Data Mining Approach to Filter Click-spam in Mobile Ad Networks

Dr.D.Vasumati, M. Sree Vani, Dr.R.Bhramaramba, and O.Yaswanth Babu

Abstract**—**The mobility revolution introduced by the smart phones has created new advertising channels for e-businesses. However, these new channels are exploited by the unethical ad-publishers to practice click spam in fraudulent ways affecting the mobile end users in terms of time and money. Ad networks are also experiencing great loss in their revenues under utilizing the computing resources. This emphasizes the identifying and detection of click-spam leveraging appropriate data mining techniques.

Mobile apps play a vital role to attract mobile advertising. Popular apps can generate millions of dollars in profit. The presence of spam in mobile advertising is still growing under the hood though ad-networks have been taking enough security measures. Hence, it has become a great significance to solve click spam issues for the smooth development of the mobile revolution.

In this research work, we discussed the possibilities of click spam in mobile advertising by experimenting various data mining classification algorithms over a novel set of features from publisher's apps, App developers, Ad-control location, User interest ratings. A Spam Rank algorithm was developed further that reduce the revenue losses when grouped the ad publishers based on the computed degree of the spam rank. We validated our methodology using data sets of FDMA 2012. BuzzCity provides a snapshot of their click and publisher database. Our approach flags publisher as FRAUD for discounting the click. It provided a good performance on ROC and precision-recall curves.

Keywords**—** Advertisement networks, Click-Spam, Data mining, Mobile Apps

I. Introduction

A typical mobile advertising system has seven participants: mobile-user, mobile-app, e-advertiser, ad-publisher, ad-server, ad-exchange and Ad-network.

1. Mobile-user: The person who uses the mobile app.
2. Mobile-app: The running program on a mobile operating system.
3. e-Advertiser: The person who represent a company for promoting the unique products/services.
4. Ad-publisher: The person who run ads through the ad-control module on the mobile apps.
5. Ad-server: The server that cater ads from the centralized repository.
6. Ad-exchange: The server that aggregate ads from different ad-networks
7. Ad-network: The Ad agency that collect, store and market Ads for the e-Advertisers

IMobile advertisements that run in the apps are the primary source of income for the mobile ad-publishers. The ad-publisher earns money when the mobile users click on the published ads. The mobile users will click on these ads having interest on the products/services offered by the ad-advertiser.

The publisher's revenue gets calculated based on the number of clicks per ad performed by the mobile users. This practice has been exaggerated by some dishonest publishers to generate ad clicks unethically, either by employing people or deploying ClickBots [16] or running program scripts that simulate human click behavior. This kind of fraudulent activity is called click fraud which is a serious threat to the pay-per-click advertising market.

The below Fig 1 depicts the sequence flow of an ad-click event. The ad-publishers can use a client/server or web programming model by using JavaScript, AJAX, PHP, J2EE, RhoMobile etc technologies to embed the Ad-control in their mobile app. This app will be made available in mobile app stores for users to download.

When Mobile users download and run the App, the Ad-publishers send the Ad-slot information and request the Ad-server for latest Ads. The Ad-server makes a log entry for the same and pass the ad request to the Ad-exchange.

When Ad-exchange receives such requests, it will call for an auction among the competitive Ad-networks. The winner of this auction will return Ad(s) information that contains e-advertiser's URL, Ad display parameters and media type to the Ad-server with a request to store and forward to the mobile users.

If a mobile user clicks on an Ad in the app, that request will first go to the Ad-publisher who will log the ad-click for accounting purposes. The same request will also be passed to the Ad-Network for its future verification and business use.

The Ad-publisher will then contact Ad-server for the actual Ad URL. When Ad-server sends the Ad URL, the same will be sent to the ad-slot which in turn redirects user to the actual e-advertiser website.

Dr.D.Vasumati is with the JNTUCEH, Professor, Dept of CSE, JNTUH. Hyd-500032, INDIA (phone:9989184685;
e-mail:vasukumar_devara@jntuh.ac.in).

M.Sreevani is with the MGIT, Associate Professor, Dept of CSE, Hyd-500075, INDIA (phone:8790018470; e-mail: osreevani@gmail.com)

Dr.R.Bhramaramba is with GITAM University, Associate Professor, Dept of IT, GITAM University, Vizag -530045, INDIA (phone:9866227483, email:bhramrambaravi@gmail.com)

O.Yaswanth Babu is with TATA CONSULTANCY SERVICES, IT Manger, Gachibowli, Hyd-500032, INDIA (phone: 8179486490; e-mail: oyaswanth@gmail.com).

