Identifying In-App User Actions from Mobile Web Logs

Bilih Priyogi[1], Mark Sanderson[1], Flora Salim[1], Jeffrey Chan[1], Martin Tomko[2], and Yongli Ren[1]

[1] School of Science, Computer Science and Information Technology, RMIT University, Australia

[2] Department of Infrastructure Engineering, University of Melbourne, Australia

{bilih.priyogi,mark.sanderson,flora.salim,jeffrey.chan}@rmit.edu.au, tomkom@unimelb.edu.au, yongli.ren@rmit.edu.au

Abstract. We address the problem of identifying in-app user actions from Web access logs when the content of those logs is both encrypted (through HTTPS) and also contains automated Web accesses. We find that the distribution of time gaps between HTTPS accesses can distinguish user actions from automated Web accesses generated by the apps, and we determine that it is reasonable to identify meaningful user *actions* within mobile Web logs by modelling this temporal feature. A real-world experiment is conducted with multiple mobile devices running some popular apps, and the results show that the proposed clustering-based method achieves good accuracy in identifying user actions, and outperforms the state-of-the-art baseline by 17.84%.

Keywords: transaction identification, mobile web logs

1 Introduction

Mobile devices have become an important component of people's daily life, allowing near ubiquitous access to services on the Internet. Such accesses can potentially be logged. It is vital to understand users' needs by mining their Web usage logs, e.g. grouping Web accesses into meaningful units to find patterns [19, 6]. Although Web log mining has been extensively studied for many years, the focus of such work is on the logs of Web sites [6, 20]. Providers of free Internet access, such as shopping malls, airports and train stations, can collect a different sort of log: a mobile Web log capturing accesses from users accessing different services [1]. This log is quite different from those captured from a single Web site. Most of the Web requests will be encrypted (through HTTPS protocol) and there are a mix of user-driven accesses to Web sites/services, as well as automated accesses sent by the apps installed on user phones as observed by [17]. Accurately identifying and understanding in-app user actions from these mobile Web logs is critical in many fields, ranging from promoting personalised Web services to enforcing user activity monitoring and information security [11, 8, 10, 16]. For example, mobile users will be provided the right information to satisfy their needs at the moment; the providers of free Internet can automatically monitor users' Web behaviours at high level and immediately be alerted when a potential risk user (or group of users) does something dangerous; and researchers in this field can learn lessons and build various models for modelling mobile users, etc.

Users' true actions in apps can be used to understand or infer users' behaviour. Users' behaviour can be extracted from mobile Web logs using various data mining techniques, such as association rule mining [3, 12]. However, the mining result might be too coarse if we use a single log entry as a viewpoint, while a single user session might be too broad to give a fine-grained knowledge. Thus, it is necessary to group Web logs into meaningful units, in order to provide proper meaningful granularity for mobile user behaviour research [6, 19, 21].

We attempt to identify users' true actions from such logs. The challenges include: 1) as the concern of privacy and security become more prominent, more mobile apps are encrypting their Web accesses. This means that only hostname would be visible and it is impossible to know the content requested, including textual content and filename suffixes, which makes existing approaches infeasible [6], e.g. identifying access by using file suffixes. 2) the multitasking nature of a mobile device allows numerous applications to access the Web almost simultaneously, which should be taken into account before identifying in-app user actions within the corresponding sequential logs. 3) Automated URL requests issued by mobile applications also introduce bias in determining user actions, because it is not directly triggered by the user. Then, a new research question appears:

How to identify in-app user actions of Web accesses from encrypted and noisy mobile Web logs?

To tackle this problem, we introduce the concept of a *transaction* in the context of mobile Web logs, and propose a method to identify them. Specifically, a *transaction* is a group of *sequential Web access* (URLs) to one or more relevant Web domains, which correspond to a singular user action in a single mobile app. Moreover, we found that the distribution of time gaps between Web accesses, when there are user actions, is significantly different from that when there are no user actions. This indicates that it is reasonable to identify *transactions* within the Web logs with clustering techniques by modelling this time gap feature. Another reason why we model transaction identification by using clustering technique is because there are generally no labels in mobile Web logs and unsupervised learning is more fitting. Finally, to evaluate the performance of the proposed method, we conducted a controlled real-world experiment with multiple devices and mobile applications. Note the labels gathered in controlled experiment are only used for testing purposes. The experimental results show the proposed method significantly outperforms the state-of-the-art in terms of identifying *transactions* from mobile Web logs.

