Netwok Science: PlayStore Apps Analysis

Gopal Ramesh Dahale

11840520, Indian Institute Of Technology Bhilai, gopald@iitbhilai.ac.in

Abstract. We focused on analyzing the apps from the Google Play store that provides a wide range of data features (price, rating, etc.). A network of apps is formed using the scraped data from the official play store website and different centrality measures were analyzed. We checked the scale-free nature of the network and evaluated 3 network community detection algorithms which led to interesting results. Our findings leverage graph theory and network analysis.

Keywords: Network · Centrality · Community Detection · Clauset-Newman-Moore · Louvain Method · Infomap · Google Play Store

1 Introduction

One of the quickest areas of the downloadable software application sector is mobile apps. We choose Google Play Store over all other markets because of its growing success and recent rapid growth. The fact that 96.7 percent of apps are free [9] is one of the key reasons for the growth. We here explore the properties formed by the network of apps.

2 Related Works

The majority of this approach was inspired by [12] and [8] which gave a description of how to construct a network from a list of Amazon product reviews. We used this approach to create a network of Google Play Store apps.

[12] helped us to decide how and which community detection algorithms to chose to analyse the network.

3 Data Set

3.1 App Data

The data was scraped from the official website of Google Play Store [4] using google-play-scraper [10] written in Javascript. Apps were scraped with over 61 categories with a disk size of 600MB. The category-wise distribution is shown in fig 1. As we can see most of the apps were collected from the "Family" category and the least amount of apps were from "Kids". For each category, we were able to fetch at most 250 apps and for some of them, we needed to search them using related terms. A total of 1,54,637 were scraped. 57,590 apps left after removing duplicates.

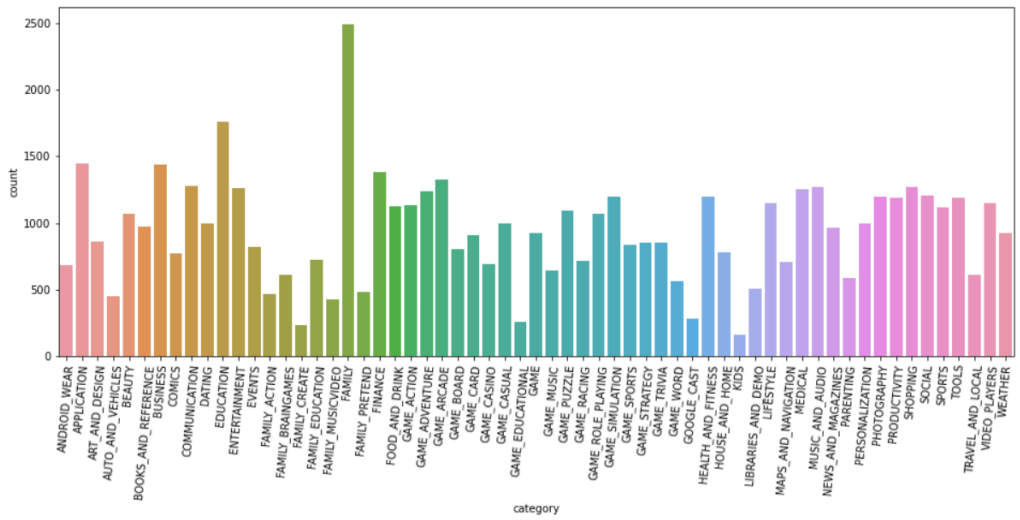

Figure 1: Category wise app distribution

3.2 Reviews Data

Reviews were scraped using google-play-scraper [6] written in Python. For each app, at most 1000 reviews were scraped batch-wise (200 reviews per batch) with a disk size of 10GB.

3.3 Features

The columns that we considered for analysis are described below:

3.3.1 App Data

- **AppId:** Unique Id for each app.
- **Price:** Price of an app. The currency is Indian Rupee.
- **scoreText:** Rating of app shown on playstore.
- **category:** Category of an app out of 61 possible categories.

3.3.2 Reviews Data

- Reviewer Id was not available, so we used **Reviewer's Name** as the edge parameter.
- For the pair of apps which a reviewer reviewed, we have the **scores** given by the reviewer as an attributes.