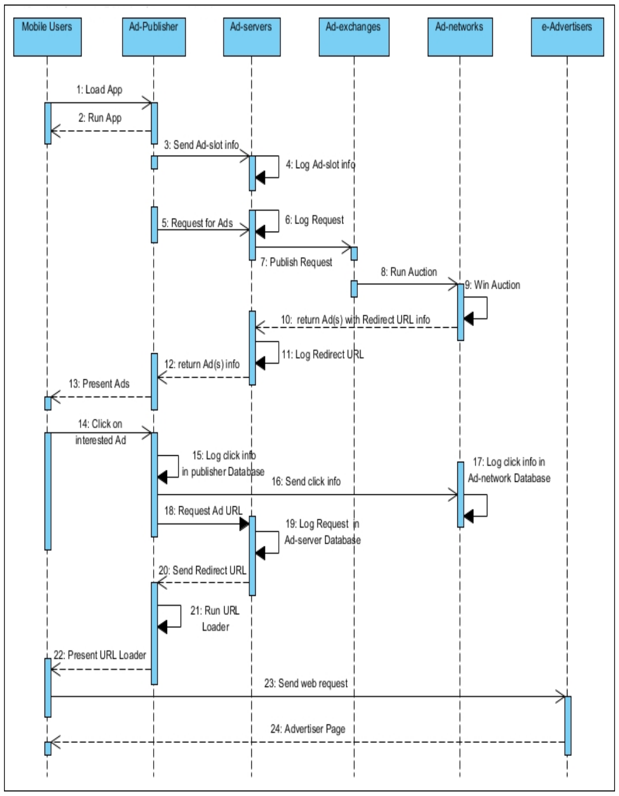


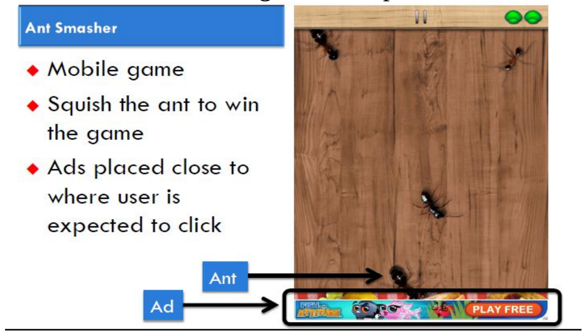
Fig. 1 Anatomy of an Ad click

A. Background and Motivation for spam in mobile advertising:

A mobile developer intentionally or accidentally keeps the in-app advertising control near to where the user must touch, or scroll on usage of smart phone. With the given micro screen real-estate, the user will be lead to mistap while working on the mobile app as shown in Fig 2.

This results in the browser to navigate unwantedly to the ad-click URL. The user might aware of his error and can switch back to his application. In the mean time the browser has already start fetching the landing-page of clicked ad and aborts the attempt. As a consequence, it appears the user had spent less time on advertiser's page. In our studies, it is understood that 95% of users spent less than a second as shown in Fig 2.The more notable thing here is that the advertiser has to pay to the ad networks even though the user spends very less time in the landing page. For an ad network it is difficult to calculate user spent time on the advertiser's landing page. If it depends on the advertiser, there is a chance for advertisers to depend on this information to get a discount. It has become a challenging problem to derive a solution for this situation by not modifying the browser and by not hurting the user experience. Auditing Apps that make users to mistap on the ad looks like a good approach, but this task may lit an arms race for apps who wants to take the advantage of above said drawback. Though it can protect advertisers from fraud Apps, it is hard for advertisers and other independent third-parties to detect spam web pages or bad Apps.

Fraudulent publishers can cost billions of dollars to the advertising companies. Their unethical activities are creating a great dissatisfaction among the other publishers


Ant Smasher

- Mobile game
- Squish the ant to win the game
- Ads placed close to where user is expected to click

Fig. 2 Illustation of click spam

The legitimate publishers are losing their trust on the online advertising system. Identifying the fraudulent behavioral patterns of unethical publishers and/or users is extremely challenging. Due to the limitations of mobile phone networks and advertising practices, it has become necessary to find new methods to evaluate the fraudulent behavior of ad publishers and new features using existing parameters to capture the behavior of the publishers. Though the fraud publishers act very cautiously when implementing fraudulent clicking behavior, these fraudulent click patterns will still deviate from the genuine click patterns of legitimate publishers.

