# AppNet: Understanding App Recommendation in Google Play

Qian Guo
Haoyu Wang
Beijing University of Posts and Telecommunications, China

Chenwei Zhang
Indiana University Bloomington, USA

Yao Guo
MOE Key Lab of High-Confidence Software Technology, Peking University, China

Guoai Xu
Beijing University of Posts and Telecommunications, China

## ABSTRACT

With the prevalence of smartphones, mobile apps have seen widespread adoption. Millions of apps in markets have made it difficult for users to find the most interesting and relevant apps. App markets such as Google Play have deployed app recommendation mechanisms in the markets, e.g., recommending a list of relevant apps when a user is browsing an app, which naturally forms a network of app recommendation relationships. In this work, we seek to shed light on the app relations from the perspective of market recommendation. We first build "**AppNet**", a large-scale network containing over 2 million nodes (i.e., Android apps) and more than 100 million edges (i.e., the recommendation relations), by crawling Google Play. We then investigate the "AppNet" from various perspectives. Our study suggests that AppNet shares some characteristics of human networks, i.e., a large portion of the apps (more than 69%) have no incoming edges (no apps link to them), while a small group of apps dominate the network with each having thousands of incoming edges. Besides, we also reveal that roughly 147K (7%) apps form a fully connected cluster, in which most of the apps are popular apps, while covering 97% of all the edges. The results also reveal several interesting implications to both app marketers and app developers, such as identifying fraudulent app promotion behaviors, improving the recommendation system, and enhancing the exposure of apps.

## CCS CONCEPTS

• **Information systems** → **Web searching and information discovery**; • **Human-centered computing** → **Ubiquitous and mobile computing**.

## KEYWORDS

App recommendation, Google Play, App Store, Android

**ACM Reference Format:**
Qian Guo, Haoyu Wang, Chenwei Zhang, Yao Guo, and Guoai Xu. 2019. AppNet: Understanding App Recommendation in Google Play. In *Proceedings of the 3rd ACM SIGSOFT International Workshop on App Market Analytics (WAMA '19), August 27, 2019, Tallinn, Estonia.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3340496.3342757

## 1 INTRODUCTION

The mobile app ecosystem grows rapidly in recent years [10]. With millions of apps in the market, *one of the biggest issues for app developer is to get their apps discovered by app users*, as more exposure will introduce more potential users. In general, mobile apps could be discovered mainly in three ways: *browsing*, *searching* and following *app recommendation.* The first is to *browse* diverse app categories and recommendation lists (e.g., most popular game apps) provided by app markets. The second is to discover apps through app store *searches* by generating keywords. The last one is to click the *recommended* related apps shown on the web pages of other apps (e.g., "similar apps" recommended in Google Play).

Although browsing app categories and searching with keywords are general mechanisms that are typically straightforward, the large number of mobile apps makes it difficult for users to locate relevant apps. Especially when considering that homogenization is a trend in the mobile app ecosystem, e.g., thousand of apps perform similar functionalities, it is not easy for users to choose the desired one. Therefore, recommending apps becomes an urgent task. App recommendation by the market could be naturally adopted by mobile uses, thus most app markets have applied some kind of recommendation mechanisms. Typically, app markets will recommend a list of related apps when a user is browsing an app. For example, Google Play would recommend some "Similar Apps" on the display page of each Android app. **The recommending relations between these apps thus form a huge recommendation network for millions of Android apps.**

The app recommendation mechanism by the market has been studied and exploited by many parties. For example, app store optimization techniques (ASO) become popular recent years, which is the process of optimizing mobile apps (e.g., optimize the title, description and keywords used in an app) to make it more visible to potential customers (e.g., appearing in the recommendation sections of many other apps, or ranking high in the searching results). However, the actual recommendation algorithms are not made public by app markets. In this paper, we are not focusing on how app markets recommend relevant apps, instead we seek to understand the network formed by these recommendation relationships: like social network or information networks, **app network itself is a very interesting subject to investigate**. It is still unknown to us that *how efficient is the recommendation mechanism and whether it could be exploited by spamming or malicious developers.*