3.4 Pre-Processing and Network Formation

For the pre-processing of data we used Python language.s Earlier we were using JSON format to store the preprocessed data but due to its huge memory requirements, we shifted to CSV format. We listed the apps reviewed by each reviewer and created a network. Each node within the network represents an app and each link represents a pair of apps that a single reviewer has reviewed. Multiple edges are possible in this network. The network statistics are as follows:

	Count	File Size
Nodes	57,588	22 MB
Edges	2,95,33,258	2.5 GB

Average degree: 967.0400

3.5 Data Representation: Edge List

The three most commonly used representations for network are Adjacency lists, Adjacency matrices, and Edge lists. Adjacency matrices take a huge amount of space (especially for such a large network). Among Adjacency lists and Edge lists, we chose Edge lists because of the simplicity they provide.

3.6 Graph Sampling

Due to the large amount of data, analysing the existing network is a difficult job. It becomes very time-consuming and computationally expensive to analyze the network. An intuitive solution is to analyze a sample of the network.

Using Graph Sampling, we pick a subset of vertices/ edges from the network. We used Random walk Induced Graph Sampling as implemented in [1] and studied in [7]. The final network statistics are as follows:

	Count	File Size
Nodes	11,453	4.3 MB
Edges	50,48,335	380 MB

Average degree: 881.5743

The category wise distribution is shown in fig 2. Most of the apps are from "Application" category, then from "Adventure and Action Games". The least are from "Kids" category again.

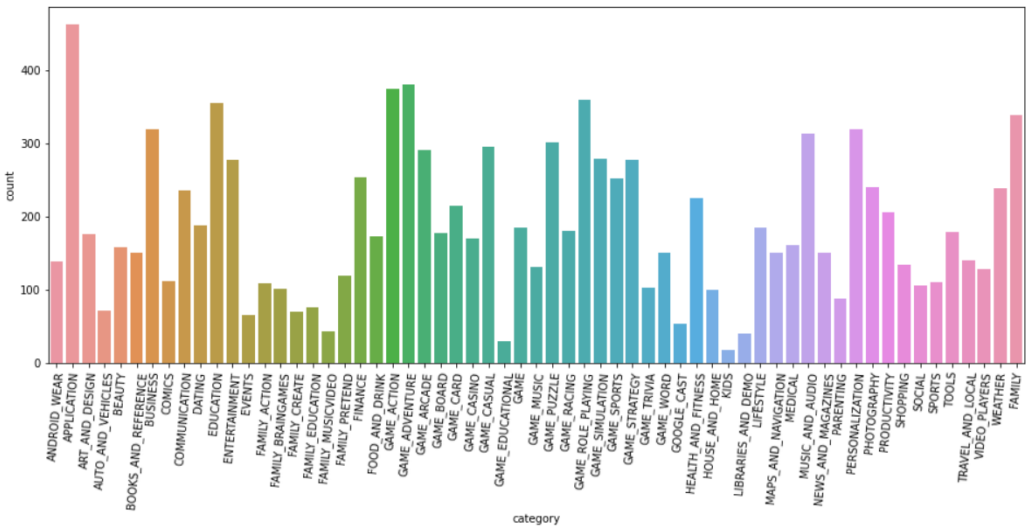

Figure 2: Category wise app distribution

4 Elementary Analysis of Original Network

Before analyzing the sampled network, we did some basic analysis (which the most we can do accounting for the time and memory constraints).

The network's density is the ratio of the actual edges in the network to all possible edges. The network density is 0.021. On a scale of 0 to 1, not a very dense network. Keeping in mind that this is the density of *whole* network, including disconnected components.

The network is disconnected with 8 connected components with the following statistics:

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
Nodes	45798	2	2	2	2	2	2	3
Edges	22151492	1	1	1	1	1	1	3

The density of the largest connected component is 0.021. From here we can conclude that the network is not very dense.