Identifying fraudulent users become harder as these users are small in number when compared with all other legitimate users. The common classification algorithms will tend to produce more errors when the class distribution is imbalanced. Furthermore, these errors can be higher with the minority class which is the crucial class in fraud detection. It is more important to identify a fraudulent user as a fraudulent user rather identifies a legitimate user as a legitimate user. Since any legitimate user can prove self authenticity, categorizing a legitimate user as a fraudulent user can be accepted rather categorizing a fraudulent user as a legitimate user which costs more. Thus, it has become a necessary to develop new methods for identifying fraudulent behaviors to control the unethical practices in e-Advertising businesses.

II. Literature Of Survey

Despite evolutionary developments in the field of mobile technology in the form of smart phones and mobile devices and associated problems of click-spam, it continues to be virgin ground for research. There are hardly any studies exploring the causes and attempt to check the click-spam in the large interests of all the stack-holders in the field-the advertisers, the publishers and the end users. In this context an attempt is made to present an overview of the work done in the area under four broad categories. 1) Ad-networks and click-spam 2)Characterizing click-spam 3)Detection of click-spam 4)Data mining techniques for click-spam detection.

A. Ad networks and click-spam

Ad network click spam related issues are relatively unexplored area in the field of data mining [11], [14]. Studies in this area observed that click-spam and Ad networks resulted in reduced revenue of interest(ROI) due to diversion by cheaters [19], [20]. In July 2006, there is a lawsuit settlement between advertisers and Google, on Google's click-spam filtering system[10]. In a recent study on the subject Chia and others have analyzed every ad click from ad networks log data and several active filters have been used to filter each fraud click and catching signature of each specific attack [12]. In 2012 VachaDev and others[5] developed a filter to catch click-spam in an ad network.

B. *Characterizing Click-spam*

To describe the behavior of click-spam, several researchers elaborated their studies on specific attacks [1, 2, 20, 19]. Research was also done on traffic quality based on purchased traffic [12][13]. The measurement and fingerprinting study in [4] found that users are tricked into clicking on ads by using bot and non-bot mechanisms to lead generation of click-spam. The active measurement technique is used in bluff ads [10] to find behavior of click-spam. The passive measurement technique is used in [5] to catch the click-spam.

C. *Click-spam detection*

Many researchers mainly focused on (on early generation) bots to detect the click-spam. When Sbotminer [11] looked for the anomalies in query distribution, he found search engine bots. Millie[19] and Gillberg[18] reported the unusual collusion in users' associated with different publishers that may be pointing to bot behavior. User-Driven Access Control [12], Bluff ads [10] and Premium Clicks [13], aim to authenticate user presence to mitigate click-spam. More general approach is proposed by Vachadev[5], to target every kind of click-spam including bot and non-bot mechanisms

D. *Data mining approaches for click-spam detection*

The traditional data mining techniques for ad response prediction were categorized into two groups namely Maximum likelihood based and feature-based [11]. In feature-based data mining algorithms, prediction models are formulated based on explicit features of an ad and/or an page. These features may include viewable content of an ad, its location on the (web)page, etc. Typically, prediction models from logistic regression family [8, 15] are used by feature-based methods. However, implementing these models requires a lot of manual intervention or domain knowledge. The data mining techniques used by the researchers in this area are decision trees, randomforests etc [11][12].

III. METHODOLOGY

Our methodology contains offline and online phases. The offline component contains three steps. Classification of publishers as spam or not based on constructed feature vector will be undergone in step 1. Computation of spam scores for each spam publishers by constructing bi-partite graph between users and publishers will be take place in step 2. Assigning OK, OBSERVATION and FRAUD flags to spam publishers based on threshold value i.e. true negative value will be taken place in step 3.

A. *Offline Component*

The overall framework for offline component is given in Fig 3. First, we extract the App features from Apps. Given the information of App, we extract App developer features. These features traditionally used in previous work for spam detection in IOS App store [6].