In this paper, we present the first large-scale explorative study to characterize and understand the app recommendation network formed by app recommending relationships at Google Play. To be specific, we have created a snapshot of Google Play with over 2 million apps and over 100 million "similar app" relations (cf. **Section 3**), which forms **AppNet**, a large-scale app recommendation network. We then characterize the AppNet from various dimensions (cf. **Section 4**) and explore its network structures (cf. **Section 5**). Among many interesting findings, the following are the most prominent:

- Although each app recommends a limited number of apps and the distribution is relatively normal, the number of recommendations received by each app is not even at all, with 70% of the apps receiving no recommendation links at all (i.e., *the forgotten majority*), while some apps receiving as much as thousands of recommendations (the "star" apps).
- We find the AppNet is a *scale-free network*, which also demonstrates the *small-world effect*. It shares similarities with a few notable networks, e.g., scientific collaboration network [9], information networks such as the World Wide Web [1], and biological networks such as neural networks [13].
- Although 98.8% of apps are connected to each other with recommendation relationships when we considering recommendation as a two-way relationship, only a very small number of apps (7%) are strongly connected, which means a user can reach all of the apps when he/she starts from any of these apps. As these apps cover more than 97% of all the edges, they form the **core** of AppNet.

Besides these findings, We also study the detailed properties of AppNet on the distribution of app recommendations, i.e., how the recommendation relationship is affected by app properties such as category and the number of downloads, etc. Finally, we discuss the limitations and implications of our work, as well as how our efforts could contribute to the relevant stakeholders.

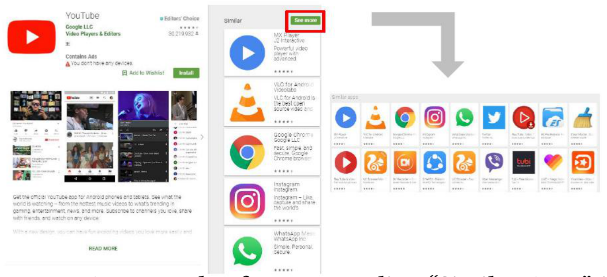
## 2 BACKGROUND AND RELATED WORK

### 2.1 App Recommendation in Google Play

As shown in Figure 1, Google Play displays the information (i.e., metadata) of an app on the description page, including app name, categorization information, the number of app installs, app descriptions, etc. As many apps perform the same or similar functionalities, to diversify users' choices, Google Play offers a variety of app recommendations. When browsing the Google Play app store, we can download popular apps on top charts page, find latest apps on new releases page, and discover certain apps by choosing category from the main screen. Besides, Google Play also provides a set of "similar apps" based on their own recommendation mechanism, which are displayed on similar apps section of the description page.

Such app recommendation mechanism can increase the exposure of some marginalized apps, making them to be more visible, thus reaching potential users. It also enables mobile users to obtain the target apps when they browse and search apps.

Figure 1 presents an example of recommending similar apps by Google Play. The description page shows detailed info of app *Youtube*, including its name, category, app description, screenshots and other relevant information. Google Play recommends series of "similar" apps based on app information and displays the similar



Get the official YouTube app for Android phones and tablets. See what the world is watching -- from the hottest music videos to what's trending in gaming, entertainment, news, and more. Subscribe to channels you love, share with friends, and watch on any device.

Figure 1: An example of recommending "Similar Apps" in Google Play (app "com.google.android.youtube").

section on the right side of the page. Generally, there are only a few apps listed in this section, but more similar apps will be recommended after clicking "see more" button.