The maximum degree in the network is of "net.tsapps.appsales"(a shopping app) with degree 4211. Other hubs include "com.technologies.subtlelabs.doodhvale"(3906, food and drink), "com.snapbreak.doors"(3883, puzzle game), "com.gamexis.sniper.professional.action.game.apps"(3868, action game) "com.randomvideochat.livevideochat"(3856, communication). The degree distribution is shown in fig 3. Fitting the distribution to power-law gave a degree exponent of $1.17 < 2$ i.e. anomalous regime and not scale-free.

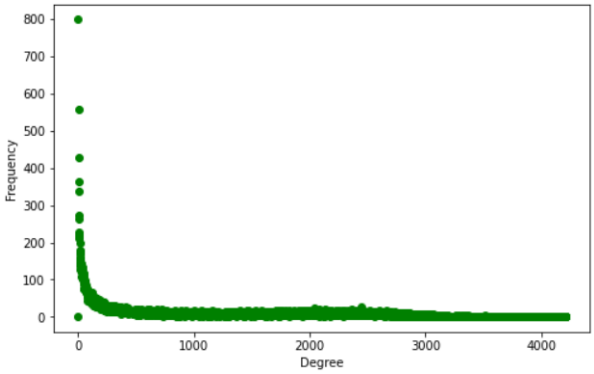

Figure 3: Degree Distribution of original network

5 Elementary Analysis of Sampled Network

5.1 Degree Centrality

The network is one single connected component with a density of 0.07. Again, not a very dense network but denser than the original one.

App Id	Price	Rating	Category	Degree	Eigenvector
net.tsapps.appsales	Free	4.3	Shopping	1914	0.018
com.snapbreak.doors	Free	4.5	Game Puzzle	1885	0.017
com.technologies.subtlelabs.doodhvale	Free	4.4	Food and Drink	1883	0.02
com.randomvideochat.livevideochat	Free	3.9	Communication	1848	0.019
com.socialnetwork.metu	Free	4.9	Social	1827	0.019

Table 1: Top 5 apps with highest degree centrality

App Id	Price	Rating	Category	Deg	Eigen
passport.Size.Photo.Maker.Editor.Countries	Free	0	Productivity	3	0
com.kiroglue.lookalikemomordadsimilarityparents	Free	4.0	Parenting	3	0
com.photovideomaker.slideshow.videostatusmaker	Free	0	Family Music	5	0
com.medpresso.Lonestar.manotes	Free	4.6	Medical	5	0
com.tutioncentral.cmanoteslite	Free	4.6	Medical	5	0

Table 2: Top 5 apps with lowest degree centrality

The highest reviewed app is a shopping app with a rating of 4.3, whereas the lowest reviewed app is a productivity app with a rating of 0. It seems that the more the number of reviewers, the higher is the rating. Although, from table 2 the two medical happens to have significant rating even though they have less degree.

The degree distribution is shown in fig 4. Fitting the distribution to power-law gave a degree exponent of $1.15 < 2$ i.e. again anomalous regime.

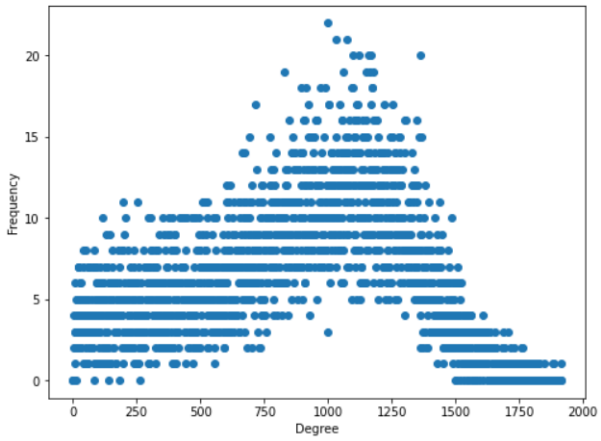

Figure 4: Degree Distribution of sampled network

5.2 Eigenvector Centrality

Eigenvector centrality, which is a kind of extension to degree centrality. It cares if an app is a hub, but it also cares how many hubs is the app connected to. The maximum eigenvector centrality value is 0.02 for the network.

