Google Play Malware Detection based on Search Rank Fraud Approach

Fareena N[1*], Yogesh C[2], Selvakumar K[3] and Sai Ramesh L[4]
[1] Department of CSE, Anna University Regional Campus, Tirunelveli, India
[e-mail: fareena.n@auttvl.ac.in]
[2]School of Computer Science and Engineering
VIT Chennai Campus, India
[e-mail: yogesh.c@vit.ac.in]
3Department of Computer Applications, NIT, Trichy, India
[e-mail: kselvakumar@nitt.edu]
[4] Department of IST, CEG Campus, Anna University, Chennai, India
[e-mail: sairamesh.ist@gmail.com]
[*]Corresponding author: Fareena N

Received September 11, 2021; accepted December 20, 2021;
published November 30, 2022

Abstract

Google Play is one of the largest Android phone app markets and it contains both free and paid apps. It provides a variety of categories for every target user who has different needs and purposes. The customer's rate every product based on their experience of apps and based on the average rating the position of an app in these arch varies. Fraudulent behaviors emerge in those apps which incorporate search rank maltreatment and malware proliferation. To distinguish the fraudulent behavior, a novel framework is structured that finds and uses follows left behind by fraudsters, to identify both malware and applications exposed to the search rank fraud method. This strategy correlates survey exercises and remarkably joins identified review relations with semantic and behavioral signals produced from Google Play application information, to distinguish dubious applications. The proposed model accomplishes 90% precision in grouping gathered informational indexes of malware, fakes, and authentic apps. It finds many fraudulent applications that right now avoid Google Bouncers recognition technology. It also helped the discovery of fake reviews using the reviewer relationship amount of reviews which are forced as positive reviews for each reviewed Google play the android app.

Keywords: Google Play, Android phone apps, fraudulent behavior, malware, and search rank fraud.

1. Introduction

Google Play (already Android Market) is a computerized circulation maintained by Google over the past two decades. Mostly, the apps available in this forum are authorized ones and the users who are downloading also be authorized by the developer and Google play store before they will start using the app. In this environment, all kinds of the app for supporting knowledge, entertainment, and gaming are available.

In the Google App store, [1] some applications may available for payment and most of them are malware-protected apps. But most of the apps are free of cost and non-secure for the environment where we are going to use them. Most of the apps require permission to access the data in the gadgets where it is going to be used including some personal sensitive information.

The business achievement of android application markets, [2] for example, google play and the motivating force model offer plentiful mainstream applications, making them engaging focuses for fraudsters. Some deceitful engineers misleadingly help the inquiry rank and fame of their applications through phony surveys and fake installation tallies, while vindictive designers use application markets as a launchpad for their malware. [3] The inspiration for such practices is the effect of application prominence floods convert into money-related advantages and sped up malware expansion. Fraudulent designers much of the time abuse publicly supporting destinations to employ groups of willing laborers to submit misrepresentation, by and large, copying reasonable, unconstrained exercises from disconnected individuals. Likewise, the endeavors of Android markets to recognize and expel malware are not generally fruitful. [4] For example, Google play utilizes the Bouncer framework to dispatch the malware. Nonetheless, out of the gathered google play applications information which is investigated using virus-out, in total, just 12% were hailed by in any event one anti-virus instrument and 2% were recognized as malware by in any event 10 tools around.

The existed portable malware recognition researches are concentrated on the powerful investigation of static examination of code by the applications on the device and authorizations. In many cases, malware is injecting unauthorized data to sidestep hostile anti-virus software. This kind of injection is considered in recent research in malware detection of the app. This work looks to recognize both malware and search rank fraud subjects in google play.

The work recommended that vindictive designers resort to search rank fraud to help the effect of their malware. The odious demonstrations have been revealed by choosing the perception that fraudulent and malevolent practices abandon indications on application markets as trails. For example, the significant expense of setting up legitimate google play accounts powers fraudsters to reuse their records across survey composing employments, making them liable to audit more applications in like manner than standard clients. Asset imperatives can propel fraudsters to post audits inside brief timeframe stretches. Genuine clients influenced by malware may report terrible encounters in their surveys.

The proposed system provides the ability to identify the fraud apps on the collected app rate and review the data set, with the use of user edge connection-based information. This proposed system uses semantic and similarity measures to identify the fraud apps and the sentiment analysis is done to predict the positive and negative reviews. Both these approaches with time-based reviews are used to predict the fraud apps. Then the results are displayed in the

graph to visualize the percentage of fraud apps from Google play.