### 2.2 Related Work on App Recommendation

Due to the huge and still rapidly growing number of mobile apps, app recommendation is necessary and thus en emerging research area. For example, Liu et al. [7] proposed to perform app recommendation based on both interest-functionality interactions and users' privacy preferences. Djinn [6] was introduced to support real context-aware recommendation that utilizes the diverse range of implicit mobile data available in a fast and scalable manner. SimApp [4] was proposed to identify similar apps using machine learning techniques, which could also be used in app recommendation. All of the above studies focus on personal app recommendation, i.e., recommending apps based on users' interests and behaviors, which is totally different with the market-level recommendation studied in this paper, although similar approaches could be used to understand the Google Play's recommendation mechanism.

## 3 CONSTRUCTING APPNET

### 3.1 Dataset

We implemented a crawler to harvest Google Play in November 2018. To crawl as many apps as possible, we have applied a combination of different strategies. We first used a list of 1.5 million package names [12] as the *searching seeds*, and then use a *breadth-first-search (BFS) approach* to crawl (1) a list of "similar apps" recommended for each app of our seeds by Google Play and (2) other apps released by the same developer. To further identify apps or small groups that have poor connectivity with other apps, we further applied a *keywords-based searching approach* to crawl apps by summarizing a list of keywords from app descriptions. Note that Google Play could be accessed in different world regions, we have instrumented our crawler to support both English and Chinese languages.

At last, we have created a list of 2.08 million of apps and 131 million app relations, which represents the most number of apps we could crawled in English and Chinese regions. We believe this dataset is representative enough to study the app recommendation mechanisms in Google Play.

### 3.2 Constructing the Network

We then construct the app relation network **AppNet**. In AppNet, each node represents an app, and each directed edges indicates the *similar app recommendation relation* between two apps. Note

**Table 1: Overall statistics of AppNet.**

| | |
|---|---|
| # nodes | 2,084,946 |
| # Directed edges | 131,723,578 |
| Average number of degree per node | 126.36 |

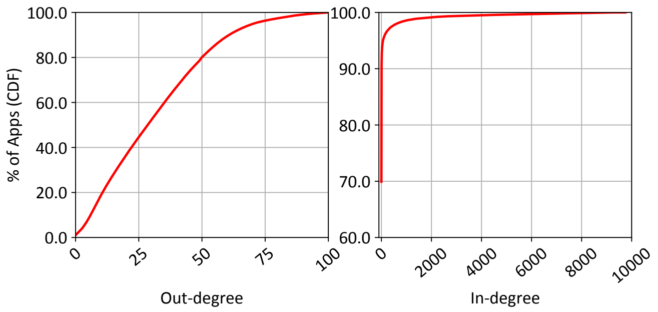


**Figure 2: Out-degree and in-degree distribution of AppNet.**

that the app recommendation relation is one-way. For example, a directed edge from $x$ to $y$ indicates that app $y$ is recommended on the webpage of app $x$, i.e., app $y$ is one of the similar apps of app $x$. However, it does not necessarily hold conversely.

**Basic Statistics.** As shown in Table 1, there are over 2 millions nodes and 131 millions edges in the resulting AppNet. The average number of degree per node is 126.36, which means that the average number of similar relationship per app is about 126, including being recommended and recommendation. To our surprise, the number of nodes with 0 in-degree is more than 1.4 million, which means that up to 70% of apps are not recommended by any apps, among which 21,456 are actually isolated nodes. This result suggests that *over 70% of apps in Google Play are unreachable though similar app recommendation and may be invisible to users for most of the time*, except when users search its name or using some specific keywords.

## 4 CHARACTERIZING APPNET

### 4.1 Distribution of Node Degrees

In AppNet, the **in-degree** of one node represents the frequency that such an app has been recommended as similar apps to the searchers of other apps, and the **out-degree** indicates the number of other apps recommended to this one. The distribution of in-degrees and out-degrees of the 2.08 million nodes in AppNet is shown in Figure 2. When investigating the distributions of the nodes' in-degrees and out-degrees, quite different patterns are revealed.