App Id	Price	Rating	Category	Degree	Eigenvector
com.technologies.subtlelabs.doodhvale	Free	4.4	Food and Drink	1883	0.02
com.randomvideochat.livevideochat	Free	3.9	Communication	1848	0.019
com.socialnetwork.metu	Free	4.9	Social	1827	0.019
com.u2opia.woo	Free	4.2	Application	1814	0.019
com.edudrive.exampur	Free	4.2	Education	1787	0.018

Table 3: Top 5 apps with highest eigenvector centrality

The highest value of eigenvector centrality is very less on a scale from 0 to 1. We see that most of the apps with a high degree and eigenvector centrality are free apps and have a reasonable rating.

6 Community Detection in Sampled Network

We tested three different community detection algorithms for this project. The algorithms are based on various community detection concepts, each with its own set of performance characteristics. Each algorithm gave different communities which lead to interesting insights.

6.1 Clauset-Newman-Moore greedy modularity maximization

We use Clauset-Newman-Moore greedy modularity maximization algorithm from [5] to find communities in the network. The CNM algorithm has a time complexity of $O(Nlog^2N)$ [3] where N is the number of nodes in the network.

The algorithm found 4 communities of sizes 6017, 5432, 2, and 2. Both the communities with size 2 have an edge between them. The average rating for each category in the two largest communities is shown in fig 5. Although, no general trend can be inferred the average rating of all the apps in both the communities is 4.19 and 4.11.

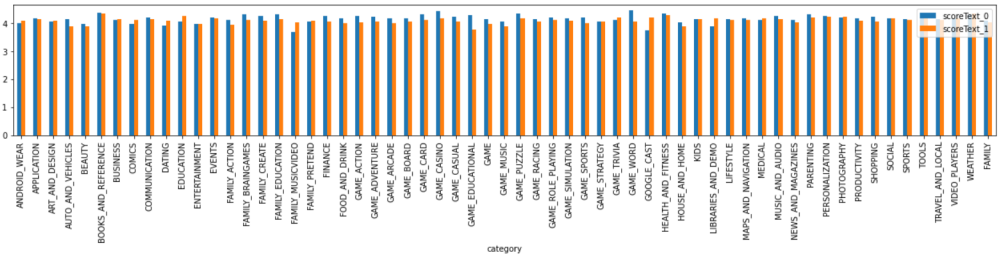

Figure 5: CNM community wise average rating per categoy

If we look at the average price in both the communities 6, we notice that most of the paid apps are in community c_1(size 6017). We did the same for average rating but didn't found anything promising.

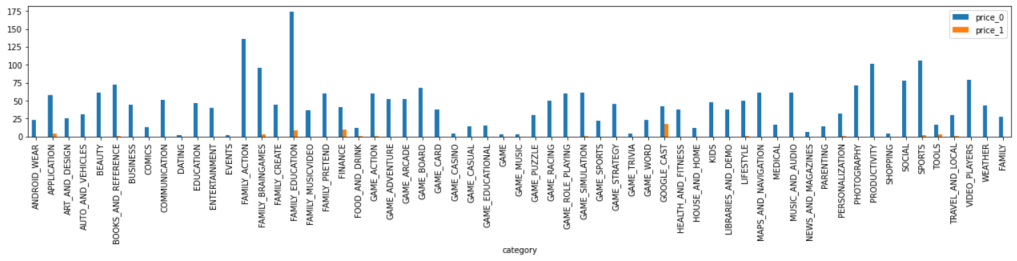

Figure 6: CNM community wise average price per category

6.2 Louvain method

The Louvain method is a method to extract communities from large networks. The inspiration for this method of community detection is the optimization of modularity as the algorithm progresses. The method has a time complexity of $O(L)$ [2] where L is the number of links in the network.

The algorithm found 3 communities of sizes 4329, 6114, 1010. The average rating didn't give many insights but looking at the community wise average price per category fig 7, most of the paid apps are in c_2(size 4329) with "Finance" and "Tools" as exceptions in c_3 (size 1010). We found that there are only 7 apps in c_3 out of 254 of the FINANCE

category out of which 3 are paid (₹470, ₹700,₹700). The most expensive app in the whole dataset is ₹920. Similarly, with the TOOLS category, there are only 6 apps in c_3 out of 179. 2 apps are paid. In both the categories, the number of apps is less whereas their prices are high.

