2017 IEEE/ACM 39th IEEE International Conference on Software Engineering Companion

IntelliAd: Assisting Mobile App Developers in Measuring Ad Costs Automatically

Cuiyun Gao*†, Yichuan Man¶, Hui Xu*†, Jieming Zhu†, Yangfan Zhou‡§, and Michael R. Lyu*†
*Shenzhen Research Institute, The Chinese University of Hong Kong, China
†Dept. of Computer Science and Engineering, The Chinese University of Hong Kong, China
‡School of Computer Science, Fudan University, Shanghai, China
§Engineering Research Center of Cyber Security Auditing and Monitoring, Ministry of Education, China
¶Beijing Jiaotong University, China

Abstract—**In-app mobile advertising serves as a primary source of revenue for most free apps. Such apps are embedded with third-party SDKs for ads displaying and are monetized by user impressions. However, ad placement can sometimes spoil user experience, for example, by too much memory consumption and battery drainage, thus leading to app uninstalling and unfavorable user feedback. Therefore, ensuring user perceptions of mobile ads can be greatly beneficial to app developers. Furthermore, various ad networks and formats make ads selection a great challenge. To achieve this, we design a tool named IntelliAd to automatically measure the ads-related consumption on mobile phones. Based on the measured costs, developers can optimize the ad-embedding schemes for their apps.**

***Keywords*-Mobile Advertising; Ad Costs; Automatic Measurement;**

I. Introduction

In-app advertising is a form of advertising within apps and on mobile devices such as phones and tablets. To embed ads into apps, developers are required to introduce third-party advertising SDKs and determine the ad networks to use (*e.g.*, AdMob [2] or MoPub [5]). Moreover, mobile ads must be defined as text, graphics or video based messages and in fixed sizes (*e.g.*, banner or interstitial). Ads rendering involves requesting and fetching ads to display from the ad networks. Successful renderings, *i.e.*, user impressions, can benefit app developers.

However, free app users may incur hidden costs, for example, using much traffic usage for data transmission and battery drainage for ads displaying. Such ad costs can spoil user experience, resulting in customer loss and finally profit reduction [8]. However, this does not mean that ads are definitely hated by all the users and should be avoided for free apps. According to a recent survey, 83% respondents stated that "Not all ads are bad, but I want to filter out the really obnoxious ones" [7]. Therefore, if we incorporate ads into apps appropriately, the user experience can still be ensured.

In the paper, we classify ad costs into four types, including memory consumption, CPU overhead, number of threads, and traffic usage. Previous research mainly focuses on mitigating the ad costs from the system side [11], leaving less flexibility for app developers to modify or customize. In our work, developers just need to embed the optional ads into apps, and then the corresponding costs can be profiled and measured by running our tool IntelliAd. Based on the measured costs, developers can determine the optimal ad schemes. This process can be handled by developers themselves and therefore provide more flexibility. The experimental results verify the effectiveness of our work.

II. The Approach

In the section, we introduce the proposed measuring tool IntelliAd. The metrics to measure include the memory consumption, CPU overhead, number of threads, and traffic usage during the app runtime.

Ad Display Control: To automatically measure the costs, we employ the dynamic analysis tool AppsPlayground [10]. We have modified AppsPlayground to accurately measure the costs of displaying ads. First, the execution rules have been adjusted. We confined the UI widgets for ads (*i.e.*, widgets ending with "View", "WebView", or "FrameLayout" in the class names) to be rendered only, not clicked. Moreover, to ensure equal displaying durations, the interval between two operations (*e.g.*, clicks) are fixed, such as 20 seconds.

Memory Consumption: Memory management in Android enables the system to allocate the precious resource. Much consumption on the resources leads users to experience lag. When the memory becomes constrained, the system slows down dramatically [9]. IntelliAd employs the tool *top* to monitor the memory consumption, measured by "Resident Set Size".

CPU Overhead: The overall busyness of the system can be quantified by the CPU utilization [4]. We suppose that different ad SDKs manage the ad lifecycle differently, and observe their CPU expended during the runtime. Also, *top* is utilized to record the CPU occupancy rate.

Number of Threads: Rendering ads in the user's interface involves the implementation of the UI thread. When the UI thread(*i.e.*, "main thread") performs intensive work, apps may appear to hang. Therefore, we also consider the number of threads as a type of ad costs. The cost is evaluated through reading the */proc/pid/stat* file at the runtime.

Traffic Usage: Different ad networks and ad formats (*e.g.*, banner, interstitial, or video) can influence the data transmitted. IntelliAd utilizes the typical tool *tcpdump* [6] to estimate the total bytes consumed in real time.

978-1-5386-1589-8/17 $31.00 © 2017 IEEE
DOI 10.1109/ICSE-C.2017.123

TABLE I
AD SCHEMES MEASURED

ID	Ad Network	Banner	Interstitial
A1	AdMob	✓	
A2	MoPub	✓	
A3	AdMob		✓
A4	AdMob	✓	✓

Measuring Frequency Setting: The *top* tool is set to run at one second interval to measure the memory/CPU overhead. IntelliAd reads the system files every 0.04s to capture the number of threads. Besides, to record all the traffic transmitted, *tcpdump* is started once the app is launched.