### 4.2 Understanding Out-degrees

The out-degree ranges from 0 to 200 in Google Play, which means that Google Play could recommend up to 200 similar apps for a given app. As shown in Figure 2(1), although 200 is the upper limit of similar recommendations, over 95% of nodes have out-degrees lower than 150. The average out-degree is 63.18. It is interesting to see that, more than 21,000 apps (1%) have 0 out-degree, which means no similar apps are shown on the pages of these apps. For example, app "com.tubemote.app" is a popular app with over 100 million installs, while it has no similar apps on its Google Play webpage by the time of our crawling. Because the total number of these apps are relatively small, we will consider them as outliers,

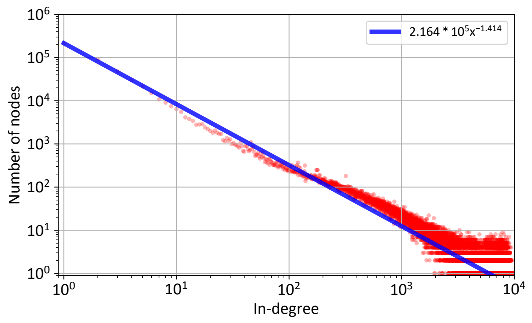


**Figure 3: Log-log scale plot of in-degree distribution.**

which will not affect the overall findings of our study. However, when we further manually investigated these apps (by checking the corresponding app descriptions and other metadata), we suspect that a majority of these apps are newly uploaded to Google Play, and some of them have poor app descriptions. As a result, Google Play does not recommend "similar apps" related to them.

### 4.3 Understanding In-degrees

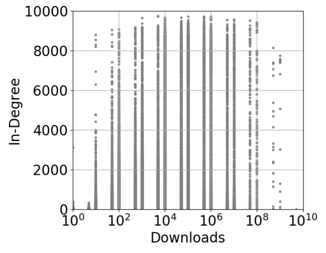
The distribution of in-degrees, which has a fat-tail, is very uneven compared to the distribution of out-degrees. It is interesting to see that, roughly 70% of apps have 0 in-degree, and over 90% of nodes have an in-degree less than 7. As a contrast, a few nodes have significantly high in-degrees, for example, more than 1,895 apps have in-degrees higher than 8k. As the in-degree can be defined as an app's exposure rate, it indicates that a small fraction of apps are more likely to be recommended on many apps' pages and thus easy to reach potential users, while the majority of apps are invisible to users and hard to be find.

We further draw a log-log plot of in-degree distribution, as shown in Figure 3, where the heavy tail characteristics can be seen intuitively. The exponent of this plot is found to be at around 1.414. The power-law distribution of in-degrees reveals that **AppNet is a scale-free network**, which has been the focus of a lot of research in the literature, such as the World Wide Web [1].

According to Barabási and Albert [2], the power-law distribution can be generated by a process of preferential attachment, which means those highly visible (high in-degree) apps will continue to gain more visibility. Due to the power-law distribution structure, the connectivity will not change a lot when random apps are removed from AppNet. However, the whole network will be largely affected if the removal is targeted at the top one percent high in-degree apps, since top 1.4% of the nodes in the AppNet account for up to 79.2% of the incoming links.

In AppNet, the distribution of in-degrees is quite different from that of out-degrees. The top 1.4% nodes mentioned above only account for 2.19% out-coming links (VS. 79.2% of the incoming links). This means that the incoming hubs in AppNet, which are those apps with very high exposure, may not be the outgoing hubs, which recommend a lot of similar apps.

*4.3.1 In-degree vs. App Metadata.* As the in-degrees of nodes in AppNet represent the exposure of apps to a certain extent, that is, the possibility that it can be seen. In this subsection, we further study the relationships between in-degrees and app metadata.


(a) App Downloads VS. In-degree

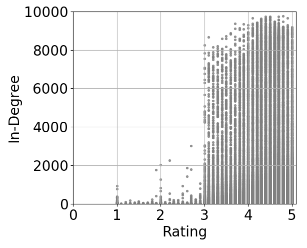

(b) App Ratings VS. In-degree

**Figure 4: App downloads/ratings VS. In-degree.**

**App Downloads.** The popularity of an app can be measured by the number of app installs. In order to obtain more downloads, developers will continually modify the metadata of apps (e.g., descriptions), to make the apps' ranking higher. This demand has even spawned plenty ASO tools that specifically help developers with leaderboards and search result optimization.