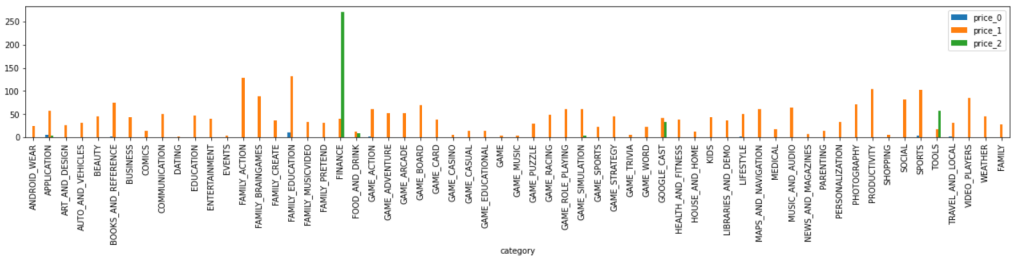

Figure 7: Louvain community wise average rating per category

6.3 Infomap Community Detection

Infomap exploits data compression for community identification. It does it by optimizing a quality function for community detection in directed and weighted networks, called the map equation. The algorithm runs in $O(NlogN)$ with flow optimization [11].

The algorithm found 3 communities with sizes 10531, 920, 2. Looking at the community-wise average rating per category fig 8, we do not find any general trend, but the average rating for the largest two communities is around 4.15 and 4.2. The algorithm did not put any apps in the c_2 (size 920) from the categories "ANDROID WEAR", "GAME CASINO", "KIDS", "LIBRARIES AND DEMO" and "NEWS AND MAGAZINES".

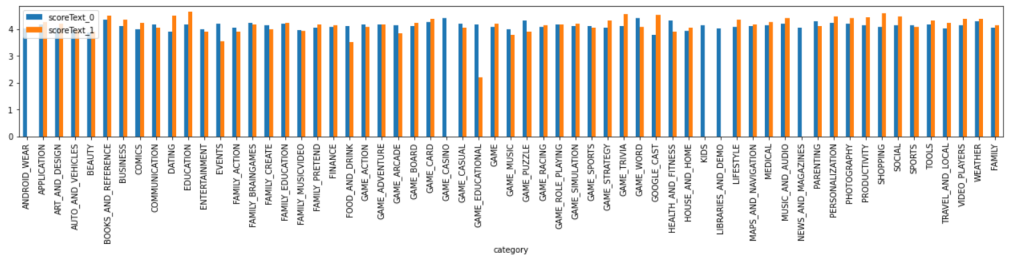

Figure 8: Infomap community wise average rating per category

Looking at the community wise average price per category 9, again the prices of c_1 (size 10531) are higher than c_2, with the same exceptions as for Louvain i.e. FINANCE and TOOLS.

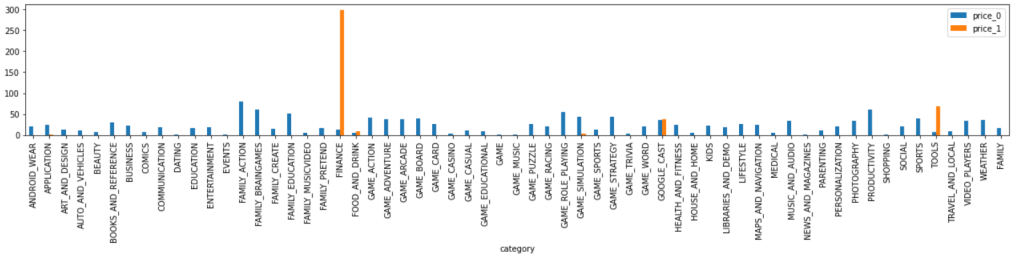

Figure 9: Infomap community wise average price per category

7 Survey

We here summarize our findings from the 3 community detection algorithms. The sections below describe a list of statistics that we computed and analyzed.

7.1 Community Sizes

The communities sizes detected by the 3 algorithms are shown in fig 10. The size of the largest community predicted by CNM and Louvain is nearly 50% whereas for Infomap it is nearly 87% of the entire network and then drops quickly.

