MITIGATING WEB SPAM TAXONOMY FOR MOBILE APP USING LINK PRUNING AND REWEIGHTING ALGORITHM

A.Mangalakshmi,R.Priyanka,V.Vinothini,G.Rajalakshmi
Assistant Professor,Christ College Engineering And Technology,Pondicherry,India.
Student ,Christ College Engineering And Technology,Pondicherry,India.

***Abstract*— The number of mobile Apps has grown at a breathtaking rate over the past few years. To stimulate the development of mobile Apps, many App stores launched daily App leaderboards, which demonstrate the chart rankings of most popular Apps. A higher rank on the leaderboard usually leads to a huge number of downloads and million dollars in revenue. 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 "bot farms" or "human water armies" to inflate the App downloads, ratings and reviews in a very short time. In the literature, while there are some related work, such as web ranking spam detection, online review spam detection and mobile App recommendation. The problem of detecting ranking fraud for mobile Apps is still underexplored. To fill this crucial void, in this project, we propose to develop a ranking fraud detection system for mobile Apps.**

I. INTRODUCTION

The term "spam" has been commonly used in the Internet era to refer to unsolicited (and possibly commercial) bulk messages. The most common form of electronic spam is e-mail spam, but in practice each new communication medium has created a new opportunity for sending unsolicited messages. The Web is not absent from this list, but as the request-response paradigm of the HTTP protocol makes it impossible for spammers to actually "send" pages directly to the users, Web spammers try to deceive search engines and thus break the trust that search engines establish with their users. All deceptive actions which try to increase the ranking of a page in search engines are generally referred to as Web spam or spamdexing (a portmanteau, or combination, of "spam" and "indexing"). A spam page or host is a page or host that is either used for spamming or receives a substantial amount of its score from other spam pages.

WITH the rapid development of mobile App industry, the number of mobile Apps available has exploded over the past few years. For example, as of the end of April 2013, there are more than 1.6 million Apps at Apple Appstore [2] and Google Play [1]. To facilitate the adoption of mobile Apps and understand the user experience with mobile Apps, many App stores provide the periodical (such as daily) App chart rankings and allow users to post ratings and reviews for their Apps. Indeed, such popularity information plays an important role in mobile App services [3], [5], [7], and opens a venue for mobile App understanding, trend analysis, and other related applications [4], [6].

While people have developed some specific approaches to explore the popularity information of mobile Apps for some particular tasks [7], [8], [10], the use of popularity information for mobile App services is still fragmented and under-researched. Indeed, there are two major challenges along this line. First, the popularity information of mobile Apps often varies frequently and has the instinct of sequence dependence. For example, although the daily rankings of different mobile Apps may be different, it is unlikely that an App with a high ranking will be ranked very low in the following day due to the momentum of popularity. Second, the popularity information is heterogeneous, but contains latent semantics and relationships. For example, while ranking = 1 and rating = 5 come from different observations, both of them can indicate the high popularity.

To this end, in this paper, we propose a systematic review of web spam detection techniques with the focus on algorithms and underlying principles. Link spam and Click spam both are web spam discussed in our proposed work. Link spam Adding links that point to the spammer's web site increases the page rankings for the site in the App Store. Similarly click spam, clicking ad banners without any intention of purchasing the product. Clicking the ads countless times can make dishonest rankings in Mobile App Store. Link pruning and reweighting algorithms are used here to detect and avoid the web spam. Link pruning and reweighting algorithms detect the "neponistic links", links that present for reasons rather than merit, for instance, navigational links on a website or links between pages in a link farm and also reports its resistance to fraudulent clicks.

II. RELATED WORK

Review spammer detection approach: Review spammer detection approach is user centric, and user behavior driven. A user centric approach is preferred over the review centric approach as gathering behavioral evidence of spammers is easier than that of spam reviews. A review involves only one reviewer and one product. The amount of evidence is limited. A reviewer on the other hand may have reviewed a number of products and hence has contributed a number of reviews. The likelihood of finding evidence against spammers will be much higher. The user centric approach is also scalable as one can always incorporate new

ISSN: 2278 – 1323

spamming behaviors as they emerge. The main building blocks of the spamming behavior detection step are the spamming behavior models based on different review patterns that suggest spamming. Each model assigns a numeric spamming behavior score to each reviewer by measuring the extent to which the reviewer practises spamming behavior of a certain type.