Figure 4(a) plots the relationship between in-degrees and the number of downloads in AppNet, where we can find that nodes with higher in-degree are more likely distributed in the area where the number of downloads ranging from 10K to 1M (i.e. medium level of downloads). We expect that the higher the exposure, the more downloads the app should have. However, among the 17,765 nodes with in-degrees higher than 2,000, 70.25% of them only have a medium level downloads. This result suggests that nodes with a medium level downloads might gain more exposure in the store.

**App Rating.** In general, the quality of an app can be also measured by app rating given by users. Figure 4(b) plots a scatter of the relationship between app quality and in-degrees, where we can see that nodes with high in-degrees tend to be distributed in areas with high ratings. More specifically, about 88.4% of the nodes with in-degrees greater than 2,000 have a user rating greater than score 4. Conversely, apps with higher user ratings do not necessarily receive higher in-degrees, as they are dispersed along a wide range of in-degrees. This result suggests that there is no strong correlation between app quality (user rating) and an app's in-degrees.

**App Category.** As mentioned earlier, the majority of nodes (69.9% in AppNet) are forgotten because no nodes point directly to them. To explore the distribution of such exposures in each category, we analyzed the distribution of zero in-degree nodes across different app categories, as shown in Figure 5. Due to space limitation, all the game sub-categories were merged into one category.

In most categories, the proportion of nodes with an in-degree of 0 is about 60%, ranging from 57.4% to 83.4%. Among these categories, the highest proportion exists in *Game* category. This result suggests that most apps across diverse categories do not gain much exposures and may be overlooked by users.

**App Update Time.** By looking into the data in Table 2, we found that all of the apps have been updated within a short time before our crawling process. We further study the relationship between the in-degree of an app and its update time. As shown in Figure 6, the update time of apps ranges from 2008 to 2018. Obviously, most of the apps with high in-degrees tend to be released/updated after 2018. For top 30k apps with the highest in-degrees, more than 93% of them were released after 2018 and more specifically, about 73% of them were updated within 3 months of our dataset collection.

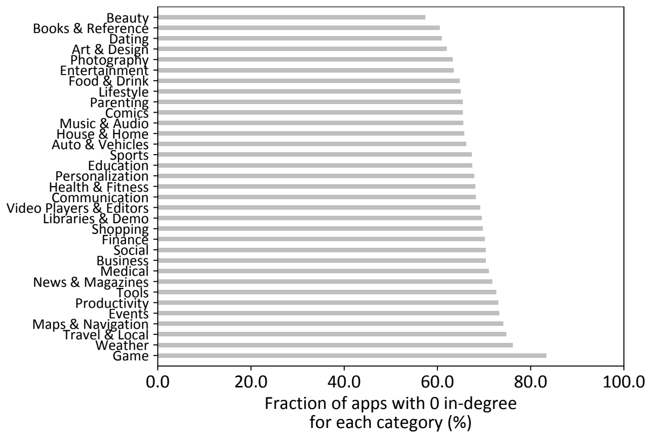


**Figure 5: Distribution of 0 in-degree nodes across categories.**

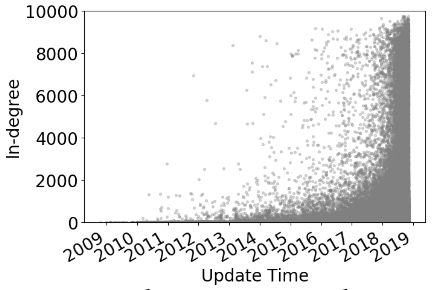


**Figure 6: In-degree VS. App update time.**

## 4.4 Characterizing the "Star Apps"

Table 2 shows the top 5 apps with the highest number of in-degrees. We can observe that all of them belong to `photography` category. Each of them has gained more than 9,000 incoming recommendations, but none of them has an out-degree of close to 200 (the maximum number of recommendation in Google Play).