2 Related Work

Here, we briefly review relevant works about transaction identification in traditional Web site logs and traffic analysis in mobile logs.

Shu-yue et al. [19] and Cooley et al. [6] discuss several preprocessing techniques needed before executing mining algorithms on Web site logs. The techniques include data cleaning, log consolidation, log formatting, user identification, session identification, and transaction identification. These preprocessing techniques aim to provide high-quality data to ensure logs become more credible, accurate, and representative so that log mining and analysis can be conducted effectively. From Web server perspective, [2] investigates automated network request in the query stream of a large search engine provider's logs. [2] also proposes some features to distinguish between queries generated by people actually searching for information and those generated by the autonomous process.

Both [19] and [6] discuss techniques to identify transactions within a log in order to cluster log entries into meaningful units. Cooley et al. [5] discussed three transaction identification methods: *Reference Length* (RL), *Maximal Forward Reference* (MFR), and *Time Window*. Both RL and MFR categorized each page accessed as either a navigational or content page. A navigational page was considered to be only used by users to locate pages of interest, while a content page is a page containing desired information. The RL technique is based on an assumption that the amount of time a user spends on a page indicates whether the page should be classified as a navigational or content page. Then, in RL, a transaction can be defined as a sequence of navigational pages that lead to a content page. In MFR a transaction is defined as a set of page accesses before a backward reference is made. Such a reference is defined as a page not already in the set of pages from a current transaction. Different from aforementioned approaches, *Time Window* does not try to identify a page as navigational or content, but assumes that meaningful transaction should have an average overall length associated with them. User sessions were partitioned into time intervals no larger than a specified threshold.

Chen et. al. [4] and Li et. al. [14] proposing their own approach to identifying transactions. However, the technique is not feasible because of encryption in modern mobile Web traffic. In work by [15] and [13] a device-level point of view is taken to investigate network activity characteristics of mobile devices. Both authors gather and analyse data from users who voluntarily installed an app on their device, which captured their applications' network activity over a period of time. Similar to the work in this paper, [15] and [13] are also explanatory in nature and serve as an initial step toward a global network utilization model of mobile devices. However, little research was conducted to understand the meaningful *transaction* unit in the context of mobile Web traffic. In this paper, in the context of mobile traffic logs, a transaction is defined as a group of URL requests without considering its content, and a data mining based transaction identification method has been investigated.

3 Transaction Identification

Here, we first define the concept of a *transaction* in mobile Web logs to describe in-app user actions, then propose a method to identify them by modelling temporal features.

3.1 Definition

Based on a transaction definition in traditional Web log mining [6], a *transaction* t is defined as a meaningful group of sequential network activities (URL requests) in one application (*app*) on a user's mobile *device*. Transaction t is a tuple:

$$t = < device_t, app_t, \{(url^t_1, time^t_1), ..., (url^t_n, time^t_n)\} > \tag{1}$$

where $(url^t_i, time^t_i)$ denotes the i-th URL request from app_t on $device_t$ in transaction t. Namely, this indicates that *transactions* are defined per user (*device*) and *app*. Note, *url* could be the full URL request, or only the hostname if it is encrypted (e.g. through HTTPS protocol).

3.2 Identifying Transactions

We first conducted a comprehensive analysis of the time gap feature in mobile Web logs, which is defined as the gap in seconds between consecutive URL requests from the same *device* and *app*. We examine the logs of six representative *apps* (Facebook, Twitter, Instagram, Path, MSN, Sina) on three different devices (Android Tablet, iOS Phone, iOS Tablet), which will be detailed in Sec. 4.1. There are other apps and devices of interest, but we believe these apps and devices provide a good representative sample to analyse on. The gaps are separated into two groups: *idle* that means there are no user actions, and *active* that means there are user actions with the device. A Kolmogorov-Smirnov (KS) test is deployed to examine whether the *idle* vs *active* time gap distributions are significantly different to each other. The KS test results are shown in Table 1, indicating that there is a statistically significant difference between *idle* and *active* time gaps across all tested apps and devices, e.g. for *facebook* on Android Table, the KS statistic $D = 0.340$ and p-value ≤ 0.0001. This indicates that it is possible to identify transactions by modelling this temporal gaps.

