Impact of Referral Incentives on Mobile App Reviews

Noor Abu-El-Rub(✉), Amanda Minnich, and Abdullah Mueen

University of New Mexico, Albuquerque, NM 87131, USA
{nabuelrub,aminnich,mueen}@unm.edu

Abstract. Product owners occasionally provide referral incentives to the customers (e.g. coupons, bonus points, referral rewards). However, clever customers can write their referral codes in online review pages to maximize incentives. While these reviews are beneficial for both writers and product owners, the core motivation behind such reviews is monetary as opposed to helping potential customers. In this paper, we analyze referral reviews in the Google Play store and identify groups of users that have been consistently taking part in writing such abusive reviews. We further explore how such referral reviews indeed help the mobile apps in gaining popularity when compared to apps that do not provide incentives. We also find an increasing trend in the number of apps being targeted by abusers, which, if continued, will render review systems as crowd advertising platforms rather than an unbiased source of helpful information.

Keywords: Online reviews · Advertisement · Referrals · Google play · Abuse · Cliques · Incentives

1 Introduction

Providing incentives is a common strategy in modern marketing. In Google Play store, users are promised that if new users apply their referral codes, both new and old users will get reward points to spend in that app or to redeem for cash or gift cards [1]. As a result, users broadcast their referral code by posting referral reviews to increase their earning through incentives, destroying the purpose of a review system. Figure 1 shows a set of referral reviews in Google Play. These reviews are advertising a referral code in the `ChampCash` app with identical text and different referral codes. Most often, There is a notable difference in the average rating between referral and non-referral reviews. For example, the app `com.tapgen.featurepoints` has 6,037 reviews at the time of writing, and 2,147 (35.6%) of them are referral reviews. The average rating of the referral reviews (4.73) and the remaining reviews (4.08) suggests that the referral reviews and ratings are creating an undesirable bias in the review system.

Existing work focus on detecting spam and collaborative frauds. Referral reviews are different; while fake and paid reviews are written by abusers for

J. Cabot et al. (Eds.): ICWE 2017, LNCS 10360, pp. 351–359, 2017.
DOI: 10.1007/978-3-319-60131-1_20

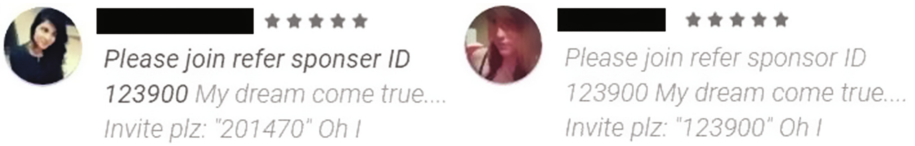

Fig. 1. Referral reviews in Google Play found in an app called `ChampCash`.

untraceable incentives, the incentive from spreading referral code via reviews is obtained from the product (i.e. app) owner, thus it is traceable. All referral reviews are untrustworthy for their monetary motivation. However, some referral reviews are worse for their spamming and adverse nature. Our goal, in this paper, is to understand how abusers are going beyond few random reviews to manipulating referral reviews and impacting the trustworthiness of the review system.

In this work, we collect and analyze referral reviews from the Google Play store. We develop a system to detect and extract referral reviews. We design a parsing pipeline that extracts app names and code words mentioned in reviews with high precision. The key challenge in the extraction process is that app names contain variable number of words of many forms (e.g. abbreviations, languages) and parts of speech. For example, it is hard to differentiate the app name "Uninstall" from the phrase "uninstall". Moreover, apps are added, edited and deleted frequently in the Play store. We use a dictionary based technique to extract app names and code words in a highly precise manner. To estimate the impact of referral reviews, we tracked the apps that provide incentives continuously for six months (October, 2015 – March, 2016). We discover that the apps that employ a rewarding mechanism gain significantly high star-ratings and downloads compared to those that do not.

2 Background and Related Work

A review is *referral* if the writer of the review gains any benefit in writing the review. For example, the reviews in Fig. 1 are broadcasting referral codes, which, if used by some new users, can earn reward points for their writers. Note that both the Android and iTunes platforms provide the functionality for incorporating a reward system in apps. In January 2016, Google began providing app developers with lists of alphanumeric codes that can be used as promotional codes [2], and Apple has supported referral codes for several years. This service encourages developers to use promotional codes. The apps we identified as rewarding apps were using these codes before Google provided the service; thus while we expect more apps to start using this reward system, we have a snapshot of them in our dataset.

