A SURVEY ON: DISCOVERY OF CATEGORY FOR MOBILE APPS

Savita Gunjal[1]
[1](Department of computer Engineering, Savtribai Phule University, adhav.savita2@gmail.com)

Abstract— *Today's age is of smart life and smart phone becomes one of important necessity of smart life. As we use different software's in personal computer system for smart work, similarly smart phone needs various mobile applications. These various applications installed in smart phone make life easier. To download various mobile application smart phone user has to visit respective play store such as Google Play Store, Apples store etc. as per operating system support of smart phone. When user visit play store then he/she is able to see the various application lists. This list is built on the basis of promotion or advertisement. User doesn't have knowledge about the application (i.e. which applications are useful or useless). So user looks at the list and downloads the applications. But sometimes it happens that the downloaded application won't work or not useful. That means it is fraud in mobile application list. To avoid this fraud, we are making application in which we are going to list the applications. To list the application first we are going to find the active period of the application named as leading session. We are also investing the three types of evidences: Ranking based evidence, Rating based evidence and Review based evidence. Using these three evidences finally we are calculating aggregation.*

Keywords— Mobile Apps; Ranking Fraud; Ranking Based; Rating Based; Review Based; Evidences Aggregating Function

1. INTRODUCTION

The number of mobile Apps has grown at a breath taking rate over the past few years. For example, as of the end of April 2016, there are more than 2.6 million Apps at Apple's App store and Google Play. The App leader board which demonstrates the chart rankings is one of the most important ways for promoting mobile Apps. A higher rank on the leader board usually leads to a huge number of downloads and million dollars in revenue. Therefore, App developers tend to explore various ways such as advertising campaigns to promote their Apps in order to have their Apps ranked as high as possible in such App leader boards. However, as a recent trend, instead of relying on traditional marketing solutions, shady App developers resort to some fraudulent means to deliberately boost their Apps and eventually manipulate the chart rankings on an App store. This is usually implemented by using so-called "boot farms" or "human water armies" to inflate the App downloads ratings and reviews in a very short time. For example, an article from Venture Beat reported that, when an App was promoted with the help of ranking manipulation, it could be propelled from number 1,800 to the top 25 in Apple's top free leader.

Mobile Apps are not generally ranked high in the leader board, but rather just in some events ranking that is fraud usually happens in leading sessions. In this manner, main target is to detect ranking fraud of mobile Apps within leading sessions. First propose an effective algorithm to distinguish the leading sessions of each App based on its historical ranking records. At that point, an algorithm developed to extract ranking based fraud evidences along with rating and review based evidences. In this way, assist two types of fraud evidences are proposed based on Apps' rating and review history. Moreover, to integrate these three types of unsupervised evidence-aggregation technique is developed which is utilized for evaluating the credibility of leading sessions from mobile Apps.

2. LITERATURE SURVEY

App ranking fraud detection was explored recently. Zhu et al. [1] used statistical hypothesis testing on app rankings to identify developers who are fraudulently trying to improve the ranking of their apps. The features used were, for example, how long an app is in the top charts and its average rating when it is in the top chart compared to its previous ratings. Ranking fraud in the mobile App market refers to fraudulentor deceptive activities which have a purpose of bumping up the Apps in the popularity list. Indeed, it becomes more and more frequent for App develops to use shady means, such as inflating their Apps' sales or posting phony App ratings, to commit ranking fraud. While the importance of preventing ranking fraud has been widely recognized, there is limited understanding and research in this area. To this end, in this paper, we provide a holistic view of ranking fraud and propose a ranking fraud detection system for mobile Apps.we investigate two types of evidences, ranking based evidences and rating based evidences, by modeling Apps' ranking and rating behaviors through statistical hypotheses tests. In addition, we propose an optimization based aggregation method to integrate all the evidences for fraud detection. Finally, we evaluate the proposed system with real-world App data collected from the Apple's App Store for a long time period. In the experiments, we validate the effectiveness of the proposed system, and show the scalability of the detection algorithm as well as some regularity of ranking fraud activities.