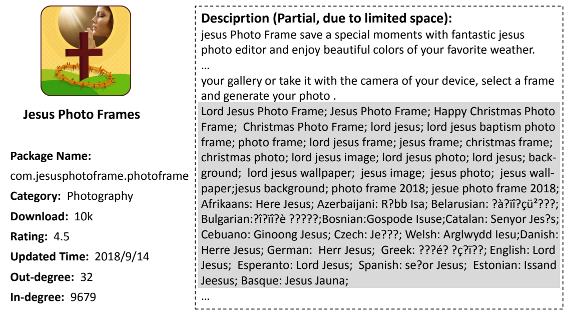
Note that the downloads of these 5 apps are not very high, only one of them has downloads higher than 1 million. Previous work [12] suggested that, only apps with installs higher than 1 million could be treated as popular apps, which means that most of these "star apps" are not popular apps. However, all these apps get high ratings: the ratings are all higher than 4.0 in a scale of 5.0. When focusing on their update time, all of them have been updated in the same period as our crawling process.

To further characterize these "Star Apps", we first manually inspect these apps. We found that plenty of them have a description sharing one remarkable feature, that is, listing many details (e.g., repetitive keywords) of an app. For example, *photo processing* apps usually list the filters that can be used in their description, *radio apps* have descriptions with a long list of available radio stations. Besides, some developers would like to insert irrelevant keywords (e.g., the names of popular apps and popular searching words) in their app descriptions, so that their apps would appear popular in the searching results or receive more recommendations.

However, Google Play specifies that *developers should not insert repetitive, irrelevant, misleading keywords and detailed information excessively in metadata, which will be regarded as spamming store listings.* Developers using such spamming techniques are mainly aimed at making their apps rank higher in user searching, but

Table 2: Top 5 apps with the highest in-degrees.

| Package name | out-degree | in-degree | # Downloads | Rating | # Rating | Category | Update time | Developer |
|---|---|---|---|---|---|---|---|---|
| com.pavahainc.fantasyframesforpictures | 127 | 9753 | 5k | 4.8 | 36 | Photography | 2018/11/2 | Pavaha Lab |
| com.sertanta.photoframes.photoframesspluscollage | 51 | 9741 | 100k | 4.7 | 391 | Photography | 2018/9/1 | Sertanta |
| com.pavahainc.sunsetframesforpictures | 121 | 9739 | 5k | 4.5 | 34 | Photography | 2018/9/19 | Pavaha Lab |
| com.dual.bookphotoframes | 132 | 9735 | 1M | 4.4 | 5451 | Photography | 2018/8/31 | Wallpaper Collection |
| com.skyinfoway.kidsphotoframes | 69 | 9724 | 500k | 4.5 | 1372 | Photography | 2018/9/13 | Sky Studio App |


**Jesus Photo Frames**

**Package Name:** com.jesusphotoframe.photoframe
**Category:** Photography
**Download:** 10k
**Rating:** 4.5
**Updated Time:** 2018/9/14
**Out-degree:** 32
**In-degree:** 9679

**Desciprtion (Partial, due to limited space):**
jesus Photo Frame save a special moments with fantastic jesus photo editor and enjoy beautiful colors of your favorite weather.
...
your gallery or take it with the camera of your device, select a frame and generate your photo .
Lord Jesus Photo Frame; Jesus Photo Frame; Happy Christmas Photo Frame; Christmas Photo Frame; lord jesus; lord jesus baptism photo frame; photo frame; lord jesus frame; jesus frame; christmas frame; christmas photo; lord jesus image; lord jesus photo; lord jesus; background; lord jesus wallpaper; jesus image; jesus photo; jesus wallpaper;jesus background; photo frame 2018; jesue photo frame 2018; Afrikaans: Here Jesus; Azerbaijani: R?bb Isa; Belarusian: ?à?ïî?çü²???; Bulgarian:?î?ïî?è ?????;Bosnian:Gospode Isuse;Catalan: Senyor Jes?s; Cebuano: Ginoong Jesus; Czech: Je???; Welsh: Arglwydd Iesu;Danish: Herre Jesus; German: Herr Jesus; Greek: ???é? ?ç?ï??; English: Lord Jesus; Esperanto: Lord Jesus; Spanish: se?or Jesus; Estonian: Issand Jeesus; Basque: Jesus Jauna;
...