Current works focus on identifying fraud reviews and reviewers, while we focus on understanding the (potentially abusive) impact of incentives on online reviews. Existing work can be categorized based on the methodologies they adopt

to detect frauds. Fraud detection using graphical/network structure is studied in [3–5] where authors exploit network effects and clique structures among reviewers and products to identify fraud. Text-based detection of fraud is studied to spot a fake review without having the context of the reviewer and reviewed product [6–8]. Temporal patterns, such as bursts, have been identified as a fraudulent behavior of businesses [9,10]. In contrast, our work looks at specific textual features of referral reviews such as referral codes, app mentions, and keywords related to a reward system. Our method also utilizes unique contextual features such as the number of downloads, the number of reviews, and the average rating, which help us gauge the impact of referral reviews.

The closest work to ours is finding fraud and malware apps in Google Play, FairPlay [11]. The article discusses a method to automatically find such apps using review-based features related to apps and their users. We focus more specifically on referral reviews and consider finding abusive users and apps which are taking part in this segment.

2.1 Data Collection

We have implemented a **two-stage** data collection process. In the *first stage*, we searched in Google play store for apps that could potentially use incentives using specific keywords. We have collected a set of 10,355 apps. For each app, we collect up to 4480[1] of the most recent reviews. In the *second stage*, we develop an algorithm to detect referral reviews and apps. We have identified 4,029 apps that have some referral reviews. To understand how these apps benefit in gaining downloads and positive ratings, we have monitored the apps continuously from October, 2015 to March, 2016. For each app, we collect its metadata (e.g. app size, app description, and rating) and developer information. The total number of reviews we have collected is 14,555,502. Each review contains title, body, date, rating, and author. The total number of unique users in our dataset is 10,327,089 users. In this stage, we have collected 74,013 referral reviews with codes.

2.2 Codes and AppNames Extraction

We develop an algorithm to detect referral reviews by identifying *codes* from the reviews. Obviously other kinds of referral reviews may exist; however, referral incentives are almost always implemented through promo codes, which gives us a significant coverage on referral reviews. We first manually generate a *blackList* and *whiteList* based on extensive examination of the dataset. The *blackList* is used to identify reviews that likely contain an app code. Some example terms from the *blackList* are: `points`, `referral`, and `code`. A *whiteList* is used to identify reviews that may contain a string that could be confused for a referral code. Some example terms from the *whiteList* include `barcode`, `PayPal`, and `zip code`.

[1] The limit is set by Google Play.

In most development platforms, codes are a random sequence of numbers, alphabets, or a combination of alphabets and numbers. We retain English reviews that contain at least one keyword from the *blackList*, then perform different checks to test if any mentioned word is a code. We use *whiteList* to exclude cases where a word can be confused for a referral code such as game scores, reward points, or abbreviations (e.g. `mp3`, `Car4you`, and `galaxy4`).

In addition to codes, abusive reviews also contain references to other apps. Knowing the app that a review is referring to will help us to measure the impact of the reward system in that referred app. Our system generates a list of app identifiers from the metadata (i.e. app title). Usually, app names are long and app developers tend to put keywords in the title describing functionality (e.g. AppCoins (How to make money)). When users refer to an app in reviews they usually use the first couple of words without mentioning the whole title. To cover all possible cases, we generate different app identifiers from each app's title. We have generated 20,682 app identifiers from 10,355 app titles. Each identifier is tagged with an appID that connects the exact app with its identifiers. Next Step takes the app identifiers and a review as input, and outputs the app name that appears in the review. We find that 2.4% of the referral reviews reference to other apps.

We detect promotional reviews with 91% precision and extract codes with 93% precision. We detect and extract the app names with 95% precision. The precision values are calculated over an unbiased sample of one hundred reviews evaluated by two judges. Note that calculating the recall rate is impossible because there is no ground truth. We also argue that our analysis does not depend on the recall rate as we have thousands of users, apps, and reviews, which are precise and large enough for accurate statistical analysis. We have provided all of our code, data, and spreadsheet of results in our supporting page [12].

3 Comparative Study Among App Groups

In this section, we categorize the apps into five groups and perform a comparative study to understand them better. The groups are: *non-promoting, promoting, source, non-promoting target*, and *promoting target* apps. Below we formally define them.