*Regression Model for Spammers:*With the labeled spammers, we now train a linear regression model to predict the number of spam votes of a given reviewer's spamming behaviors, i.e., GD, ED, TP, TG scores. To ensure that the trained regression model make as little prediction errors as possible at the highly ranked reviewers, we consider the number of spam votes in the objective function for optimizing the regression model.

*Analysis of Spammed Products and Product Groups:*To show how the products and product groups are affected by spammers, we define a spam Index for a product *oj*and a product group *gl*as:

$$s(oj) = Avgui2\text{Us}(oj)s(ui)$$

$$s(gl) = Avgui2\text{Us}(gl)s(ui)$$

where Us(*oj*) denotes the set of reviewers of *oj*each having spam scores computed by our trained regression model. Us(*gl*) denotes the set of reviewers of objects of product group *gl*.

*Spotting Opinion Spammers using Behavioral Footprints:*Opinionated social media such as product reviews are now widely used by individuals and organizations for their decision making. However, due to the reason of profit or fame, people try to game the system by opinion spamming (e.g., writing fake reviews) to promote or to demote some target products. In recent years, fake review detection has attracted significant attention from both the business and research communities. However, due to the difficulty of human labeling needed for supervised learning and evaluation, the problem remains to be highly challenging. This work proposes a novel angle to the problem by modeling spamicity as latent. An unsupervised model, called Author Spamicity Model (ASM),

Hyperparameter EM: Algorithm performs inference using uninformed priors (i.e., hyperparametersα and γ are set to Posterior estimates of spamicity can be improved if hyperparametersα and γ are estimated from the data. This is because the priors for author spamicity and latent review behaviors (α and γ) directly affect spam/non-spam cluster assignment to reviews. Algorithm 2 details hyperparameter estimation using the single sample Monte Carlo EM, which is recommended by as it is both computationally efficient and often outperforms multiple-sample Monte Carlo EM.

Learning Observed Feature Thresholds: The feature constructions contain thresholds. The thresholds can either be set heuristically or learned using some weak supervision. In this work, we use weak supervision to learn the thresholds from the Amazon group spam dataset in, which provides a small set of labeled spammer groups and their reviews. Using this small set of available data is not a limitation of our method because the actual spamicty modeling of ASM still remains unsupervised as it does not use any spamicty labels for authors/reviews in model building. In fact, ASM does not have any response variable where supervision can be fed using labels.

III. SYSTEM ANALYSIS

A. EXISTING SYSTEM

App stores launched daily App leaderboards, to stimulate the development of mobile Apps. The App leaderboard is one of the most important ways for promoting mobile Apps. A higher rank App on the leaderboard usually leads to a huge number of downloads. Therefore, App developers advertising campaigns to promote their Apps in order to have their Apps ranked as high as possible in such App leaderboards.

B. PROBLEM DEFINIITON:

- Difficult to detect the time when fraud happens.
- It is difficult to manually label ranking fraud for each App.
- Is not easy to identify and confirm ranking fraud.
- It's not detecting the web spam.

C. PROPOSED SYSTEM

Apple's App store and Google Play became a de facto place to search and download Mobile Apps Store. Though due to web spam phenomenon, search results are not always as good as desired. So, we had to detect and avoid the web spam taxonomy. In our proposed work, we present a systematic review of web spam detection techniques with the focus on algorithms and underlying principles. Link spam and Click spam both are web spam discussed in our proposed work. Link spam Adding links that point to the spammer's web site increases the page rankings for the site in the App Store. Similarly click spam, clicking ad banners without any intention of purchasing the product. Clicking the ads countless times can make dishonest rankings in Mobile App Store. Link pruning and reweighting algorithms are used here to detect and avoid the web spam. Link pruning and reweighting algorithms detect the "neponistic links", links that present for reasons rather than merit, for instance, navigational links on a website or links between pages in a link farm and also reports its resistance to fraudulent clicks.