The remainder of the paper is composed as follows: Section 2 gives the writing overview which highlights the work related to Google play malware detection using the classification method. Section 3 gives the details of the system architecture and modules presents the overall block diagram and explains all the modules in detail. Section 4 describes the implementation and experiments and results of cosine similarity and sentiment analysis using a classification algorithm. Finally, Section 5 highlights the conclusion and future work**.**

2. Literature Survey

To handle the security issues brought about by malware of Android OS, Hengshu Zhu et al. [6] have proposed an exceptionally productive crossover distinguishing plan for Android malware. In this paper, the creator proposed some recognizing methods, for example, highlights dependent on conventional Permission and API call highlights to improve the exhibition of static location. The crumbling issue of conventional capacity called graph-based malware identification was likewise kept away. The visitation results demonstrated that the recommended scheme accomplished high malware identifying accuracy, and the plan could be utilized to set up Android malware recognizing cloud administrations, which can naturally adjust high productivity dissecting techniques as indicated by the properties of the Android applications.

Erika Chin et al. [5] have concentrated to exhibit the Android malware available in the SDK files in the Android stage itself. [7] Based on the examination conducted at four different agents, the identification of malware using this approach will achieve a better percentage than the existing pessimistic scenarios.[8] The results of these experiments are more likely to create a cutting edge against versatile malware arrangements.

Micheal C Grace et al. [9] have introduced a Machine Learning (ML) based framework for the identification of malware on Android gadgets and the framework extricates a few highlights and prepares a One-Class Support Vector Machine (SVM) in a disconnected (off-gadget) way, to use the higher processing intensity of a server or group of servers[10].

Patrik Traynor et al. [12] have examined the present status of portable malware in the wild and dissected the motivations. After the perception that 4 picks of malware use root adventure to mount advanced attacks on Android mobiles and analyzed the motivations that cause non-malignant cell phone hobbyists to distribute root misuses and reviewed the accessibility of root abuses.

Ee-PengLim et al. have proposed a novel method for processing a rank aggregation dependent on matrix consummation to dodge commotion and deficient information. The proposed technique takes care of an organized matrix finishing issue over the space of skew-symmetric matrices. The creator demonstrates a recuperation hypothesis enumerating when the proposed approach will work.

Nikita Spirin and Jiawei Han. [11] has detailed an overview of webspam recognition, which completely presents the standards and calculations in the writing. Undoubtedly, crafted by Web ranking spam identification is founded on the investigation of positioning standards of web crawlers, for example, Page Rank and inquiry term recurrence. This is not quite the same as ranking fraud extortion recognition for versatile Apps. Sulthana et al [11] stated their work is to extract the real opinion of the user for reviews on Twitter. So, this article will show the importance of sentiment analysis using natural language processing forgetting the real opinion of the user, and how it helps in future prediction

Chia-Mei Chen et al. [2] have proposed a static strategy to distinguish malware in portable applications. In this framework, figuring out the idea is utilized to create source code for the dubious APK documents. After that utilizing a structured mapping creator fabricates the structure for the classes. At long last utilizing an information stream idea, a few examples of the diverse kind of threats have been made and used to distinguish the malware in applications. Contingent on the number of scourging designs the adequacy of this strategy is determined.

David F Gleich and Lek-heng Lim. [10] has proposed a novel method for figuring a rank aggregation dependent on matrix consummation to stay away from the clamor and deficient information. The method discussed in this work organized the given framework in matrix format. The matrices are skew-symmetric based to identify the issues in the ranking. And also the app creator who developed that app designed the theorem for recovering the app from malware using some celluloid information. They likewise play out a nitty-gritty assessment with celluloid information and a narrative examination with Netflix appraisals. To discover the arrangements, they used the SVM for solving the completion of the matrix. Rank accumulation is joined with the structure of skew-symmetric matrices. The creator improved the current approach for matrix fruition to deal with skew-symmetric information.

The Ranking Risks of Android Apps Using Probabilistic Generative Models given by Alaa Salman et al. [1] is one of Android's safeguard systems against malevolent applications where a hazard correspondence is examined. This methodology is ineffectual as it presents the hazard data of each application in an independent manner and in a way that requires an excessive amount of specialized information and time to distill valuable data.

Kent Shi and Kamal Ali. [8] has examined to ensure the audit of spanners or spam surveys. The spammer may target just explicit protection. From that point forward, they gave fake surveys to that specific versatile application by making an alternate record to survey that account. The creator proposes a novel-based scoring technique to distinguish every survey of the specific item. The creator makes exceptionally suspicious as a subset. By utilizing online spammer assessment programming the phoniness of the survey is determined. After the fulfillment of the assessment, the outcome demonstrates the viability to anticipate bogus audits.

The Evaluation of ML classifiers for versatile malware recognition was proposed by Jyrki Kivinen and Manfred K Warmuth. [7] passes on the rising number of clients' welcome drudges to create vindictive applications. Furthermore, the security of sensible information accessible on cell phones is taken lightly. Given the experimental results, the classifiers used in this work, precisely increase the precision and specificity while compared with CNN techniques. Selvakumar et al [9] provide the clustering mechanisms to group the users based on their similarities. The K - Means is used in this work to cluster a similar user in a single group. By using this kind of method, the app provider will easily identify similar users who need a similar app for installing their malware app. Another work by Selvakumar and Sendhilkumar [10] helps to provide personalized search results for the user using the neural networks approach. This will also find a similar user group to easily identify the user needed for their app.