Sources: Source apps are apps that have been mentioned in promotional or referral reviews written on other apps' review pages at least once. Source apps can have some promotional reviews in their own pages. We find 25 such apps. In Fig. 2a we show the distribution of the source apps over various app categories in Google Play. The most frequent source-type is entertainment, while source apps exist in six other categories.

Promoting Targets: An app is "*targeted*" by a source app when users write reviews in the target app about promotions in the source app. If a target app has a rewarding system implemented, we call them promoting targets. A promoting target app can also be a source app in some reviews. We use a threshold of minimum five targeted reviews to separate a source from a promoting target. We find 126 apps in this category.

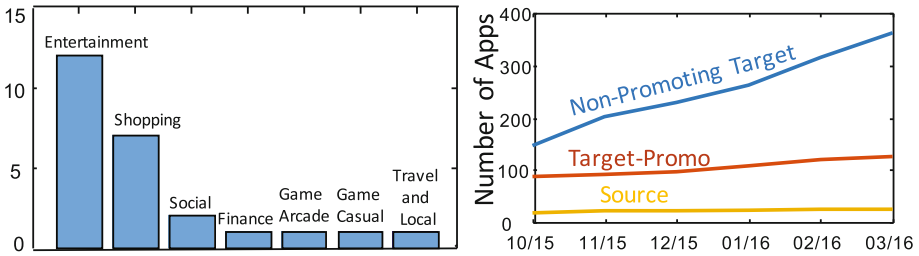

Fig. 2. (a) Source Apps Categories. (b) Growth trends of three groups of apps related to referral reviews.

Non-promoting Targets: Non-promoting targets are apps whose review page has been abused by some reviewers and have not implemented a rewarding system. We find 361 apps that are being targeted "by" source apps. These apps are mostly popular apps from top developers including Skype, Facebook, Twitter, Google, Amazon, and eBay.

Benign-Promoting Apps: Benign promoting apps have a rewarding system implemented and have reviews with promotional or referral codes. However, they are not sources or targets. We have 1150 apps in this category. We call the apps benign to distinguish them from the sources and targets. In reality, they are also abused by the reviewers.

Non-Promoting Apps: A set of randomly selected 6,328 apps that have no referral or promotional codes in reviews. We have collected the reviews for non-promoting apps during the period between October 2015 to March 2016.

Table 1 shows the summary statistics for the five app groups. Source apps have the highest average rating with the lowest variance.

Table 1. Statistics for app groups

	Benign promoters	Sources	Targets-Promo	Targets-NonPromo	Non-promoter
Number of apps	3,408	25	126	361	6,328
Average rating	3.98	4.17	3.99	4.02	3.99
ccStandard deviation rating	1.46	1.39	1.48	1.45	1.45
Number of promotional reviews	23,643	13,353	32,926	1,724	-

3.1 Feature Comparison

We compare the five groups of apps based on their total number of reviews, the number of downloads, burstiness and the number of promotional reviews. We show the results in box-plots in Fig. 3.

Based on the number of reviews, we see that the apps participating in reward systems have more reviews than random non-promoting apps. We also observe

that the source and target apps have a greater median number of reviews than the benign promoters (see Fig. 3a).

Considering the number of downloads, target apps are more popular than source apps, which explains why they are targets. Non-promoting and benign apps are very similar in the number of downloads, while source apps have significantly greater downloads than benign and non-promoting apps. This can be a demonstration of their successful referral reward systems, which are earning them a large number of downloads (see Fig. 3b).

In [10,13], authors have shown that bursts of reviews indicate spamming activities. We measure the maximum and average number of reviews an app has received in a day and take their ratio as a measure of "burstiness." We see a relatively high burstiness in source apps compared to the benign promoters. Non-promoting targets, which are also popular apps, show similar burstiness as source apps (See Fig. 3c). If we only consider the number of promotional reviews, we identify that benign apps have very few promotional reviews while source apps have a large number of such reviews. This is a significant difference that motivates further analysis on the source apps. Target apps, although having a large number of total reviews, show much less promotional reviews compared to the source apps because target apps mostly do not have their own referral systems (Fig. 3d).