We propose a novel set of features particular to the mobile advertisement-control location based, e.g. Ad-control is located at underneath buttons or any other object which users may accidentally click while interacting with your application or users will randomly click or place their fingers on the screen. These features are also defined Using external resources collected from Mobile game user experiences. Finally, given a feature vector for each App, we transform the spam detection problem into a classification problem for which we can use many well established tools and techniques to solve. Like some previous works in detecting spam content on web, we apply a decision tree classifier i.e randomforest classification algorithm to classify the Apps into spam or non-

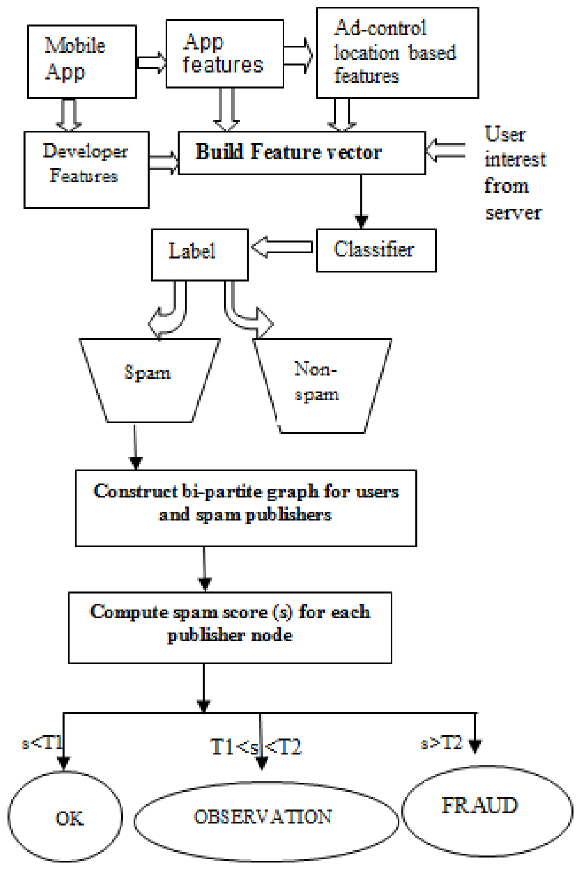

Mobile App
App features
Ad-control location based features
Developer Features
Build Feature vector
User interest from server
Label
Classifier
Spam
Non-spam
Construct bi-partite graph for users and spam publishers
Compute spam score (s) for each publisher node
s<T1
T1<s<T2
s>T2
OK
OBSERVATION
FRAUD

Fig. 3 framework for offline component

spam category On WEKA tool, several classification algorithms namely FTtree, RandomForest, REPtree, LADtree, BayesNet etc are tested on newly extracted feature set. The RandomForest algorithm is giving good average precision.

Next attempt is made to construct a Bi-partite graph between users and spammed publishers. Weighted Edges presents w_{ij} which is the revenue generated by user i to publisher j. Later a spam rank was assigned to each likely to be fraudulent publisher to know the intensity of fraud based on the set of mutual dependency principles. This iterative procedure guarantees that spam score will remain for a certain number of iterations. Each publisher was sorted based on spam rank and stores in an array of size N. Finally, the point-wise difference between the publisher's array and the baseline array for each publisher was computed. Given a threshold $\tau 1$ and $\tau 2$(which characterizes the width of the band around the baseline), if the click-spam score is less than or equal to $N\tau 1$ the publisher is flagged as OK, and if the click-spam score is in between $N\tau 1$ and $N\tau 2$ then the publisher is flagged as OBSERVATION, and if the click-spam score is greater than $N\tau 2$ the publisher is flagged as FRAUD

B. Online Component

The process of online component as described and shown in the below Fig 4. When the user clicks an Ad-control button, click event will be logged by Ad-server database. The click database contains click identification, user identification, user phone model, authentication of an advertisement and user region and user time zone. From click record publisher identification (PID) will be extracted. Then search for PID in Fraud publisher group .If PID is present in that group then filter the click and give discount for user click to Advertiser. Algorithm 3.1 to detect fraud click in online component is given below.

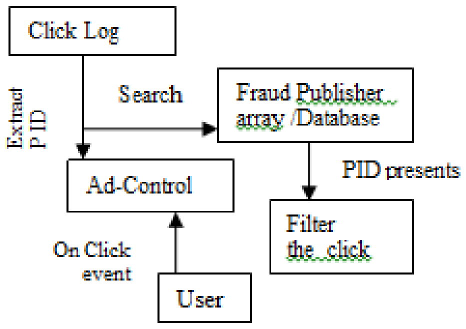


Fig. 4 Process of filtering the click

IV. Experimental Configuration

A. Dataset Description

For FDMA competition BUZZCITY ad network provided two separate databases in CSV format: publisher database and click database. These are the databases for our work.