Chandy et al. [2] developed a model to detect fake ratings using rating-related features, such as the average rating of an app and the users' average ratings. However, the intention of ranking fraud is only one scenario in miscategorization and generally, ranking fraud happens after the publication of the apps. The Android market is the fastest growing mobile application platform. A recent report from Lookout shows that the Android Market is

growing at three times the rate of Apple's App store. However, unlike Apple's App store which may check each available application manually by software security experts, there is lack of complete app inspecting process before the applications being published on the Android market. Google takes passive mechanism that allows anyone to publish applications on the Android market. If an application is reported as malware by users, it will then be removed. The openness of the Android market attracts both benign and malicious developers. Furthermore, Android allows installing third-party applications that may increase the spread of Android malware. Android security model highly relies on permission-based mechanism. There are about 130 permissions that govern access to different resources. Whenever the user install a new app, he would be prompt to approve or reject all permissions requested by the application. However, users are with rare knowledge to determine if permissions might be harmful or not. They need extra information to help them making the correct choice.

Fu et al. [3] identified the major factors resulting lower app ratings by mining app reviews and ratings using LDA and linear regression. A plenty of apps are released to enable users to make the best use of their cell phones. Facing the large amount of apps, app retrieval and app recommendation become important, since users can easily use them to acquire their desired apps. To obtain high-quality retrieval and recommending results, it needs to obtain the precise app relationship calculating results. Unfortunately, the recent methods are conducted mostly relying on user's log or app's context, which can only detect whether two apps are downloaded, installed meanwhile or provide similar functions or not. In fact, apps contain many general relationships other than similarity, such as one app needs another app as its tool. These relationships cannot be dug via user's log or app's context. Reviews contain user's viewpoint and judgment to apps, thus they can be used to calculate relationship between apps. To use reviews, this paper proposes an iterative process by combining review similarity and app relationship together. Experimental results demonstrate that via this iterative process, relationship between apps can be calculated exactly. Furthermore, this process is improved in two aspects. One is to obtain excellent results even with weak initialization. The other is to apply matrix product to reduce running time.

Liu et al. [4] used the similarity between app reviews to identify relationships between apps such as functional similarity and complementary or dependent app pairs. The recent 10 to 20 years, smart phone becomes essential in people's daily life. They can be used not only for communication, but also for entertainment, business, and travel, etc. The popularity of smart phones causes many apps released to help users make the best use of their phones. In general, the way used to recognize and extract entities can be directly used on apps. These methods acquire considerably high performances in various scenarios. In fact, facing the situation that the amount of apps grows fast, how to de-fine their relationship is more useful. To define relation-ship between apps, or furthermore to classify relationship between apps, it needs to invent a method to calculate relationship between apps exactly at first. For example, given two apps respectively designed for Android and IOS, to determine whether they refer to two unrelated apps or are just the same app adapted to two different systems, the first step is to figure out whether these two apps have relationship or not. In general, the task of calculating relationship between apps is more valuable and challengeable. With it, we can group related apps by linking them to form a graph. With this graph, app retrieval and app recommendation are easy to be performed. The problem of app categorization is closely related to the classification problem. In the classification problem, one aims to identify a category (or a class) membership of an observation from a set of categories. The classification is done by a classifier that groups observations which share similar properties under the same category/class. Another aspect of the classification problem is novelty detection. In novelty detection, one aims to recognize observations that have different patterns from other observations. Several methods have been used as the de-facto approaches in classification and novelty detection including clustering-based, domain based methods and generative models.

Clustering-based methods group a set of observation points to several groups/clusters. The most common method to group a set of n observation points to k clusters is kmeans clustering algorithm. Each cluster is represented by a centroid or a cluster center, which is a mean of all nearest observation points within that cluster. Observation points that have large distance to the cluster center in each cluster are considered to be anomalous. The k-means is widely adopted in many different applications. Domain-based methods separate observations based on a boundary learned from the training data. The de-facto approach to domain-based novelty detection has been one class Support Vector Machines (OC-SVM) [7]. The presence of mobile devices has increased in our lives offering almost the same functionality as a personal computer. Android devices have appeared lately and, since then, the number of applications available for this operating system have increased exponentially. Google already has its Android Market where applications are offered and, as happens with every popular media, is prone to misuse. A malware writer may insert a malicious application into this market without being noticed. Indeed, there are already several cases of Android malware within the Android Market. Therefore, an approach that can automatically characterise the different types of applications can be helpful for both organising the Android Market and detecting fraudulent or malicious applications. In this paper, we propose a new method for categorising Android applications through machine-learning techniques.