The overview of the related work is about identifying fake mobile apps using web-based software tools or rank aggregation methods. The main challenge in this project is to analyze the fake mobile apps using semantic and similarity measures and the reviews are analyzed based on their sentiment and interrelation between the co-reviewers action the entire fake apps are identified.

3. Proposed System

This section discusses Google play fraud app detection system to detect malicious and fraudulent apps and is depicted in **Fig. 1**.

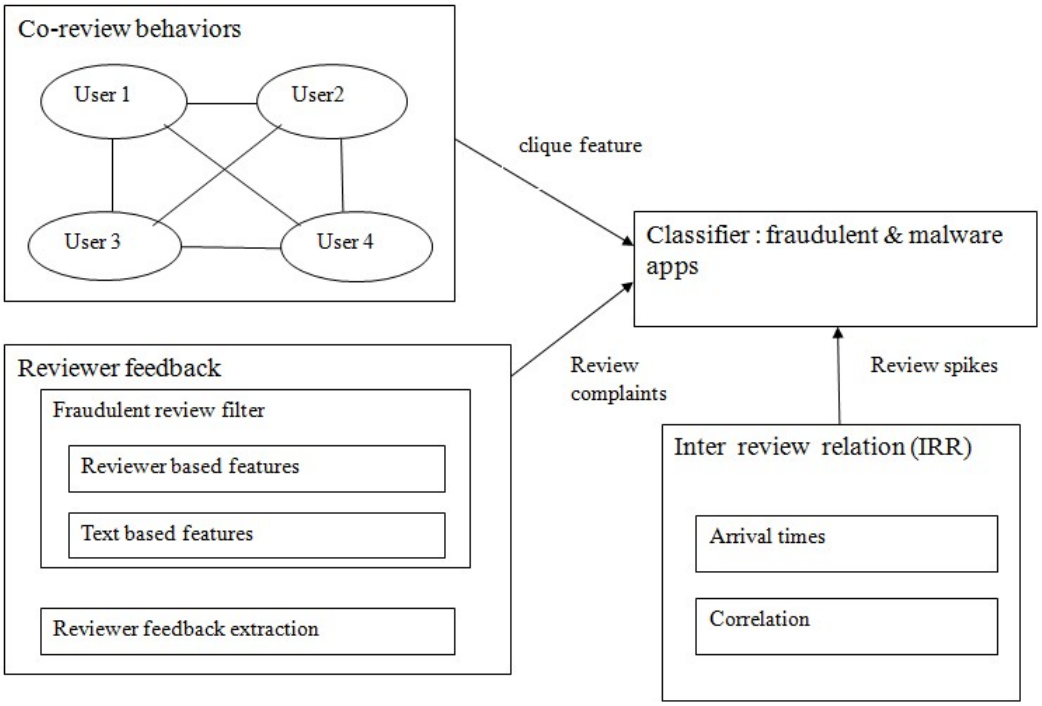

Fig. 1. The architecture of the Google Play Fraud App Detection System

The modules involved and the description of the modules are given below,
- Co-Review Behaviors
- Reviewer Feedback
- Inter Review Relation (IRR).

3.1 Co-Review Behaviors

This module features the perception that fraudsters who control numerous records will reuse them across different occupations. The objective is to identify sub-sets of application commentators that have performed noteworthy basic audit exercises before. It portrays the co-review graph idea, is utilized to officially introduce the weighted maximal clique enumeration problem, and at that point presents a proficient heuristic that uses characteristic confinements in the practices of fraudsters.

3.1.1 Co-review graph concept

Let the co-review graph of an application, be where hubs relate to client accounts who investigated the application and undirected edges have a weight that demonstrates the number of applications looked into in like manner by the edges endpoint clients. The co-review graph idea normally recognizes client accounts with critical past survey exercises. **Table 1**, shows the attribute description of the collected data set from the google play app store.

Table 1. Attributes Table

S. No.	Attribute name
1	application name
2	apps description
3	average rating
4	author reviews
5	author rating
6	author-name
7	review date and time

3.2 Reviewer Feedback

Reviews written by veritable clients of malware and fake applications may depict negative encounters. The RF module misuses this perception through a two-space methodology: (i) recognize and sift through false surveys, and at that point (ii) distinguish malware and fraud demonstrative criticism from the rest of the surveys.

3.2.1 Fraudulent Review Filter

This approach posits that certain characteristics can exactly pinpoint genuine and fake reviews. It also proposes various such features, which are shown in Table 1 for a summary, are defined for a review R written by user U for an app A.

3.2.2 Reviewer-Based Features

The skill of user U for application A characterized as the number of recaps U composed for applications that are like A, as recorded by Google Play. The predisposition of U towards A: the number of surveys composed by U for different applications created by A programmer. Likewise, it removes the complete cash paid by U on applications it has assessed, the number of applications that U has preferred, and the quantity of Google+ adherents of U.

