app and do not care about their versions. Thus, we remove the version number behind the slash('/').

4) Classifying the unidentified user-agents: In the last step, we classify unidentified user-agents into identified ones. This is done by calculating the conditional probability of occurrence in the records of a same user. For example, BaiduMap(A) and MobileMap(B) are two different user-agents, but we have no idea what is MobileMap. Since the ratio of users using both A and B to users using B(the ratio equals to $P(A|B)$) equals 0.9918 and $P(B|A)$ equals 0.9995, we can infer that user-agents MobileMap and BaiduMap belong to the same navigating application.

With the above four steps, we identify 1300 user-agents that cover more than 85 percent records of the whole dataset. We further filtered out inactive users that use less than three apps one day and finally obtain 730, 000 users.

III. Mining the Frequent App-sets

In this section, we introduce how we use the dataset and the frequent itemset mining algorithm to obtain a comprehensive understanding of app usage pattern. Fig. 2 shows the framework of our proposed mining system. The process of the system can be divided into three main stages. In the first stage, we apply the algorithm of FP-Growth[3] to our app dataset to find out the frequent app-set, and further analyze the underlaying physical insights. In the second stage, we generate strong association rules[1] to design a recommendation system. In the final stage, we compress the frequent app-sets and use them as the main features to classify users.

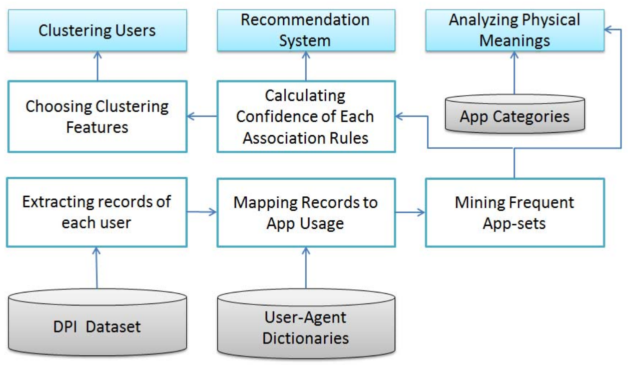

Clustering Users
Recommendation System
Analyzing Physical Meanings
Choosing Clustering Features
Calculating Confidence of Each Association Rules
App Categories
Extracting records of each user
Mapping Records to App Usage
Mining Frequent App-sets
DPI Dataset
User-Agent Dictionaries

Fig. 2. System framework of app frequent itemset mining and corresponding applications

A. Basic Definitions

Let $I = \{I_1, I_2, \cdots, I_m\}$ be the set of all apps that we identified from the dataset, where I_i refers to a unique item of app. Let D be the dataset, and the set of apps used by each user is defined as transaction T ($T \subseteq I$). Let A be a set of items, or itemset as we call it later. If a transaction T contains every element in A, then $A \subseteq T$. The support, denoted as S, of A is defined to be

$$S(A) = \frac{\#(T|A \subseteq T)}{\#(D)}. \quad (1)$$

With a threshold S_{min}, A is a frequent itemset if it satisfies

$$S(A) > S_{min}. \quad (2)$$

Once we acquire the frequent itemset results, we can generate association rules, expressed as $A \Rightarrow B$, where $A \cap B = \emptyset$. A valid association rule should satisfy both the support threshold and confidence threshold. "Confidence", denoted as C, is another basic concept in frequent pattern mining.

$$C(A \Rightarrow B) = P(A|B) = \frac{S(A \cup B)}{S(B)}. \quad (3)$$

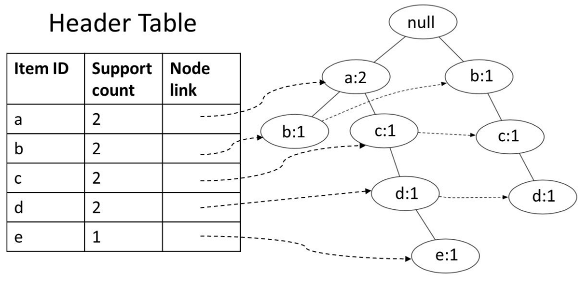
An association rule is valid if it satisfies two requirements below.