III. EXPERIMENTS

In the section, we have measured the ad costs for different ad schemes, which evaluates the influence of schemes on ad costs and also the effectiveness of IntelliAd.

A. Experimental Settings

In the experiment, we employ two leading ad networks, *i.e.*, AdMob and MoPub, which occupy over half of the whole android advertising market (68.0%) [1]. The ad formats are defined as images and in two types of size (banner and interstitial). Table I depicts the four ad schemes we have experimented with. We choose the schemes for analyzing the effects of different ad networks (A1 and A2), or formats (A1 and A3), or the numbers of ads (A1 and A4).

Ad Cost Separation: Since only the costs produced by ads are required, we create a prototype app for analysis. The costs of each scheme are calculated by subtracting the costs of the prototype app from those of the ad-embedded app.

One Page, One Ad: According to the mobile advertising policies [3], the number of ads on a single screen should not exceed one. Therefore, we design one button in the main activity for navigating to an empty activity or interstitial ad accordingly.

Mitigating Background Noise: To mitigate the background noise, we restore the system environment to its original state in the beginning of each measurement. Furthermore, the costs of each ad schemes are measured for four times and the average results are calculated for analysis.

B. Experimental Results

The mobile device we have used is the LG Nexus 5 smartphone with a rooted Android 5.0.1 operating system. Figure 1 illustrates the increase rates for the corresponding ad costs.

As Fig. 1 indicates, the ad costs for different ad schemes exhibit obvious differences. The average increase rates are 1.12, 0.11, 2.52, and 8.84 times for memory consumed, CPU overhead, the number of threads, and traffic usage, respectively, with the standard deviations at 0.17, 0.13, 0.24, and 4.24. We can discover that the traffic usage displays the highest increase rate among all the cost types, and also largest fluctuation along with ad schemes. This implies that ad schemes can affect traffic consumption greatly. We then analyze the effects of ad networks and formats on the generate costs.

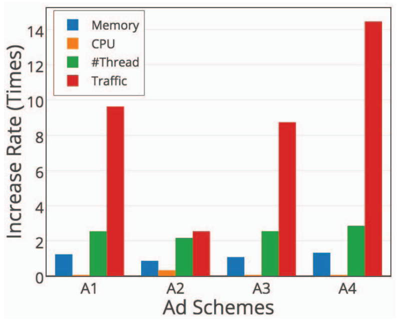

Fig. 1. Increase Rates of Ad Costs for Different Ad Schemes.

(1) **Ad Network:** According to Figure 1, despite embedded with the same ad format (banner), A1 (AdMob) presents nearly 2.8 times more than A2 (MoPub) regarding the traffic growth, which is also verified in [11]. However, for the CPU consumed, A2 displays 13.3 times more increase than A1. Therefore, different ad networks impact ad costs differently.

(2) **Ad Format:** In Figure 1, although A1 (banner) and A3 (interstitial) are embedded with the same ad network, their ad costs are different due to ad formats. A3 shows more increase on CPU utilization (47.8%), which might be attributed to more rendering work for the interstitial ad. The differences also exhibit on the memory consumed and traffic usage. Thus, the ad format is also an influence on the ad costs.

Moreover, by comparing A1 with A4 (banner and interstitial), we can discover that more ads embedded would increase the ad costs seriously. For example, A4 presents much more consumption on the data traffic and CPU overhead than A1, 50.1% and 95.7%, respectively. Based on the experimental results, we can conclude that ad schemes can affect ad costs, which also signifies the usefulness of IntelliAd.

IV. CONCLUSIONS

We introduce a tool named IntelliAd to automatically measure and profile ad costs of different ad schemes. Based on the measured costs, developers can better determine the ads to embed into their apps.

ACKNOWLEDGMENTS

This work was supported by the Key Project of National Natural Science Foundation of China (Project No. 61332010), the National Basic Research Program of China (973 Project No. 2014CB347701), the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14205214 of the General Research Fund), and 2015 Microsoft Research Asia Collaborative Research Program (Project No. FY16-RES-THEME-005). Yangfan Zhou is the corresponding author.

REFERENCES

[1] Ad libraries provided by AppBrain. http://www.appbrain.com/stats/libraries/ad.

[2] AdMob. https://www.google.com/admob/.

[3] Behavioural policies. https://support.google.com/admob/answer/2753860?hl=en-GB.

[4] CPU time. https://en.wikipedia.org/wiki/CPU_time.

[5] MoPub. http://www.mopub.com/.

[6] Tcpdump.

[7] User survey. https://www.vieodesign.com/blog/new-data-why-people-hate-ads.

[8] Why do people uninstall apps? https://www.quora.com/Why-do-people-uninstall-the-apps.

[9] Why does the iPhone need so much less RAM than Android devices? https://www.quora.com/.

[10] V. Rastogi, Y. Chen, and W. Enck. Appsplayground: automatic security analysis of smartphone applications. In *Proceedings of the 3rd Conference on Data and Application Security and Privacy (CODASPY)*, pages 209–220. ACM, 2013.

[11] N. Vallina-Rodriguez, J. Shah, A. Finamore, Y. Grunenberger, K. Papagiannaki, H. Haddadi, and J. Crowcroft. Breaking for commercials: characterizing mobile advertising. In *Proceedings of the 2012 Conference on Internet Measurement Conference (IMC)*, pages 343–356. ACM, 2012.