3.2.3 Text-Based Features

NLTK library is accustomed to grouping the sentiment analysis through Random Forest classifier, prepared on two datasets: (i) 1,041 Timelines of positive surveys for 2 applications from the fake application dataset. The first application has different spikes while the subsequent one has just a single noteworthy spike. Sentences were removed from arbitrarily chosen 350 positives and 410 negative Google Play audits, and (ii) 10,663 sentences were extricated from 700 positives and 700 negative IMDB movie surveys.

The 10-fold cross-validation of the Random Forest classifier over these datasets uncovers a bogus negative pace of 16.1%and a bogus positive pace of 19.65%, for a general precision of 81.74%. They ran a binomial test for a given precision of p=0.817 over N=1,041 cases utilizing the binomial circulation binomial(p, N) to survey the 95% certainty span for my outcome. The deviation of the binomial dissemination is 0.011. In this way, they are 95% sure that the genuine presentation of the classifier is in the span (79.55,83.85)

3.2.4 Extraction of Reviews

The guess is that since no application is great, a reasonable survey that contains both applications' positive and negative opinions is bound to be real, and there should exist in connection between the audit commanding feeling and its rating. This model concentrates criticism from the rest of the surveys. For this, NLTK is utilized to remove 5,106 action words, 7,260 things, and 13,128 modifiers from the 97,071 audits gathered from the 613 highest quality level applications. At that point non-ASCII characters and stop words are expelled and applied lemmatization and disposed of words that show up all things considered once.

3.3 Inter-Review Relation (IRR)

These modules use temporal relations between surveys, just as relations between the audits, rating, and introduce checks of applications, to recognize dubious practices.

3.3.1 Temporal Relations

To make up for a negative audit, an assailant needs to post countless positive surveys. Let RA indicate the normal rating of an application A not long before getting a one-star survey. To make up for the one-star survey, an assailant needs to post in any event positive audits are extract temporal features the number of days with identified spikes, and the most extreme abundancy of a spike: (i) the proportion of introduced to appraisals as two highlights, I1/Rt1 and I2/Rt2 and (ii) the proportion of introduces to surveys, as I1/Rv1 and I2/Rv2. (I1, I2) indicates the introduced tally time frame application, (Rt1, Rt2) its rating stretch, and (Rv1, Rv2) its (certified) survey span.

4. Implementation and Results

The algorithms used in this proposed Fairplay system for fraudulent detection are discussed and experimental results are analyzed based on the performance measures.

4.1 Dataset Preprocessing

The data set is developed based on the collection of apps from Google play. The ratings and reviews posted for the apps are handled to detect fraud reviews. The processing is done for the collected ratings and reviews data and inconsistencies are removed if any using the data cleaning method. The processed data are analyzed to detect the behavior of the users based on the reviews posted for the apps. The data with the respective features are collected from the Google play store.

4.2 Co-Reviewers Edge Connection Analysis

The preprocessed data was then analyzed to detect the behavior of the users based on their reviews and ratings posted below the apps. There will be many reviewers for a particular application. All the reviewers may not be users, the person working for the app, or the opponent app. They can able to access illegal things by applying positive reviews for their particular app or grading negative reviews for the different apps. To find the malware actions the co-reviewers behavior is completely undertaken based on different parametric features such as semantic and similarity measurements are done. In the semantic-based approach, the reviewers' review text type is compared with other reviewers and a threshold value is fixed. If the threshold value is greater than equal to that particular review the user connection between the reviewers is similar and the same behavioral reviewers are extracted from the dataset and

are detected as co-reviewers and the behavior is completely analyzed between the reviewers and is depicted in the different user behavior of the apps.