Furthermore, Fig. 1 illustrates the example of difference between *idle* and *active* time gaps, in terms of *Expected Cumulative Distribution Function* (ECDF). We can observe that time gaps on *active* device seems to be more rapid than its counterpart. In other words, user actions on *active* device evidently caused a burst of consecutive URL requests. In contrast, consecutive URL requests occurred intermittently when there is no user actions.

Based on this analysis, we treat the problem of *transaction identification* as a clustering problem on the temporal gap feature. Specifically, we deploy the classic DBSCAN clustering method [7] to perform the transaction identification. The main reason is that, when modelling the temporal gaps, DBSCAN takes into account the sequential nature of the data (network activities) via its unique density connectivity concept. Namely, DBSCAN clusters periods of high activity, separated by gaps of idles, which is the characteristics of the transaction identification problem.

Table 1. *KS*-test results of comparing time gap distribution between *idle* and *active* mobile usage. Note D is the two-sample KS statistic, indicating it is greater than the corresponding critical value.

Device	*app*	D	p-value
Android Tablet	Facebook	0.340	<0.0001
	Twitter	0.333	<0.0001
	Instagram	0.364	<0.0001
	Path	0.341	<0.0001
	MSN	0.460	<0.0001
	Sina	0.494	<0.0001
iOS Phone	Facebook	0.297	<0.0001
	Twitter	0.311	<0.0001
	Instagram	0.294	0.016
	Path	0.306	<0.0001
	MSN	0.453	<0.0001
	Sina	0.490	<0.0001
iOS Tablet	Facebook	0.299	<0.0001
	Twitter	0.299	<0.0001
	Instagram	0.297	<0.0001
	Path	0.299	<0.0001
	MSN	0.404	<0.0001
	Sina	0.389	<0.0001

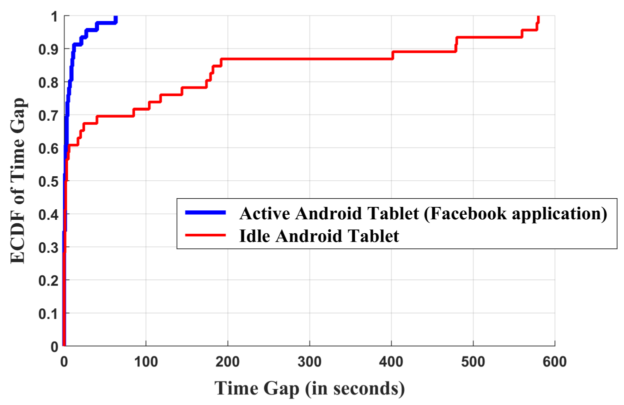

Fig. 1. ECDF of time gap feature in two cases: *Idle* VS. *Active*

4 Experiment

4.1 Experiment Environment

We configured a controlled Wi-Fi network to collect Web logs from connected mobile devices, as shown in Fig. 2. Each of mobile devices connected to the wireless access point via wi-fi, as the only available communication line to the Internet. A computer acts as a proxy server that records all URL requests coming through a wireless access point. Untangle [3] used as network gateway management software, which provides the capability of capturing network traffic. There are four different devices in the experiment: an Android tablet (Nexus 7 3G), iOS phone (iPhone 4S), iOS tablet (iPad 2), and iOS music player (iPod Touch 4th Gen). A factory reset was performed on all devices prior to the experiment to minimize noise introduced by previously installed apps. All devices were installed with the latest version of each application that was available for the devices latest OS version.