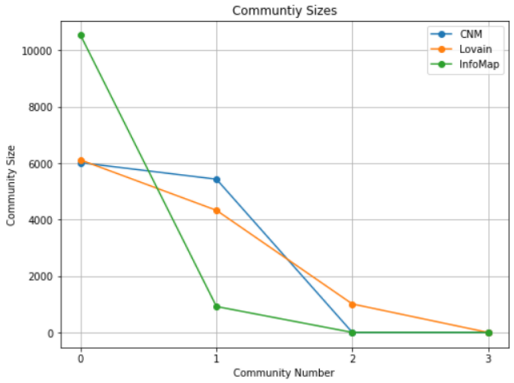

Figure 10: Community Sizes

7.2 Average Rating

The average rating of a community is calculated by averaging the rating of all the apps in that community. The average rating for community 2 is generally higher than other communities for all 3 algorithms. Although, there is no clear indication of any relationship between the apps in community 2 for all the algorithms, the rating of the largest community approximately coincides at 4.17 (4.19 for both CNM and Louvain, 4.15 for Infomap). This appears to indicate that a score of about 4.17 is preferable for apps downloaded in parallel.

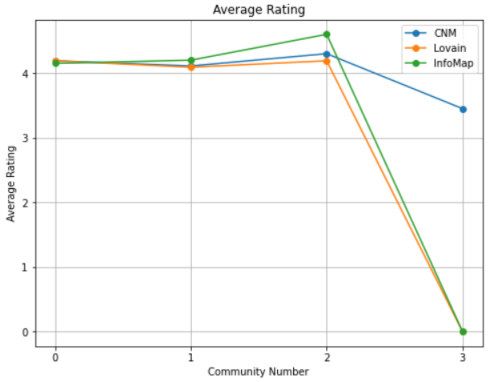

Figure 11: Average Rating

7.3 Average Price

Fig 12 shows the graph of the average price for each community for each algorithm. For all the algorithms, the average price of the largest community is much higher than other communities and then suddenly drops. For Louvain, the average price of community 3 is relatively high than CNM and Infomap and is comparable to the price of community 2 for CNM. This is due to the exceptions of apps in the FINANCE and TOOLS category.

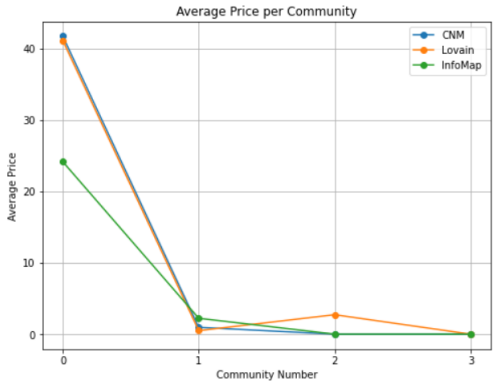

Figure 12: Average Price

7.4 Average Reviews

Average reviews for each community are the sum of degrees of apps contained in that community. Fig 13 shows the plot for the same. For all the algorithms, the number of reviews agrees at the largest community with approximately 869 reviews. There is a spike in communities 2 and 3 for Infomap and Louvain respectively. This suggests that the algorithms might have found some relationship between several reviews and other parameters which have caused the nodes to be in a specific community.

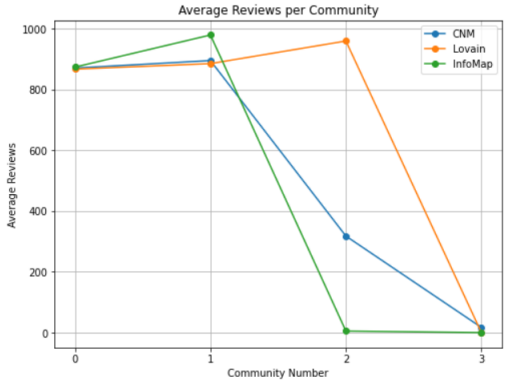

Figure 13: Average Reviews

7.5 Summary

In this sampled network, we were able to identify many interesting communities with distinctive properties. From the above analysis, it is clear that communities are not formed just based on the app's category but are a combination of various other factors including price and reviews.