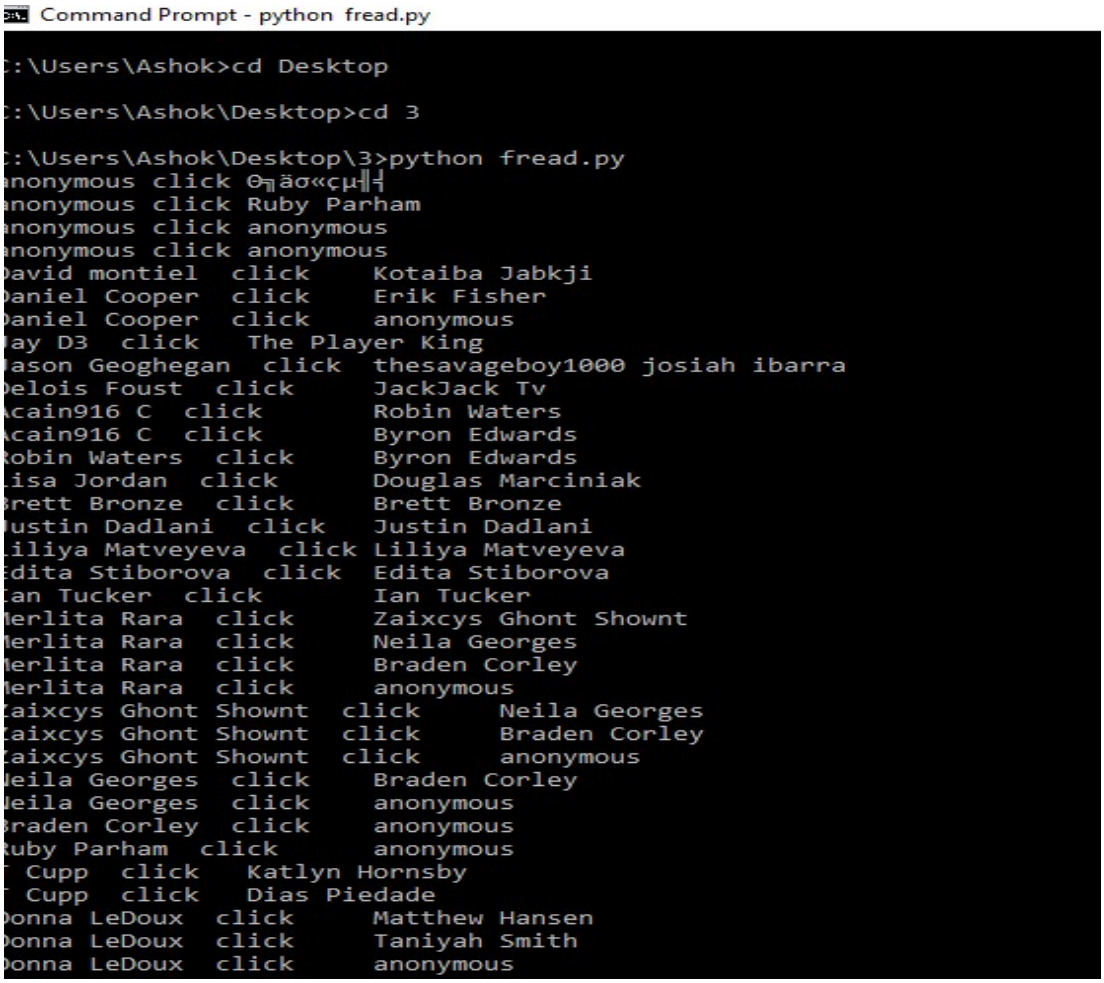

Fig. 2. Co-Reviewers Behavior Analysis

The co-reviewers behavior analysis is identified based on the similarity analysis between the users who posted reviews for the apps. The similarity score is identified based on the cosine similarity analysis, when the similarity score is greater than 0.5 then the reviews are extracted and a relationship is identified between the users which is shown in **Fig. 2** with the username, and the matched reviews are highlighted. The user edge connection based on the review relationship is projected with the graph, where the users act as the nodes, and the connected user edges are given to discover the relationship between the users which is depicted in **Fig. 3** and it gives the visualization between the co-reviewers to analyze the relations of the posted by the users.

Algorithm 1 Clique Finder Algorithm

Input: Reviews

Output: Undirected weighted graph

```
Initialize cliques
for each author in the authors do
    for each author1 in authors do
        if author[comment] is equal to author1[comment] then
            if  edge already exists between author and author1 then
                increment weight by 1
            else
                create a new edge
                set weight as 1
             end if
        end if
    end for
end for
set threshold for weighted graph
for each clique in cliques do
    if weightless than the threshold  then
        add clique edge with weights to the graph
    else
        discard the edge
    end if
end for
display the clique graph with weighted labels
```

The network applied for user edge connection based on the reviews posted by users is given as input and it is depicted in Algorithm 1, where the undirected weighted graph is given as output. The total number of users who posted the reviews is compared with the other users to find any edge connection between them based on the similarity of the reviews. If user1 review is equal to user2 then there exists an edge weight between them, whereas if an edge already exists between user1 and user2 then increment the weight by 1, otherwise create a new edge with the weight of 1. The threshold is set for weighted graphs. If the weighted graph is greater than the threshold adds a clique edge with weights to the graph or discard the edge, then display the clique graph with weighted labels.