OC-SVM defines a hyperplane that maximizes the separating margin between classes. Observation points that lie near the boundary are called support vectors. The bigger distance of an observation point to the defined hyperplane, the further an observation from the commonly observed behavior. OCSVM has also been studied extensively for various applications.

3. SYSTEM ARCHITECTURE

The increase in popularity of mobile apps increases the fraudulent apps and fraud rankings of mobile apps in play store. The various leader boards of different online mobile app stores contain fraud mobile ranking apps. These fraud ranking misguide the app users. Hence, it becomes necessary to discover fraudulent mobile apps. This paper proposes a simple and effective system. Fig. 1 shows the system architecture, framework of the fraud ranking discovery in mobile app.

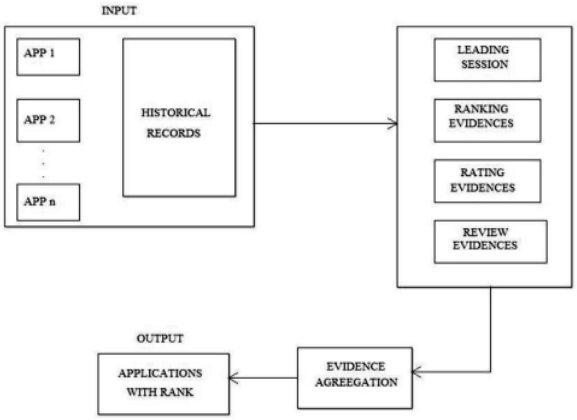

Fig. 1 System architecture, Category for Mobile Apps

An application server, its location based product that resides in the local host. The new application is adding to this server. It provides application services for customer as well as security, along with storing customer rating and feedback. User uses the applications and put the feedback for used application, this feedback is also saved on the application server. By using FRAC model we remove the malicious app and ranking the application based on the user feedback. When new user download the application that time user get notification on his window that application is malicious or not. On that basis user will understand this application is useful or not to his mobile or windows system. Reviews contains some textual comments as reviews by app user and before downloading or using the app user mostly prefer to refer the reviews given by most of the users. Therefore, although due to some previous works on review spam detection, there still issue on locating the local anomaly of reviews in leading sessions. So based on apps review behaviours, fraud evidences are used to detect the ranking fraud in Mobile app. The reviews are composite of either positive words or negative words. The aggregation and computation of words gives whether given comment if positive or negative for app under consideration. This analysis helps in finding fraudulent apps in a leader board. The proposed new effective algorithm find out fraud ranking in mobile app by aggregating results from above three functions and analysis accordingly.

4. CONCLUSION

The Advanced FRAC, framework for App Categorization, to detect malicious app. The key ideas include (i) it impairs the integrity of existing categories, (ii) it allows some app developers to get an unfair advantage over others, (iii) it makes auditing and ensuring quality/regulatory control more difficult and (iv) it might mislead users and entice them to pay for apps that do not provide the expected utility. The user blindly uses such malicious application because they attracts them with fake promises like mobile recharge, lucky winner to in dollars etc. and this applications demands user profile before they start and redirects to some other site which in unsecure for user. The system will efficiently detects and reports the malicious application based users reviews and by applying generative process of FRAC. The Advanced FRAC model stems further research directions in related to topic modeling. In this work we considered a flat topic structure for apps as it is used in the current app markets.