The training dataset consists of 3,173,834 clicks from 3,081 publishers with status from 9/2/2012 to 11/2/2012. The validation dataset consists of 2,689,005 clicks from 3,064 publishers with status from 23/2/2012 to 25/2/2012. The test dataset consists of 2,598,815 clicks from 3,000 publishers with status from 8/3/2012 to 11/3/2012. The dataset characteristics are given in Table I. The publisher database records the publishers profile. The example of publisher data is shown in Table II. Only training data have status, the status of validation and test data are withheld by the organizers.

TABLE I
DATASET CHARACTERISTICS

Dataset	# of publishers	# of total clicks	Time window
Training	3,081	3,173,834	9th -11th /2/12
Validation	3,081	2,689,005	23rd to 25th /2/2012
Test	3,000	2,598,815	8th to11th march 2012

TABLE II
DATA FOR PUBLISHER'S DETAILS

Partner id	Bank account	Address	Status
8jcuw	?	2bw2ihs0ygkkwog4	OK
8jcru	5498xi9fdu040ogk	325gub2zocgks84	OK
8jciv	?	48v6lbinzwaocsok	OK

where partnerid is unique identifier of a publisher; Bankaccount is Bank account associated with a publisher (may be empty). Address is mailing address of a publisher. Status is label of a publisher, which can be "OK", "Observation" or "Fraud". The label of publishers is identified by experts. On the other hand, the click database records the click traffics. The example data for the click log is shown in Table III.

Where id is unique identifier of a particular click, iplong is public IP address of a clicker/visitor, agent is phone model used by a clicker/visitor; cid is unique identifier of a given advertisement campaign, cntr is Country from which the surfer is, timeat is Timestamp of a given click (in YYYY-MM-DD format) category is Publisher's channel type and referrer is URL where the ad banners were clicked (obfuscated; may be empty). The aim of our research work is filtering spam click by identifying fraudulent publishers. In particular, the task is to detect "Fraud" publishers (positive cases) and separate them from "OK" and "Observation" publishers (negative cases), based on their click traffic and account profiles.

TABLE III
DATA FOR PUBLISHER'S CLICK LOG

Id	Ipling	agent	Partner id	cid	Cntr	Timeat	Type	Referrer
13417861	28851801931	MAUI	8jkhl	8gp8w	tz	2012-02-09 00:00:00	mc	?
13417871	702430144	MAUI	8ihpi	8ffid	tz	2012-02-09 00:00:00	mc	?

V.Results And Discussion

Deployment of data mining web spam filter algorithm in any major ad network is possible and be evaluated online. The ad network serves ads to many publishers that cater to both general and niche audiences. The training set contained more than 3 million clicks associated with 3081 publishers, which were categorized into honest publishers (OK, 95.1%), Publisher's under scrutiny (Observation, 2.6%), and fraudulent publishers (Fraud, 2.3%). The provided training set included information about the time stamp of each click, the IP address.

TABLE IV
PERFORMANCE FOR DIFFERENT CLASSIFICATION ALGORITHMS ON NEW SET OF FEATURES

Method	Average Precision
FT Tree	36.3%
randomForest	52.3%
REPTree	35.8%
LADTree	37%
NBTree	37.9%
RandomsubSpace	38.9%
RotationForest	42.9%
BayesNet	33.7%
RPROP	48.3%

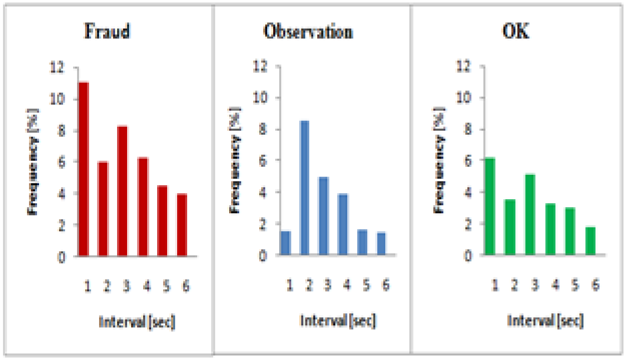


Fig. 5 Click frequency per interval based on all click profiles from the same URL

The RandomForest classification algorithm used with new set of features for classifying spam web pages showed that around used by a clicker/visitor; cid is unique identifier of a given of the computer that generated the click, the country where that computer was located, etc. The performance of the models was evaluated based on the average precision

$$AP = \frac{1}{m}\sum_{i=1}^{k} precision(i) \qquad (1)$$

On WEKA tool, we tested our feature set on several classification algorithms namely FTtree, RandomForest, REPtree, LADtree, BayesNet etc. We found RandomForest algorithm is giving good average precision.