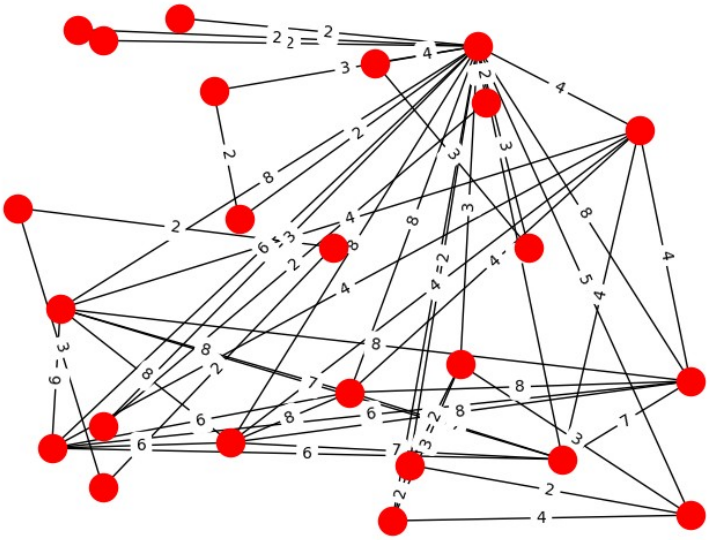

Fig. 3. User Review Edge Connection Analysis

4.3 Behaviour of Co-Reviewers

The objective is to distinguish subsets of application reviewers that have performed critical regular survey exercises previously. They depict the co-survey graph idea, which is utilized to officially introduce the weighted maximal clique enumeration issue, at that point present a proficient heuristic that uses normal restrictions in the practices of fraudsters. The similarity between the co-reviewers is identified using the cosine similarity algorithm and the reviewers' feedback is analyzed based on a sentiment analysis algorithm.

4.3.1 Similarity Co-reviewer Analysis

Cosine comparability is used to find the similarity between the points where it is represented by the matrix point between them. The value 1 is represented for cosine 0 where some of the edges are less than 1 which is represented in Algorithm 2. Based on the representation given in the algorithm, the likeness using cosine is 1 when the direction between the vectors is the same or otherwise 0 when it is and two vectors contradicted have comparability of - 1, free of their greatness.

Algorithm 2 Cosine-similarity Algorithm

Input: List of documents

Output: Cosine similarity score

1: for iteratej f rom0tonumbero fdocuments do

2: document1=documents[j]

3: document2 documents[j]

4: vector1 = to_vector(document1,word_to_id)

5: vector2 = to_vector(document2, word_to_id)

6: initialize sum ,norm1, norm2.

7: for iteratei over from 0 to size of word do

```
        a = vector1[i]
        b = vector2[i]
          sum += a*b
          norm1 += a*a
          norm2 += b*b
    end for
    normalization = sqrt(norm1)*sqrt(norm2)
    if normalization is nonzero then
          cosine_similarity = sum / normalization
    end if
end for
```

Cosine comparability is mainly applicable in the space where we need positive results and it may be limited in [0, 1]. In this model, the user's reviews are compared with all the reviews, and the similarity score is calculated between the reviews when the review is more similar the score will be high than the threshold value. If the similarity score is greater or equal to 1 then the particular review users are given a co-reviewer connection.

4.4 Feedback Reviewer Module

Sentiment investigation is indirectly or directly grasping the natural language processing for handling the semantics and extracting the opinion from them which helps to evaluate or distinguish the emotions of the given review as it appears in Algorithm 3. The reviews posted by the users for the particular app may be positive or negative based on the app's performance. The performance of the app is identified based on sentiment analysis. It is analyzed by the NLTK to identify under which classification the reviews are undertaken.

Algorithm 3 Sentiment Analysis Algorithm

Input: Reviews

Output: Classified output

```
for reviewinreviews  do
    T  = tokenize(review)
    senti = 0
    for t1inT  do
          senti = senti + score(t1)
    end for
    if senti> 0 then
          Set Positive
    else
          Set Negative
```

11: end if

12: end for

The extracted co-reviewer feedback is analyzed based on the reviews posted for the apps. The similar reviews are analyzed and the sentiment of the feedback is analyzed to predict the positive and negative reviews. If the same user has posted many positive or negative reviews about the apps is analyzed that is the behavior of the particular user is detected. The analyzed user sentiment is extracted and then it will be compared with the co-reviewer behavior for the detection of fraud reviewers and apps. The fraud apps are detected with the user edge connections and if the review rate is greater than the threshold value then it is detected as fraud apps.