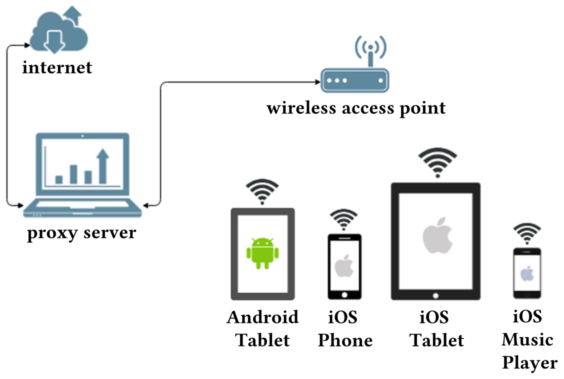

Fig. 2. Experiment Configuration

To better simulate the real-world usage, six representative popular applications were installed on each device: Facebook, Twitter, Instagram, Path, MSN, and Sina. However, on iOS music player, MSN application could not be installed due to incompatible older OS version. These applications were selected because presumably, their rich and constantly updated content nature will give an adequate volume of network traffic to observe for. This set-up expected to give controlled environment in terms of unrelated adwares network access that might be embedded in some applications, while still represents real world mobile device usage. All applications were logged into with the same user account across all devices, except for MSN and Sina, to ensure they receive a similar volume of content. Note that, in this experiment, we only consider URL requests issued

[3] www.untangle.com

by applications, and not from the browser. Then, a user usage scenario was simulated to capture network activities on the active (used) device. Specifically, five user actions are defined, conducted, and considered within 1-minute time boundary to simulate real world application usage by users, as listed in Table 2. Note that in this scenario, automated and user triggered URL requests might get mixed in the Web logs captured. Thus, only URL requests to Web domains that related with the tested application at the time are considered as URL requests triggered by user, otherwise it will be labelled as automated URL requests. Furthermore, URL requests that occurred outside the time boundary of scheduled usage scenario will also be considered as automated URL requests.

Table 2. User Actions Examined

Minute	User Action	Description
1	Open App	starting an application session
2	Browsing	reading content and scrolling through at normal reading speed
3	Dwelling	reading one post and stop scrolling through
4	Skimming	reading content and scrolling through at skim reading speed
5	Close App	closing an application by pressing devices home button

The collected mobile Web logs are divided per device and application. Transactions within the logs are identified and labelled manually according to time boundary of use. We only consider URL requests directly triggered by user action to form a transaction, while the automated URL requests are discarded. The resulting labels will form a ground truth to evaluate the experiment result. We use the *Time Window* [6] approach as a baseline method. For the threshold parameter in *Time Window*, it is calculated from the average duration of transactions occurred on a device caused by an application, where duration means the time interval between the first and last Web accesses of a particular transaction in seconds.

4.2 Measurement Metrics

Accuracy and *purity* are selected to measure the performance of the proposed method. Given the clustering result C, the majority of the log entries in a cluster $c \in C$ determines which transaction it represents. If two clusters represent the same transaction, the large one is selected to calculate *accuracy*. Specifically, for each selected cluster c, we define N_c as the number of the log entries correctly identified to the corresponding represented transaction. Then, *accuracy* is defined as follows:

$$accuracy = \frac{\sum_{i=1}^{j} N_{c_i}}{\#\ of\ all\ log\ entries}, \tag{2}$$

where j denotes the number of selected clusters, representing each transaction. *Purity* is also reported here because *accuracy* is based on an assumption that one transaction will be clustered into exactly one cluster. However, a single transaction might be divided into several clusters in the clustering process, and it is acceptable as long as a cluster is not a mix of log entries from different transactions. Thus, *purity* is defined as follows:

$$purity = \frac{\sum_{i=1}^{|C|} N_{c_i}}{\sum_{i=1}^{|C|} |c_i|}, \tag{3}$$

where $|C|$ denotes the number of generated clusters, and $|c_i|$ denotes the number of log entries in cluster c_i.

4.3 Parameters

There are two parameters in the deployed DBSCAN algorithm: *MinPts* and *Epsilon*. *MinPts* is the minimum number of data points (URL log entries) within a transaction, and *Epsilon* is the maximum distance (time gap) from each data point to any other data point within the same cluster. We set *MinPts*$= 3$ as a fair number of URL requests in a single transaction. This is based on observation that a user action on an application is often triggered by more than one or two URL requests. But we are also aware that a higher number of *MinPts* might have risk of mixing few transactions together because it needs to reach *MinPts* to be considered as a cluster.