A. SpamRank

Publishers from the training set are ranked from left to right based on spam ranks assigned based on the integral set of mutual dependency principles by constructing Bi-partite graph between users and spammed publishers. Fraudulent publishers (red) and those under observation (blue) are concentrated towards the left hand side. Thus, the larger red flag, the more suspicious is the publisher. Redflag is in fact a significant indicator of fraudulent behavior ($P < 0{:}001$, Kruskal-Wallis test). Red bars denote publishers with status Fraud; blue bars denote publishers with status Observation; white bars denote publishers with status OK. Fraudulent publishers and those under observation are significantly concentrated towards the left hand side. Fig 5 shows the click frequencies per interval, derived from all click profiles from identical URLs. Again, we observe that, overall, quick Consecutive clicks occur more often in fraudulent publishers than in those with status Observation or OK.

VI. CONCLUSION

52.3 percent of the publishers are involved in spam. Grouping of publishers based on their spam rank calculated by using Bi-partite graph and mutual dependency algorithm showed that publishers with large spam rank were grouped in the FRAUD category. Broadly, 40% were in FRAUD group, 20% OBSERVATION group consisting of those in the middle category i.e. less intensity group. The remaining percentage is in the OK group i.e. non-fraud category.

The data mining web spam filter algorithm was found to be effective in filtering the fraud clicks. It is further observed overall quick consecutive clicks occur more often in fraudulent publishers than in those within other groups i.e. OBSERVATION and OK groups.

REFERENCES

[1] M. Najork. Web spam detection. In L. Liu and M. T.□Ozsu, editors, Encyclopedia of Database Systems, pages 3520{3523. Springer US, 2009.
[2] Googleadmob. ttp://www.google.com/ads/admob/.
[3] iadappnetwork. http://developer.apple.com/support/appstore/iad-app-network/.
[4] Microsoft advertising. http://advertising.microsoft.com/en-us/splitter.
[5] S. Alrwais, A. Gerber, O. Spatscheck,M. Gupta, and E. Osterweil. Dissecting ghost clicks: Ad fraud via misdirected human clicks. ACSAC, 2012.
http://dx.doi.org/10.1145/2420950.2420954
[6] T. Blizard and N. Livic. Click-fraud monetizing malware: A survey and case study. 2012.
[7] P. Chia, Y. Yamamoto, and N. Asokan. Is this app safe? a large scale study on application permissions and risk signals. In WWW, 2012.
[8] V. Dave, S. Guha, and Y. Zhang. Measuring and fingerprinting click-spam in ad networks. In ACM SIGCOMM, 2012.
[9] C. Cadar D. Dunbar and D. Engler. Klee: In USENIX OSDI, 2008.
[10] P. Gilbert, B. Chun, J. Jung. Vision:automated security validation of mobile apps at app markets. In MCS, 2011.
[11] H. Haddadi. Fighting online click-fraud using bluff ads. ACM Computer Communication Review, 40(2):21–25, 2010.14
[12] C. Hu and I. Neamtiu. Automating gui testing for android applications. In AST, 2011.
[13] A. MacHiry, R. Tahiliani, and M. Naik. Dynodroid: An input generation system for android apps. In FSE, 2013.
[14] A. Mesbah and A. van Deursen. Invariant-based automatic testing of ajax user interfaces. In ICSE, 2009.
[15] Ali Mesbah, Arie van Deursen, and Stefan Lenselink. Crawling ajax-based web applications through dynamic analysis of user interface state changes. ACM Transactions on the Web, 6(1):1–30, 2012.
http://dx.doi.org/10.1145/2109205.2109208
[16] A. Metwally, D. Agrawal, and A. El Abbadi. Detectives:Detecting coalition hit inflation attacks in advertising networks streams. In WWW, 2007.
[17] A. Metwally F. Emekci, D. Agrawal, and A. El Abbadi.Sleuth: Single-publisher attack detection using correlation hunting. In PVLDB, 2008.
[18] B. Miller, P. Pearce, C. Grier, C. Kreibich, and V. Paxson. What's clicking what? techniques and innovations of today's clickbots. In DIMVA, 2011.
[19] L. Ravindranath, J. Padhye, S. Agarwal, R. Mahajan,I. Appinsight: mobileapp performance monitoring inthe wild. In USENIX OSDI, 2012.
[20] W. Yang, M. Prasad, and T. Xie. A grey-box approach for automated gui-model generation of mobile applications. In FASE, 2013.