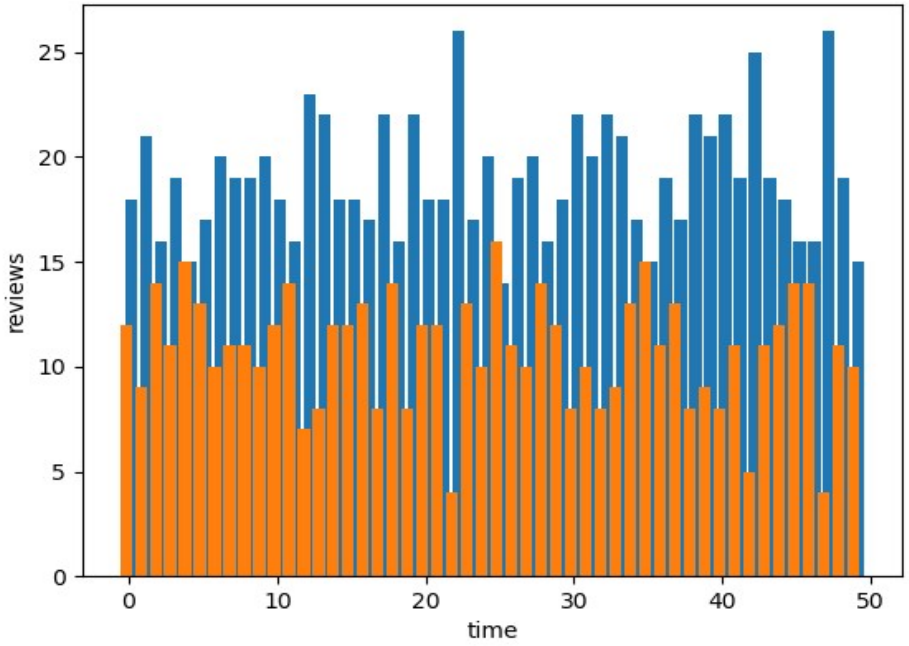

Fig. 4. Reviewer Analysis Based on Time

The co-viewer data and the sentiment review data about the apps are analyzed based on the time-based feature. Based on the interconnection the fraud apps are detected. If the review is posted at the same time for different apps and the sentiment of the app is also similar then it is considered a fraud app. The time difference between reviews is calculated and it is depicted in **Fig. 4** which detects the fraud apps based on the interconnection.

4.5 Validation Results

On account of classifiers, for example, random forests, the yield is a constant worth that associates with the probability of the perception of having a place with class 0 or 1. In this way, a threshold must be characterized to decide the last class (0 or 1), given the genuine worth acquired with the classifier. The recipient operating characteristic curve (ROC) offers a method of imagining various results. Fawcett portrays ROC arcs as delineating the relative exchange offs between benefits (true positives) and costs (false positives), working very well by and by as a general proportion of classifier execution. Perceptions with a score under the threshold are named class 0, while a score over the limit would foresee that the perception has

a place with class 1. Cross-approval from the three picked groups of models (logistic regression, SVM, and random forests), it is necessary to predict the best performance for identifying fake apps. Moreover, it wants to determine the best hyper parameters for every model by classifying the fraud apps.

Table 2. Accuracy Results of Google Fairplay

Strategy	FP	FN	Accuracy
Random forest	1.01	3.52	97.74
Decision tree	3.01	3.01	96.98

4.6 Performance Measures

Performance measures before evaluating the results of each model, the analysis is done for the performance measures to evaluate Fairplay as shown in **Table 2**. The collected dataset is about all the 97, 071 audits of the 613 best quality level malware, fake and kindhearted applications, composed of 75, 949 clients, just as the 890, 139 applications evaluated by these clients. In the accompanying, the assessment is done depending on the capacity of regulated learning calculations to accurately characterize applications as either kindhearted, false, or malware. In particular, in the main trial, the information is prepared uniquely on false and kind application information and tests the capacity to precisely characterize an application as either fake or amiable. In the subsequent investigation, the information is prepared and tried distinctly on malware and considerate applications. In the third investigation, the classifier is prepared on deceitful and amiable applications, at that point test its exactness to order applications as either malware or considerate.

At last, the most effective highlights while grouping deceitful. There are two other potential results in which the expectation brings about a blunder. The type-I error happens when the positive class is anticipated, yet the perception name is 0 (these expectations are called bogus positives). The type-II error happens when the expectation of class 0 doesn't concur with the perception of genuine class which would be 1 (false negatives).

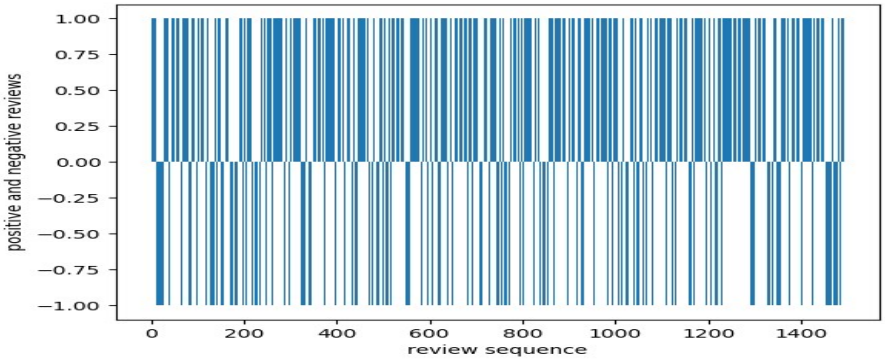

Fig. 5. Positive and Negative Review Analysis

The performance measure is given in **Fig. 5** as the evaluation analysis. The entire informational collection was recently part of preparing and testing sets. Presently a further split is performed on the preparation set. The part is utilized in real preparation, while the rest, known as the approval set, is utilized in finding the best boundaries of each model. It can

evaluate the exhibition by rehashing the preparation approval methodology on various occasions.

5. Conclusion and Future Work

The Fairplay detection system is introduced to distinguish both, fake and malware Google Play applications. The investigations on a recently contributed longitudinal application dataset, have demonstrated that a high level of malware is engaged with the search rank fraud approach; both are precisely recognized by the FairPlay framework. Likewise, the capacity of the Fairplay framework to find many applications that dodge Google Play location innovation, including another kind of coercive fraud assaults is distinguished utilizing the Fairplay framework. The future work is to train the fair play system with neural networks to obtain efficiency in a better way and to obtain the fraud apps abundantly using malware tools.