$$S(A \cup B) > S_{min}. \quad (4)$$

$$C(A \Rightarrow B) > C_{min}. \quad (5)$$

1) Frequent App-sets Mining: Frequent pattern mining searches for patterns that appear frequently in a dataset, which leads to interesting findings such as inconspicuous relations between different items. For simplity, frequent pattern mining finds out itemset that has a support higher than S_{min}.

In our dataset, we define each user to be a "transaction" T, and the mobile apps as items I_i. Therefore, the frequent itemsets, or app-sets, are collections of apps that are frequently used. Algorithms like Apriori[2] and FPGrowth[3] are designed to search for the itemsets that has a support higher than S_{min}. We apply FPGrowth to our dataset since it finds frequent itemsets without generating candidates and saves much space. The FPGrowth algorithm creates a prefix tree and scan the whole dataset twice. In the first scan, the set of frequent items are inserted into the empty tree in a descending order. In the second scan, each transaction is inserted into the tree by sharing their collective frequent items as prefixes. A header table is created to connect the same items in the tree, and is later used to create the conditional pattern base for each item. After that, a recursive process is carried out to find the frequent itemsets that end with different suffixes. A simple example of an FP-tree is shown in Fig. 3.



Item ID	Support count	Node link
a	2	
b	2	
c	2	
d	2	
e	1	

Fig. 3. FP-tree compresses the whole dataset.

B. Apps Classification

To analyze the frequent app-sets that we found, we have to first understand the main functions of each app. Since we crawl the apps by category in the first place, we categorize each app into the category which we crawl them from(e.g., Games, News and Finance). However, minor modifications are made to specify several kinds of apps. For example, we separate "Travel" into two groups, "Travel" and "Taxi" to distinguish

TABLE II
THE TOP FIVE FREQUENT ITEMSETS WITH HIGHEST SUPPORTS

app-set	**support**	**app-set**	**support**
Taobao, Alipay	0.3527	GaodeMap, Taobao	0.2930
WeChat, Taobao	0.2790	GaodeMap, Alipay	0.2619
WeChat, QQ	0.2520	-	-

apps that provide taxi services for short-distance travel from apps that help people book plane tickets for long-distance trips. We have 18 categories, and wiith these categories, we are able to further analyze the role of app functions in the interdependency among apps to find insightful results.

IV. RESULTS

The distribution of the support of the obtained 3963 frequent app-sets is high, where the most frequently used app has a support of nearly 60% and the most frequent combination of apps has a support of 35%. Only around 500 app-sets are used by more than 3 percent of the total mobile users. The results shows that only a limited number of apps are popular among the mainstream. Notice that even a small number of apps can come up with a great many kinds of combinations. This is good news for us because the very few dominant apps are able to grasp the nature of app usage behavior of the masses.

A. App-items with Highest Support

In order to investigate the underlaying patterns, the support of the first five most frequent app-sets are listed in Table II. The most frequent app-set is Taobao (the most popular E-commerce app) and Alipay (the largest third-party online payment app). The popularity of this combination is obvious because more and more people are shopping online using their mobile phones these days and Alipay is the most prevalent online payment method. GaodeMap and Taobao are often used together for people who enjoy both online shopping and traveling around. Wechat is the most popular instant message and socialization app. People using both Taobao and Wechat are likely to recommend the items they buy on Taobao to their friends by forwarding the link of the items from Taobao to their Wechat friends. Gaode Map and Alipay are often used together, because when people go to a new place for entertainment, they can use GaodeMap to look up what public transportation to take or which route saves their time the most, and they will use Alipay to do payments for their entertaining activities. QQ is the previous mainstream instant message app for a decade, thus a large portion of people in the transition period are still using both WeChat and QQ.