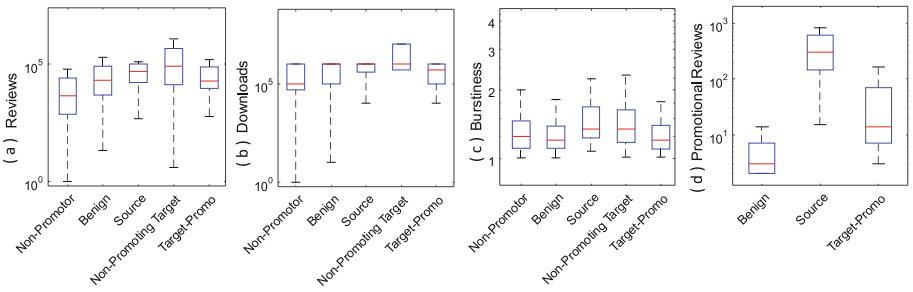

Fig. 3. Comparison among the five app groups based on four features.

Although we categorize source and target apps separately based on the reviews they have received, we have no evidence to say that the app owners have initiated such reviews.

3.2 Trend Comparison

As we demonstrate the significant difference between the app groups, we need to understand if the number of apps in the source and target groups is increasing. We show in Fig. 2b the trends for each group over the six months period of data collection. We observe that source apps are growing at a much smaller rate than the target apps. The most alarming fact is that the non-promoting target apps have almost doubled in six months. This suggests that we need to save non-promoting apps from abusive reviewers of promoting apps.

4 Discovering Abusive User Groups

In this section, we analyze the *users* who participate in writing referral reviews. We apply graph mining techniques to discover user groups who are involved in collaborative abuse. We also perform temporal analysis to understand trends in the users who are writing referral reviews. We create three different graphs connecting the review writers: *app-graph*, *text-graph* and *code-graph*.

App-graph: Two reviewers writing reviews for the same source app are connected by an edge. We expect a random graph to be formed in an unbiased review system where the reviewers mention other apps randomly without any bias.
Text-graph: We use Levenshtein distance [14] as a metric to measure text similarity. We set an error threshold of 6 for the distance function to allow an approximately 10% difference in a review as the average review length is 75 characters. An edge is added between two users if at least one pair of promotional reviews between the users has a distance less than or equal to the threshold. As shown in the Introduction, there are near duplicate reviews in the app reviews. The major reason for near duplicates is that writing an identical review to the most "helpful" review increases the chance of being ranked highly on the review page. If a group of users is posting similar text, we investigate further to identify if they are copying from each other. We find 6237 reviews using 401 unique templates by just changing the code part. The templates range from 6 to 497 characters in length, not including the code.
Code-graph: We add an edge between two users if they promote the same code. Codes are generated at random in an unbiased system, such edges, therefore, should not exist. However, we find many users who post the same code. Thus, code-graph creates an opportunity to spot groups of abusive users.

4.1 Clique Discovery

We consider a clique only if it has at least 3 users and set a minimum edge weight of 1 to find the cliques. We describe the largest cliques we have found in the three graphs defined above. In the app-graph, the largest clique was of size 346 users, which means all these users were referring to some common apps (not necessarily the same). In the text-graph, 36 users form the largest clique, and in the code-graph, the largest clique contains 65. we observe that the app-clique is disjoint to the code-clique and text-clique, while text-clique is a subset of the code-clique.

Table 2. 12 User Names from code-clique

Shreya Gupta	Shreya Gupta	chetan sahu	John Smith	Bhavna Sharma	Faqat Khan
Nancy Gupta	samita sah	chetan sahu	Harvey Dend	Sagar Sharma	Ankita Diwan

We perform a qualitative check on the code cliques. A random subset of 12 users is shown in Table 2. 100% of the reviews these users have ever written are promotional reviews, 82% have exactly 2 distinct codes and the remaining have one code and they all share the same codes (`123900` and `201470`). Three users have the same name and profile picture and 72% users have changed their profile names at least once.