Figure 7: Example of repetitive keywords in app description.

accidentally, their apps gain higher in-degrees in AppNet. Figure 7 presents an example. Due to the space limitation, only a part of the description is shown. App "com.jesusphotoframe.photoframe" has a long description with many keywords, which are relevant but repetitive. Apart from these repetitive keywords, the developer also tried to list keywords in different language. With more than 100 keywords listed, this app gains an in-degree up to 9,679.

Such kind of spamming-like descriptions may result in inaccuracies in similar app recommendation. When investigating 9,679 apps that recommend this app, we found that over 16% of apps do not share the same category with this one. It would make no sense for weather or keyboard apps to recommend this photo frame app. As for its outgoing recommendations, 8 out of 32 apps are not similar at all, including dictionary apps and keyboard apps. In this way, some spamming apps may become very popular "Stars" in AppNet.

We then try to measure how many apps in this star app group are spamming-like. We found that some certain words were found in the descriptions of these apps, which are often followed by repetitive/irrelevant keywords, such as "app highlights", "key features", "tags", "keyword", etc. Thus, we first summarized a list of such words/word phrases, and then we checked how many apps have embedded such words in their descriptions and followed by a number of (possible) keywords (over 20 words). We believe these apps have a suspicion of keyword/detail stuffing. At last, we found that for apps with in-degree higher than 5,000, over 25% of them have been exposed behaviors that presenting repetitive/irrelevant keywords in app descriptions. For apps with in-degree higher than 2,000, over 20.4% have been exposed such behaviors. This result suggests that the recommendation system of Google Play has been manipulated by some spamming developers.

## 4.5 Characterizing the Forgotten Majority

From the degree distribution of the AppNet, we found that up to 90% of the nodes have very low in-degrees and about 70% of them have zero incoming recommendations. It seems that these apps are forgotten during similar app recommendation in Google Play. However, to our surprise, some of them have millions of app installs.

Table 3: Top 5 popular apps with zero in-degree.

| Package Name | # Downloads | Rating | Category |
|---|---|---|---|
| com.google.android.gms | 5B | 4.0 | Tools |
| com.sec.spp.push | 1B | 4.1 | Communication |
| com.skype.raider | 1B | 4.1 | Communication |
| com.facebook.orca | 1B | 4.1 | Communication |
| com.facebook.katana | 1B | 4.1 | Social |

Table 3 displays the top 5 of apps with zero in-degrees based on the number of downloads. These 5 apps are all high quality and popular apps but gained no exposure in AppNet. To further investigate the characteristics of these apps, we sort these apps by their popularity and quality. Several characteristics of these apps were found and they can be classified into the following categories.

**Apps providing basic services** Apps that fall into this category, which empower users to get more out of their devices, are generally developed by mobile phone manufacturers. They seek to provide various technical support for apps of their own. For example, *Google Play Services* (com.google.android.gms), developed by *Google* is an app with more than 5 billion downloads, which helps users update apps from Google Play. *Samsung Pay* (com.samsung.android.spay), a mobile payment app developed by *Samsung Electronics Co., Ltd.*, enables users to make electronic payments. These apps provide services that are compatible with certain mobile phones, and some of them are even pre-installed, such as *com.sec.android.easyMover*. As a result, with a relatively fixed number of potential users, their demand for such exposure may not be great.