REFERENCES

[1] H. Zhu, H. Xiong, Y. Ge, and E. Chen, “Ranking fraud detection for mobile apps: A holistic view,” in Conference on Information and Knowledge Management (CIKM), 2013, pp. 619–628.
[2] R. Chandy and H. Gu, “Identifying spam in the iOS app store,” in Joint WICOW/AIRWeb Workshop on Web Quality, 2012, pp. 56–59.
[3] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, and N. Sadeh, “Why people hate your app: making sense of user feedback in a mobile app store,” in International Conference on Knowledge Discovery and Data Mining (SIGKDD), 2013, pp. 1276–1284.
[4] M. Liu, C. Wu, X.-N. Zhao, C.-Y. Lin, and X.-L. Wang, “App relationship calculation: An iterative process,” IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE), vol. 27, no. 8, pp. 2049–2063, 2015.
[5] Y. Zhou, H. Yu, and X. Cai, “A novel k-means algorithm for clustering and outlier detection,” in IEEE International Conference on Future Information Technology and Management Engineering, 2009, pp. 476 – 480.
[6] K.-A. Yoon, O.-S. Kwon, and D.-H. Bae, “An approach to outlier detection of software measurement data using the k-means clustering method,” in International Symposium on Empirical Software Engineering and Measurement, 2007.
[7] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, “Support vector method for novelty detection,” in Advances in Neural Information Processing Systems (NIPS), 2000, pp. 582 – 588.
[8] A. Gorla, I. Tavecchia, F. Gross, and A. Zeller, “Checking app behavior against app descriptions,” in International Conference on Software Engineering (ICSE), 2014, pp. 1025–1035.
[9] X. Wei, L. Gomez, I. Neamtiu, and M. Faloutsos, “Profiledroid: multi-layer profiling of Android applications,” in International Conference on Mobile Computing and Networking (MobiCom), 2012, pp. 137–148.
[10] B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, and P. G. Bringas, “On the automatic categorisation of android applications,” in IEEE Consumer Communications & Networking Conference (CCNC), 2012, pp. 149–153.
[11] S. Ma, S. Wang, D. Lo, R. H. Deng, and C. Sun, “Active semi supervised approach for checking app behavior against its description,” in IEEE 39th Annual International Computers, Software and Applications (COMPSAC), vol. 2. IEEE, 2015, pp. 179–184.
[12] A. Shabtai, Y. Fledel, and Y. Elovici, “Automated static code analysis for classifying Android applications using machine learning,” in Computational Intelligence Society (CIS). IEEE Computer Society, 2010, pp. 329–333.
[13] G. Berardi, A. Esuli, T. Fagni, and F. Sebastiani, “Multi-store metadata-based supervised mobile app classification,” in Proceedings of the 30th Annual ACM Symposium on Applied Computing (SAC). ACM, 2015, pp. 585–588.
[14] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty, “Latent Dirichlet Allocation,” Journal of Machine Learning Research (JMLR), vol. 3, pp. 993–1022, 2003.
[15] A. Ahmed, Y. Low, M. Aly, V. Josifovski, and A. J. Smola, “Scalable distributed inference of dynamic user interests for behavioral targeting,” in International Conference on Knowledge Discovery and Data Mining (SIGKDD), 2011.
[16] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, “Labeled LDA: a supervised topic model for credit attribution in

multilabeled corpora," in Conference on Empirical Methods in Natural Language Processing, 2009, pp. 248 – 256.

[17] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unit hypersphere using von Mises-Fisher distributions," Journal of Machine Learning Research (JMLR), vol. 6, pp. 1345–1382, 2005.

[18] D. Surian and S. Chawla, "Mining outlier participants: Insights using directional distributions in latent models," in The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD) (3), 2013, pp. 337–352.

[19] C. C. Aggarwal, S. C. Gates, and P. S. Yu, "On the merits of building categorization systems by supervised clustering," in International Conference on Knowledge Discovery and Data Mining (SIGKDD), 1999, pp. 352–356.

[20] X. Cao, G. Cong, B. Cui, C. S. Jensen, and Q. Yuan, "Approaches to exploring category information for question retrieval in community question-answer archives," ACM Transactions on Information Systems (TOIS), vol. 30, pp. 7.1–7.38, 2012