Meanwhile, *Epsilon* can be determined for each device-application by using the heuristic suggested in [7]. Specifically, the distances to the k-th nearest neighbour (k = *MinPts*) of all URL requests are sorted and plotted, as shown in Fig. 3. The corresponding distance value on the first valley or knee point in the graph indicates the threshold value for *Epsilon*. This heuristic gives an intuitive way to find a suitable *Epsilon* value, as a smaller *Epsilon* might break a transaction into several clusters. Vice versa, a bigger *Epsilon* value might group several transactions into one cluster. In this work, we use four knee point detection formulas similar to [18, 9], in order to automatically find a suitable *Epsilon* value. The formulas are modified to suit our case in which the distance distribution is monotonic and non-increasing, as illustrated in Fig. 3. Then, we compare the DBSCAN clustering performance with four knee point detection approaches, including:

$$\jmath_{i-1} - \jmath_i : difference\ between\ magnitudes \tag{4}$$

$$\frac{\jmath_{i-1} - \jmath_i}{\jmath_i} : relative\ difference\ between\ magnitudes \tag{5}$$

$$\frac{\jmath_{i-1}}{\jmath_i} - \frac{\jmath_i}{\jmath_{i+1}} : difference\ between\ ratios \tag{6}$$

$$\frac{\jmath_{i-1} - \jmath_i}{\jmath_i - \jmath_{i+1}} : difference\ between\ magnitude\ ratios \tag{7}$$

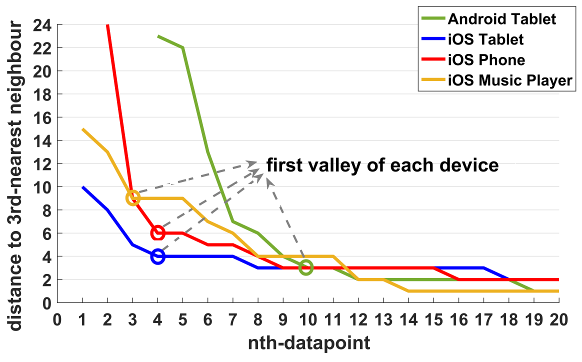

Fig. 3. Example distribution of the sorted 3rd-nearest neighbour distance

Table 3 displays the clustering performance with the four knee point detection approaches, in terms of average clustering accuracy and purity. The result indicates that the difference between magnitudes approach works best with 80.19% of average accuracy. The difference between magnitude's ratios approach gives second best performance, while the other two approaches seems to fail with average accuracy below 50%. Moreover, the four approaches gives similar performance in terms of average purity, which all of them exceed 99%. Based on this result, we use the difference between magnitudes approach to estimate *Epsilon* parameter in our experiment.

Table 3. Comparison of different knee point detection approaches to find k.

Method	$\jmath_{i-1} - \jmath_i$	$\frac{\jmath_{i-1} - \jmath_i}{\jmath_i}$	$\frac{\jmath_{i-1}}{\jmath_i} - \frac{\jmath_i}{\jmath_{i+1}}$	$\frac{\jmath_{i-1} - \jmath_i}{\jmath_i - \jmath_{i+1}}$
Description	Magnitude Difference	Relative Magnitude Difference	Ratio Difference	Magnitude's Ratio Difference
Accuracy	**80.19%**	42.32%	39.36%	72.22%
Purity	**99.27%**	99.72%	99.93%	99.52%

4.4 Experiment Results

Table 4 shows the comparison of experiment results between *clustering with DBSCAN* and *Time Window* (the baseline), in terms of accuracy. It is observed that on average, the proposed clustering-based method achieves better accuracy than the baseline. Specifically, the average accuracy of *clustering with DBSCAN* over all device-applications is 80.19%, with the highest and lowest accuracy are 100% and 51.35%, respectively. In contrast, the average accuracy of *Time Window* over all device-applications is 62.35%, with the highest and lowest accuracy are 85.00% and 50.48%, respectively. The *clustering with DBSCAN* method significantly outperforms *Time Window* in 20 out of 23 configurations, with equivalent or slight lower performance in the rest 3 configurations. The reason for the higher accuracy of the clustering-based method might be due to its flexibility in determining a transaction boundary based on the density of URL requests over time, rather than just a fixed time interval partitioning.