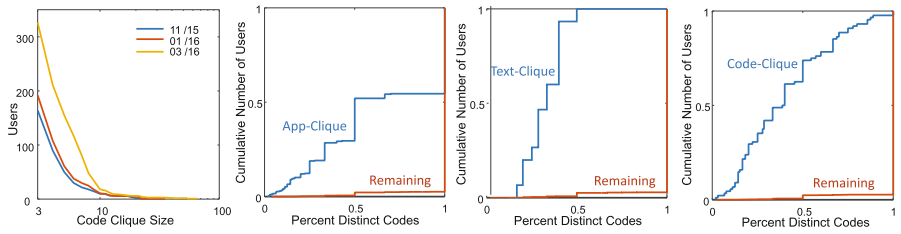

Fig. 4. (a): Size distribution of code-cliques over time, Remaining: Empirical CDFs for the three graphs.

4.2 Clique Properties

To perform more principled analysis on the cliques, we select a stricter edge weight of 10 and find the extreme incentivized users. We find 317 cliques in the app-graph, 37 cliques in the code-graph, and 6 cliques in the text-graph. For each graph, we compare the users participating in any clique against the remaining users who did not participate in cliques. We use the percentage of distinct codes over the number of promotional reviews. If the percentage is 100%, it means the reviewers are not reusing codes in their reviews. If the percentage is 20%, it means the reviewers are, on average, writing five reviews per referral code. We show the CDF (cumulative distribution function) of this metric over all the users who are in some clique (see Fig. 4). We also show the CDFs for users who are not in any clique. There is a significant difference between the CDFs for all of the three cliques, demonstrating that the users forming cliques are reusing promotional codes in multiple reviews.

We show the size distribution of the cliques from the code-graph in Fig. 4a. Naturally, we have a large number of small cliques and a few large cliques. We show three distributions for three datasets accumulated at two months interval. We identify a trend in both number and size of the cliques. This is an alarming indication that the number of abusers is growing rapidly.

5 Conclusion

This paper identifies a new type of abuse in review systems. Mobile apps support referral rewards, which create opportunities for users to write incorrect,

untruthful, and abusive reviews. In this paper, we identify referral reviews, and analyze them to understand the following: ⋆ the number of referral reviews is rapidly increasing, ⋆ and apps are indirectly benefited from referral reviews in terms of the number of reviews and downloads.

References

1. Champcash: champcash (2015). https://play.google.com/store/apps/details?id=com.cash.champ
2. Google: developer console promotional code terms of service (2016). https://play.google.com/about/promo-code-developer-terms.html
3. Akoglu, L., Chandy, R., Faloutsos, C.: Opinion fraud detection in online reviews by network effects. In: Proceedings of ICWSM, pp. 2–11 (2013)
4. Chau, D.H., Pandit, S., Faloutsos, C.: Detecting fraudulent personalities in networks of online auctioneers. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 103–114. Springer, Heidelberg (2006). doi:10.1007/11871637_14
5. Wang, G., Xie, S., Liu, B., Yu, P.S.: Review graph based online store review spammer detection. In: Proceedings - IEEE International Conference on Data Mining, ICDM 2011, pp. 1242–1247 (2011)
6. Ott, M., Choi, Y., Cardie, C., Hancock, J.T.: Finding deceptive opinion spam by any stretch of the imagination. In: Proceedings of HLT 2011, pp. 309–319 (2011)
7. Jindal, N., Liu, B.: Opinion spam and analysis. In: Proceedings of WSDM 2008, pp. 219–230. ACM, New York (2008)
8. Sun, H., Morales, A., Yan, X.: Synthetic review spamming and defense. In: KDD 2013, p. 1088 (2013)
9. Xie, S., Wang, G., Lin, S., Yu, P.S.: Review spam detection via temporal pattern discovery. In: Proceedings of KDD 2012, p. 823 (2012)
10. Fei, G., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M., Ghosh, R.: Exploiting burstiness in reviews for review spammer detection. In: Proceedings of ICWSM, pp. 175–184 (2013)
11. Rahman, M., Rahman, M., Carbunar, B., Duen, H., Chau, G., Tech: fairplay: fraud and malware detection in google play. In: Proceedings of the 2016 SIAM International Conference on Data Mining, pp. 99–107. SIAM (2016)
12. Supporting Page - supporting webpage containing experimental result, data and code. http://www.cs.unm.edu/~nabuelrub/referral_reviews/
13. Minnich, A.J., Chavoshi, N., Mueen, A., Luan, S., Faloutsos, M.: TrueView: harnessing the power of multiple review sites. In: Proceedings of WWW 2015, pp. 787–797 (2015)
14. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. J. ACM (JACM) **21**(1), 168–173 (1974)