Through these analyses, we are able to find out that apps frequently used together have strong association between themselves, since they reflect the daily activities of people. Driven by these results, we generate more interesting association rules between apps in the next paragraph.

TABLE III
ASSOCIATION RULES WITH HIGHEST CONFIDENCE

A	B	$A \Rightarrow B$
Meituan,Mito XiuXiu	WeChat	0.9934
WeChat, Headlines Today, SinaNews	QQNews	0.9913
Alipay, XianYu	Taobao	0.9911
WeChat, GaodeMap, DianPing, DidiTaxi, Meituan	Dida Carpool	0.9594

B. App-items with Highest Confidence

To generate valid association rules, we calculate the confidence $p(B|A)$ where B is a single app picked from each of the frequent app-set, and A is the rest apps in that app-set. The association rules with the highest confidence is listed in Table III. The strongest association rule contains Meituan(a discount sales platform) and Mito Xiuxiu (a photo enhancement app like photoshop). Meituan users complete their payments through WeChat Wallet, a mobile payment service, and Mito Xiuxiu users are very likely to post their selfies on a social app like WeChat. It is reasonable to infer that users using these apps are young women who like to hang out with their friends, using Meituan to get a discount, finishing their payments through WeChat, and posting one or two pictures on WeChat via the photo enhancement of Mito Xiuxiu.

Users who use WeChat, Headlines Today and SinaNews are those who enjoy reading news, and there is a high possiblity that they use different kinds of News reading applications, such as QQNews. They can use WeChat as a platform to share news with their friends by forwarding the articles on a news app to their WeChat groups or WeChat friends.

The third association rule has a lot to do with online transactions. Users who use these apps are those who prefer shopping online to gain a discount. Alipay is a third-party online payment app, and XianYu is a flea market platform where people buy and sell second-hand stuffs at a very low price. Taobao is an E-commerce platform where users can also sell their own stuffs. Users pay for products on XianYu using Alipay, and they also use Alipay with their frequent shoppings on Taobao.

C. Apps with Low Correlation

Now, we find out what apps have a very low chance of being used together by using a low Kulc filter[1] to achieve this goal. Kulc value is a metric measuring the association between two items. Suppose there are two apps A and B, then the Kulc value for A and B(which we denote as $K(A, B)$) is defined below.

$$K(A, B) = \frac{P(A|B) + P(B|A)}{2}. \tag{6}$$

An association rule with a very low Kulc value means that the existence of either side significantly undermines the other side. In our dataset, this means that users tend not to use those two apps together. The association rules with low Kulc value

TABLE IV
RESULTS OF ASSOCIATION RULES WITH LOW KULC

A	B	$Kulc(A,B)$
iQiyi Video	Headlines Today	0.1002
QQLive	Didi Taxi	0.1091
Happy Elements	QQNews	0.1200
Qzone	WeChat	0.1333

is listed in Table IV. From the result, we can observe that iQiyi Video (an app to watch TV series and movies) and Headlines Today are not often used by one person, since iQiyi Video is favored by those who enjoy watching films and TV series played by their favorite real-world idols or virtual characters, different from people who spend time reading news happening in the real world. QQLive and Didi Taxi is not used together, because watching live broadcast consumes a lot of traffic, and is costly for mobile phone users not connected to a WiFi. This corresponds to the fact that people have fewer chances to connect to a WiFi when taking taxis. Happy Elements is a puzzle game designed to occupy one's free time, so people watching news on QQNews frequently do not often play it. For the last association rule, Qzone is a blog-like website frequently used by teenagers, and have different target users from WeChat, of which the majority of users are adults.