D. ADVANTAGE

1. Automatic rating
2. Web spam detected
3. Link spam and click spam are detected
4. Providing high quality search result.

IV. SYSTEM DESIGN

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development.

ISSN: 2278 – 1323

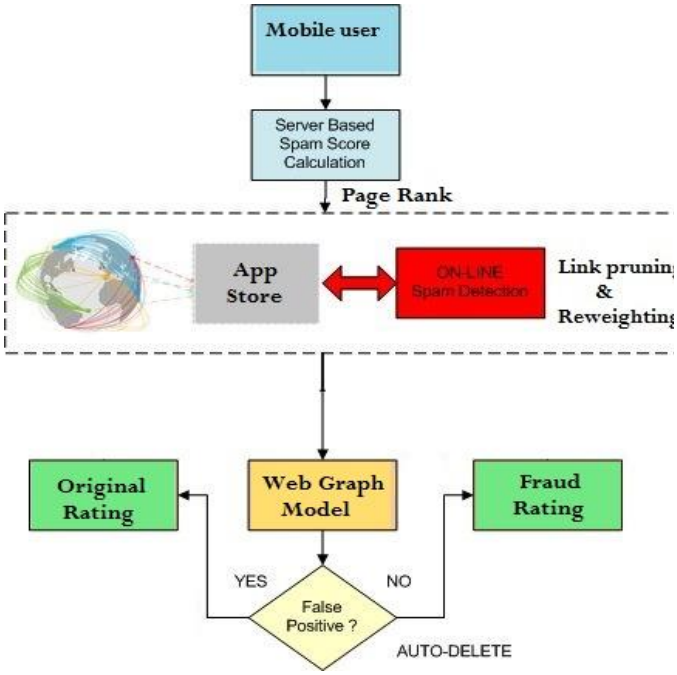

Fig: Architecture Diagram

V. ALGORITHM

A. LINK PRUNING AND REWEIGHTING ALGORITHMS

Algorithms belonging to this category tend to find unreliable links and demote them. The seminal work raises problems in HITS algorithm, such as domination of mutually reinforcing relationships and neighbour graph topic drift, and proposed methods of their solution by augmenting a link analysis with a content analysis. They propose to assign each edge an authority weight of 1 k if there are k pages from one site link to a single page on another site, and assign a hub weight of 1 l if a single page from the first site is pointing to l pages on the other site. To combat against topic drift they suggest using query expansion, by taking top-K frequent words from each initially retrieved page; and candidate page set pruning, by taking page relevance as a factor in HITS computation. a projection-based method is proposed to compute authority scores. They modify eigenvector part of HITS algorithm in the following way. Instead of computing a principal eigenvector of ATA, they compute all eigenvectors of the matrix and then take the eigenvector with the biggest projection on the root set (set of pages originally retrieved by keyword search engine, as in HITS), finally they report authority scores as the corresponding components of this eigenvector. Another group introduces the concept of tightly-knit community (TKC) and proposes SALSA algorithm, which performs two random walks to estimate authority and hub scores for pages in a subgraph initially retrieved by keywordbased search. It is worth noting that the original and an inverted subgraphs are considered to get two different scores.

An extension of this work considers clustering structure on pages and their linkage patterns to downweight bad links. The key trick is to count number of clusters pointing to a page instead of number of individual no