Table 4. Comparison on Accuracy

Device	Application	TimeWindow	DBSCAN
Android Tablet	Facebook	71.74%	**86.96%**
	Twitter	53.57%	**89.29%**
	Instagram	56.18%	**98.88%**
	Path	62.07%	**86.21%**
	MSN	**76.83%**	67.89%
	Sina	60.31%	**68.70%**
iOS Phone	Facebook	76.27%	**88.14%**
	Twitter	51.43%	**87.14%**
	Instagram	54.55%	**90.91%**
	Path	71.76%	**81.18%**
	MSN	65.50%	**89.96%**
	Sina	57.45%	**100%**
iOS Tablet	Facebook	**85.00%**	**85.00%**
	Twitter	60.68%	**99.15%**
	Instagram	56.06%	**56.92%**
	Path	50.48%	**51.43%**
	MSN	76.83%	**98.35%**
	Sina	55.48%	**81.51%**
iOS Music Player	Facebook	56.25%	**71.88%**
	Twitter	61.54%	**65.38%**
	Instagram	**51.35%**	**51.35%**
	Path	57.32%	**73.17%**
	MSN	N/A	N/A
	Sina	74.67%	**74.93%**
Average Accuracy		62.35%	**80.19%**

Moreover, we also compare the two methods in terms of *purity*. It is observed that the *purity* is similar between the two methods, with *clustering with DBSCAN* performing slightly better than *Time Window*. Specifically, the average purity of *clustering with DBSCAN* over all device-applications is 99.27%; and the average purity of *Time Window* over all device-applications is 97.84%. In addition, the surprisingly good performance of *Time Window* approach in purity might be caused by our method in choosing suitable time interval for each device-application pair based on adequate prior knowledge or good estimation of average transaction duration, which would not be available in a real life situation.

It is observed that generally clustering *purity* is higher than *accuracy*. In this case, this phenomenon mainly because a single transaction often split into several clusters, which will penalised accuracy as much as mixing different transactions into one cluster. Thus, although each cluster tends to be quite pure as it consists of URLs within a single transaction, the accuracy will seem to be relative low. Nonetheless, high purity shows that inter-transaction time gap is quite significant to distinguish one transaction from another, while future work should also be addressed to recognize the substantial variation in intra-transaction time gap.

Overall, the experiment results indicate that it is possible to solve the challenges in the mobile Web logs, and so as to identify transactions from mobile Web logs by modelling the temporal gaps (via DBSCAN). Note, this is different to previous approaches, as they relied on the full URL information or Web content without considering time gap pattern.

5 Conclusion

The paper introduces the concept of a *transaction* to determine in-app user actions from encrypted and noisy mobile Web logs. A clustering algorithm is deployed to examine the possibility of identifying these transactions by only using time stamp information and without the assumption of knowing Web content, which is infeasible due to Web traffic encryption. The experiment results indicate that the proposed method can identify transactions successfully. In the future, we plan to improve the model to distinguish different user actions in a particular app (e.g. browsing on Facebook).

Acknowledgments

This research is supported by LPDP (Indonesia Endowment Fund for Education) and a Linkage Project grant of the Australian Research Council (LP120200413).

References

1. Arora, D., Neville, S.W., Li, K.F.: Mining wifi data for business intelligence. In: 2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. pp. 394–398 (Oct 2013). https://doi.org/10.1109/3PGCIC.2013.67
2. Buehrer, G., Stokes, J.W., Chellapilla, K.: A large-scale study of automated web search traffic. In: Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web. pp. 1–8. AIRWeb '08 (2008)
3. Bulut, E., Szymanski, B.K.: Understanding user behavior via mobile data analysis. In: 2015 IEEE International Conference on Communication Workshop (ICCW). pp. 1563–1568 (June 2015). https://doi.org/10.1109/ICCW.2015.7247402
4. Chen, L., Li, Z., Ju, S.: Based on forward reference object transaction identification algorithm on web mining. In: ALPIT 2007. pp. 469–473 (Aug 2007)
5. Cooley, R., Mobasher, B., Srivastava, J.: Grouping web page references into transactions for mining world wide web browsing patterns. In: Proceedings 1997 IEEE Knowledge and Data Engineering Exchange Workshop. pp. 2–9 (Nov 1997). https://doi.org/10.1109/KDEX.1997.629824
6. Cooley, R., Mobasher, B., Srivastava, J.: Data preparation for mining world wide web browsing patterns. KIS **1**(1), 5–32 (1999)

7. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. pp. 226–231. AAAI Press (1996)
8. Fan, Y.C., Chen, Y.C., Tung, K.C., Wu, K.C., Chen, A.L.P.: A framework for enabling user preference profiling through wi-fi logs. IEEE Transactions on Knowledge and Data Engineering **28**(3), 592–603 (March 2016). https://doi.org/10.1109/TKDE.2015.2489657
9. Foss, A., Wang, W., Zaane, O.R.: A non-parametric approach to web log analysis (2001)
10. Gu, Y., Quan, L., Ren, F.: Wifi-assisted human activity recognition. In: 2014 IEEE Asia Pacific Conference on Wireless and Mobile. pp. 60–65 (Aug 2014). https://doi.org/10.1109/APWiMob.2014.6920266
11. Guerbas, A., Addam, O., Zaarour, O., Nagi, M., Elhajj, A., Ridley, M., Alhajj, R.: Effective web log mining and online navigational pattern prediction. Knowledge-Based Systems **49**(Supplement C), 50 – 62 (2013). https://doi.org/https://doi.org/10.1016/j.knosys.2013.04.014, http://www.sciencedirect.com/science/article/pii/S0950705113001263
12. Huang, J., Xu, F., Lin, Y., Li, Y.: On the understanding of interdependency of mobile app usage. In: 2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS). pp. 471–475 (Oct 2017). https://doi.org/10.1109/MASS.2017.89
13. Lee, J., Seeling, P.: An overview of mobile device network traffic and network interface usage patterns. In: IEEE EIT 2013. pp. 1–5 (May 2013)
14. Li, Y., q. Feng, B.: The construction of transactions for web usage mining. In: CINC 2009. vol. 1, pp. 121–124 (June 2009)
15. Mead, S., Veeramachaneni, N., Seeling, P.: An overview of mobile device network activities: Characteristics of heterogeneous network interfaces. In: CCNC 2016. pp. 305–306 (Jan 2016)
16. Morichetta, A., Bocchi, E., Metwalley, H., Mellia, M.: Clue: Clustering for mining web urls. In: 2016 28th International Teletraffic Congress (ITC 28). vol. 01, pp. 286–294 (Sept 2016). https://doi.org/10.1109/ITC-28.2016.146
17. Qian, F., Wang, Z., Gao, Y., Huang, J., Gerber, A., Mao, Z., Sen, S., Spatscheck, O.: Periodic transfers in mobile applications: Network-wide origin, impact, and optimization. In: Proceedings of the 21st International Conference on World Wide Web. pp. 51–60. WWW '12, ACM, New York, NY, USA (2012). https://doi.org/10.1145/2187836.2187844, http://doi.acm.org/10.1145/2187836.2187844
18. Sadri, A., Ren, Y., Salim, F.D.: Information gain-based metric for recognizing transitions in human activities. Pervasive and Mobile Computing **38**(Part 1), 92 – 109 (2017). https://doi.org/https://doi.org/10.1016/j.pmcj.2017.01.003, http://www.sciencedirect.com/science/article/pii/S1574119217300081
19. Shu-yue, M., Wen-cai, L., Shuo, W.: The study on the preprocessing in web log mining. In: KAM 2011. pp. 315–317 (Oct 2011)
20. Suadaa, L.H.: A survey on web usage mining techniques and applications. In: 2014 International Conference on Information Technology Systems and Innovation (ICITSI). pp. 39–43 (Nov 2014). https://doi.org/10.1109/ICITSI.2014.7048235
21. Woon, Y.K., Ng, W.K., Lim, E.P.: Online and incremental mining of separately-grouped web access logs. In: Proceedings of the Third International Conference on Web Information Systems Engineering, 2002. WISE 2002. pp. 53–62 (Dec 2002). https://doi.org/10.1109/WISE.2002.1181643