From the community analysis, it seems that an app with a rating of near 4.17 and 870 reviews is more likely to get downloaded. Most of the paid apps have with a price near 40 are likely to get purchased. These findings can be used by Google play store to develop smarter recommendation engines or by developers to effectively advertise their application.

8 Conclusion

In this paper, we explored the network formed by the apps of the google play store. We formed a network of the data collected from the official play store website and analyzed a sampled network. We computed various centrality measures, checked scale-free nature, and tested a variety of network community detection algorithms. We were able to identify many interesting communities with unique traits.

The community detection algorithms we used were CNM, Louvain, and Infomap. Although the time complexities of these algorithms are similar, CNM took nearly 26 minutes to run whereas Louvain and Infomap were able to predict communities in nearly 5 minutes (tested on Google Colab).

Identifying communities and related transactions and downloads using these metrics help Google play store to develop a powerful recommendation system and developers to increase revenue.

9 Future Work

Many of the app data features were not used in this paper including the developer of an app, its description, and whether an app is trending/ grossing. Similarly, for the reviews data, the features like review content, the score given by the reviewer to an app, thumbs up on the review are also not used in our analysis.

We have worked on a sampled data. Techniques are available to shrink the size of network preserving the network properties. Working on the original network can give better insights. One can perform NLP on app's description and its review content also.

References

[1] Ashish Aggarwal. *Graph Sampling Package*. https://github.com/Ashish7129/Graph_Sampling.

[2] Vincent D Blondel et al. "Fast unfolding of communities in large networks". In: *Journal of Statistical Mechanics: Theory and Experiment* 2008.10 (Oct. 2008), P10008. ISSN: 1742-5468. DOI: 10.1088/1742-5468/2008/10/p10008. URL: http://dx.doi.org/10.1088/1742-5468/2008/10/P10008.

[3] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. "Finding community structure in very large networks". In: *Phys. Rev. E* 70 (6 Dec. 2004), p. 066111. DOI: 10.1103/PhysRevE.70.066111. URL: https://link.aps.org/doi/10.1103/PhysRevE.70.066111.

[4] Google. *Google Play Store*. https://play.google.com/store. 2008.

[5] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. "Exploring Network Structure, Dynamics, and Function using NetworkX". In: *Proceedings of the 7th Python in Science Conference*. Ed. by Gaël Varoquaux, Travis Vaught, and Jarrod Millman. Pasadena, CA USA, 2008, pp. 11–15.

[6] JoMingyu. *Google-Play-Scraper*. https://github.com/JoMingyu/google-play-scraper.

[7] Jure Leskovec and Christos Faloutsos. "Sampling from Large Graphs". In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '06. Philadelphia, PA, USA: Association for Computing Machinery, 2006, pp. 631–636. ISBN: 1595933395. DOI: 10.1145/1150402.1150479. URL: https://doi.org/10.1145/1150402.1150479.

[8] Julian McAuley, Rahul Pandey, and Jure Leskovec. "Inferring Networks of Substitutable and Complementary Products". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15. Sydney, NSW, Australia: Association for Computing Machinery, 2015, pp. 785–794. ISBN: 9781450336642. DOI: 10.1145/2783258.2783381. URL: https://doi.org/10.1145/2783258.2783381.

[9] *Number of free apps on Google play store*. 2021. URL: https://www.statista.com/.

[10] Facundo Olano. *google-play-scraper*. https://github.com/facundoolano/google-play-scraper. 2019.

[11] Martin Rosvall and Carl T. Bergstrom. "Maps of random walks on complex networks reveal community structure". In: *Proceedings of the National Academy of Sciences* 105.4 (2008), pp. 1118–1123. ISSN: 0027-8424. DOI: 10.1073/pnas.0706851105. eprint: https://www.pnas.org/content/105/4/1118.full.pdf. URL: https://www.pnas.org/content/105/4/1118.

[12] Shihui Song and Jason Zhao. "Survey of Graph Clustering Algorithms Using Amazon Reviews". In: ().