$$a_j = \sum_{k:\, j \in l(k)} \frac{1}{\sum_{i:\, j \in l(i)} S_{ik}},$$

small-in-large-out" problem of HITS and proposes to reweight incoming and outgoing links for pages from the root set in the following way. If there is a page whose in-degree is among the three smallest ones and whose outdegree is among the three largest ones, then set the weight 4 for all in-links of all root pages, otherwise set to 1. Run one iteration of HITS algorithm without normalization. Then if there exists a root page whose authority value is among the three smallest ones and whose hub value is among the three largest ones, set the weight 4 for all in-links of all root pages, and then run the HITS algorithm again. [11] introduces the concept of "neponistic" links – links that present for reasons rather than merit, for instance, navigational links on a website or links between pages in a link farm. They apply C4.5 algorithm to recognize neponistic links using 75 different binary features such as IsSimilarHeaders, IsSimilarHost, is number of shared in-links is greater than a threshold. Then they suggest pruning or downweightingneponistic links. In their other work they continue studying links in densely connected link farms. The algorithm operates in three stages. First, it selects a set of bad seed pages guiding by the definition that a page is bad if intersection of its incoming and outgoing neighbours is greater than a threshold. Second, it expands the set of bad pages following the idea that a page is bad if it points to alot of bad pages from the seed set. Finally, links between expanded set of bad pages are removed or downweighted and any link-based ranking algorithm can be applied. Similar ideas are studied on a host level.

VI. CONCLUSION

In this project, we developed a web spam detection system for mobile Apps. To draw a general picture of the web spam phenomenon, we first provide numeric estimates of spam on the Web, discuss how spam affects users rating for mobile apps, and motivate academic research. In our project, we present a systematic review of web spam detection techniques with the focus on algorithms and underlying principles. Link spam and Click spam both are web spam discussed in our work. According to this work, web spam detection research has gone through a few generations: starting from simple content based methods to approaches using sophisticated link mining and user behaviour mining techniques.

REFERENCE

[1] (2014). Google Play [Online]. Available: http://play.google.com

[2] (2014). Apple Appstore [Online]. Available: http://www.apple.com/ iphone/from-the-app-store/

[3] M. Böhmer, L. Ganev, and A. Krüger, "AppFunnel: A framework for usage-centric evaluation of recommender systems that suggest mobile applications," in Proc. Int. Conf. Intell. User Interfaces (IUI), New York, NY, USA, 2013, pp. 267–276.

ISSN: 2278 – 1323

[4] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw, "Detecting product review spammers using rating behaviors," in Proc. 19th ACM Int. Conf. Inf. Knowl. Manage. (CIKM), New York, NY, USA, 2010, pp. 939–948

[5] K. Shi and K. Ali, "Getjar mobile application recommendations with very sparse datasets," in Proc. 18th ACM SIGKDD Int. Conf. Knowl.Discov. Data Mining (KDD), New York, NY, USA, 2012, pp. 204–212.

[6] Z. Wu, J. Wu, J. Cao, and D. Tao, "HySAD: A semi-supervised hybrid shilling attack detector for trustworthy product recommendation," in Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD), New York, NY, USA, 2012, pp. 985–993.

[7] B. Yan and G. Chen, "AppJoy: Personalized mobile application discovery," in Proc. 9th Int. Conf. Mobile Syst. Appl. Serv. (MobiSys), New York, NY, USA, 2011, pp. 113–126.

[8] H. Zhu, E. Chen, H. Xiong, H. Cao, and J. Tian, "Mobile App classification with enriched contextual information," IEEE Trans. Mobile Comput., vol. 13, no. 7, pp. 1550–1563, Jul. 2014. [9] H. Zhu et al., "Mining personal context-aware preferences for mobile users," in Proc. IEEE 12th Int. Conf. Data Mining (ICDM), Brussels, Belgium, 2012, pp. 1212–1217.

[10] H. Zhu, H. Xiong, Y. Ge, and E. Chen, "Ranking fraud detection for mobile Apps: A holistic view," in Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage. (CIKM), San Francisco, CA, USA, 2013, pp. 619–628.

[11] B. Davison. Recognizing nepotistic links on the web.In Workshop on Artificial Intelligence for Web Search, AAAI'00.

[12] D. Fetterly, M. Manasse, and M. Najork. Detecting phrase-level duplication on the World Wide Web. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'05, Salvador, Brazil.

[13] D. Fetterly, M. Manasse, and M. Najork. On the evolution of clusters of near-duplicate web pages. J. Web Eng., 2, Oct. 2003.

[14] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In Proceedings of the 7th International Workshop on the Web and Databases: collocated with ACM SIGMOD/PODS 2004, WebDB'04, Paris, France, 2004.

[15] D. Fogaras. Where to start browsing the web.In Proceedings of IICS.Springer-Verlag, 2003.

ISSN: 2278 – 1323