Acknowledgement

Mr. S. Shyam sundar was awarded for research fellowship under Visvesvaraya PhD scheme for Electronics & IT doe carrying out this research project. (Awardee unique number: MEITY-PHD-1877)

References

[1] A.Z. Yang, S. Hassan, Y. Zou and A.E. Hassan, "An empirical study on release notes patterns of popular apps in the Google Play Store," *Empirical Software Engineering*, vol. 27(2), pp. 1-38, 2022. Article (CrossRef Link)

[2] K. Joshi, S. Kumar, J. Rawat, A. Kumari, A. Gupta et al., "Fraud App Detection of Google Play Store Apps Using Decision Tree," in *Proc. of 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, vol. 2, pp. 243-246, 2022. Article (CrossRef Link)

[3] E. Noei and K. Lyons, "A study of gender in user reviews on the Google Play Store," *Empirical Software Engineering*, vol. 27(2), pp. 1-28, 2022. Article (CrossRef Link)

[4] M.C. Meacham, E.A. Vogel and J. Thrul, "Vaping-related mobile apps available in the Google Play Store after the Apple ban: content review," *Journal of medical Internet research*, vol. 22(11), p. e20009, 2020. Article (CrossRef Link)

[5] D. Saravanan, J. Feroskhan, R. Parthiban and S. Usharani, "Secure Violent Detection in Android Application with Trust Analysis in Google Play," *Journal of Physics: Conference Series*, vol. 1717(1), pp. 012055, 2021. Article (CrossRef Link)

[6] W. Venter, J. Coleman, V.L. Chan, Z. Shubber, M. Phatsoane et al., "Improving linkage to HIV care through mobile phone apps: randomized controlled trial," *JMIR mHealth and uHealth*, vol. 6(7), pp. e8376, 2018. Article (CrossRef Link)

[7] J.D. Akkara and A. Kuriakose, "Innovative smartphone apps for ophthalmologists," *Kerala Journal of Ophthalmology*, vol. 30(2), pp. 138-144, 2018. Article (CrossRef Link)

[8] D. Dewiyanti, A.M. Puspasari, F.S.A. Kamil and B.K. Ningtyas, "Exploratory study of visual enhancement to display smart apps on android phones for selasar imaji library," *International Journal of Design (INJUDES)*, vol. 1, pp. 17-26, 2021. Article (CrossRef Link)

[9] L. Li, T.F. Bissyandé and J. Klein, "Rebooting research on detecting repackaged android apps: Literature review and benchmark," *IEEE Transactions on Software Engineering*, vol. 47(4), pp. 676-693, 2021. Article (CrossRef Link)

[10] D.H. Yang, Z.Y. Li, X.H. Wang, K. Salamatian and G.G. Xie, "Exploiting the Community Structure of Fraudulent Keywords for Fraud Detection in Web Search," *Journal of Computer Science and Technology*, vol. 36(5), pp. 1167-1183, 2021. Article (CrossRef Link)
[11] D. Jovanovic, M. Antonijevic, M. Stankovic, M. Zivkovic, M. Tanaskovic et al., "Tuning Machine Learning Models Using a Group Search Firefly Algorithm for Credit Card Fraud Detection," *Mathematics*, vol. 10(13), pp. 2272, 2022. Article (CrossRef Link)

Dr. N. Fareena Working as Assistant Professor in the Department of Computer Science and Engineering, Anna University-Regional Centre, Tirunelveli. She completed her Ph.D. in the area of MANET from Anna University, Tamilnadu, India. She did her B.E. and M. E under the affiliation of Anna University, Tamilnadu, India. She has more than ten years of teaching and research experience. She published more than twenty research articles in reputed journals and Scopus indexed conferences. Her current area of research is MANET, machine learning and IoT.

Dr Yogesh C is currently an Associate Professor at School of Computing and Engineering, Vellore Institute of Technology, Chennai Campus. He obtained his PhD in Computer Science and Engineering at Universiti Malaysia Perlis, Malaysia. He published more than 20 research articles in reputed International and National journals. His area of research interests are Machine Learning, Feature Extraction and Selection, Optimization and Deep learning.

Dr. K. Selvakumar is currently working as an Assistant Professor in the Department of Computer Applications, National Institute of Technology, Tiruchirapalli, India. He completed his M.E. in CSE in 2006 and Ph.D in 2017 from Anna University, Chennai. He has fifteen years of teaching and research experience. He has published more than 50 research articles in reputed international journals and conferences. His area of research is Wireless Networks, Data Analytics and Artificial Intelligence.

Dr. L. SaiRamesh is currently working as a Faculty in Department of Information Science and Technology, Anna University, Chennai, India. He completed his M.E. in CSE in 2007 and Ph.D. in 2015 from Anna University, Chennai, India. He has fifteen years of teaching and research experience. He published more than 70 research articles in reputed journal and international conferences. His are of research is Wireless Networks, Cloud computing and Machine Learning.