D. High Lift within Same Type

Lift measure [9] is used to ensure that apps in chosen itemsets truly promotes the use of other ones in that same app-set. The definition of Lift(which we denote as L) is shown below.

$$L(A,B) = \frac{P(A \cup B)}{P(A)P(B)} = \frac{C(A \Rightarrow B)}{S(B)}. \qquad (7)$$

If $L(A,B) > 1$, then the app A and B promotes the use of each other. On the other hand, two apps undermine the use of each other if $L(A,B) < 1$. Using the definition of Lift, we obtain the results in Table V. From the results, we can find that most types of apps promote the use of other apps in the same cateogry, including News, Games, E-commerce, etc. However, Navigation maps undermine each other slightly, since one map app is enough for common navigating use and downloading another map does not bring more information. Also, Qzone and WeChat undermines each other seriously because these two social apps target at users at quite different ages.

Through these analysis, we provide the answers to the questions of what apps are frequently used together, and find that the association within these frequent app-sets truly reflects people's daily activities. Moreover, we also detect apps of which category are seldomly used together, and reveal the reasons behind these phenomena. In summary, we design an intuitively method that reveals both comprehensive and primary understanding of the correlations among apps.

TABLE V
LIFT OF APPS IN THE SAME CATEGORY

Cateogry	A	B	**Lift**
News	QQNews, Headlines Today	Sina News	22.4364
Games	Happy Joker	QQ Game	15.8857
Life Service	Dianping, Baidu Nuomi	Meituan	5.3960
E-commerce	Alipay,Taobao	XianYu	1.9941
Social	WeChat, MomoChat	QQ	1.5628
Video	iQiyi Video	QQLive	1.2912
Map	BaiduMap	GaodeMap	0.9852
Social	Qzone	WeChat	0.3618

V. CONCLUSION AND FUTURE WORK

In this paper, we utilize the frequent itemset mining algorithm to understand the patterns of app usage behavior. Using different indexes like confidence, Kulc and Lift, we extract various insightful association rules. As future work, we will make use of the association rules by implementing applications of recommendation system and user clustering.

REFERENCES

[1] J. Han, M. Kamber, "Data Mining: Concepts and Techniques," third edition, 2012, pp. 157–210.

[2] R. Agrawal, R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," in *International Conference on Very Large Data Bases*, 1994, pp. 487–499.

[3] J. Han, J. Pei, Y. Yin, "Mining frequent patterns without candidate generation," in *Proc. ACM Sigmod Record*, 2000, pp. 1–12.

[4] T. M. T. Do, J. Blom, D. Gatica-Perez, "Smartphone usage in the wild: a large-scale analysis of applications and context," in *Proceedings of the 2011 International Conference on Multimodal Interfaces.* ACM, 353-360.

[5] B. Cici, M. Gjoka, A. Markopoulou, and C. T. Butts, "On the decomposition of cell phone activity patterns and their connection with urban ecology," in *Proc. ACM MobiHoc* (Hangzhou, Chin), Jun. 22-25, 2015, pp. 317–326.

[6] H. Li, X. Lu, X. Liu, T. Xie, K. Bian, "Characterizing Smartphone Usage Patterns from Millions of Android Users," in *Proc. ACM IMC*,2015, pp. 459–472.

[7] Z. Liao, Y. Pan, W. Peng, P. Lei, "On mining mobile apps usage behavior for predicting apps usage in smartphones," in *Proc. ACM CIKM*, 2013, pp. 609-618.

[8] S. L. Jones, D. Ferreira, S. Hosio, J. Goncalves, V. Kostakos, "Revisitation analysis of smartphone app use," in *Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015, pp. 1197–1208.

[9] CC. Aggarwal, PS. Yu, "A new framework for itemset generation," in *Proc. 1988 ACM Symp. PODS'98*, 1999, pp. 18–24.

[10] K. S. Jones "A statistical interpretation of term specificity and its application in retrieval," in *Journal of Documentation*, 1972, pp. 11–21.

[11] J. Tang, J. Liu, M. Zhang, Q. Mei, "Visualizing Large-scale and High-dimensional Data," in *International Conference on World Wide Web*, 2016, pp. 287-297