**Well-known apps.** Some of these low in-degree apps have unique app names that consist of words created by developers or with new meanings. Besides, such apps usually rank high in both top charts and searching. These features make them easy to be found anyway. For instance, *Skype* (com.skype.raider) falls into this category. It has over 1 billion downloads, but gains 0 incoming recommendation in AppNet. While some apps in this category may not be that well-known, they are developed by a well-known developer, or their developers have at least one famous app. For example, *Rovio Entertainment Corporation*, a developer famous for a popular game *Angry Birds*, have 22 apps in our AppNet but none of them gains any incoming recommendations.

Apps that fall into both categories above usually have a high level of downloads, despite they gain zero exposure in AppNet. While apps in categories listed below tend to have very low downloads, some of them even have downloads lower than 1, although they have been released to Google Play for more than 2 years.

**Apps with poor descriptions.** As ASO tools spare no effort to emphasize the importance of description for an app, a high quality description can effectively improve the exposure of an app. When going through those apps with 0 in-degrees, we found that a number of them have low quality descriptions. For example, the description of *com.konversi.qpondoncr* only has a single letter "a", which is barely helpful for users to understand what this app is for. As a result, Google Play does not link it to any related apps.

**Apps used for temporary experiments.** Apps of this category are not aimed to get feedback from the market to make them better, they are usually simple experimental projects developed by novices or just developed to accomplish a specific function. For example, *com.app.mohamedgomaa.arabic_books* is an app that claims to test in-app bill and has only 1 download. In fact, there exists some overlap between this category with previous one, but apps falling into this category all intend to do only a temporary experiment. Particularly, we found some developer IDs that are only created to release such apps, such as *GPDC UX Tester* with 251 apps, *Test Developer : GPDC* with 188 apps. In this case, what such apps want is exactly no exposure, so they can do their own experiments.

## 5 NETWORK STRUCTURE

### 5.1 Small-world Effect

In complex network theory, the small-world effect refers to the fact that although the size of the entire network is huge, a very small shortest path can be found between each pair of nodes. As early as in 1967, Milgram [8] have found that two people could be reached by approximately six acquaintance links on average. It has been proven that many networks have such property. Networks with a small-world effect usually have a small mean shortest-path length but a not so small clustering coefficient.

The average distance between nodes in AppNet is 5.45 and the diameter is 21. We found that if any two nodes are reachable (when considering each recommendation link as directed) in AppNet, their distance will be very small, 75% of which is less than 7. In other words, among more than 2 million apps in the AppNet, it only takes about 6 steps on average for one app to reach any other non-isolated app by relying only on the recommendation relationship.

*Clustering coefficient* measures the degree of tendency that two nodes in a network cluster together. Based on Watts and Strogatz's [5] average of the local clustering coefficients of all the nodes in the network, we find that AppNet has a clustering coefficient of 0.342. In this sense, AppNet shares some similarities with other real-world networks, such as the Internet network [14].

### 5.2 Connectivity

We then study the Connectivity of AppNet based on its weakly connected components (WCC) and strongly connected components (SCC). WCC means that considering the directed network as an undirected one, for every pair of distinct nodes $x$ and $y$, there exists an undirected path from $x$ to $y$. In contrast, SCC considering all links as directed and represents such kind of components: inside each component, any other node is reachable when choosing a node as start. In other words, there is a directed path between any two nodes in such a component, and the subset of these nodes is not a part of some larger SCCs. Obviously, nodes belong to a SCC is connected more closely with the recommendation relationship.

**WCC partitioning.** We get a giant WCC with 2.06 million (98.8%) nodes, 20 WCCs with sizes smaller than 100 and 21k isolated nodes. It means that, except for a very few nodes, most of nodes are well connected to each other. If both forward and backward directed links are available, users could find nearly all nodes.

**Table 4: Distribution of the sizes of SCCs in AppNet.**

| Size of SCCs | # SCCs | Size of SCCs | # SCCs |
|---|---|---|---|
| 147,437 | 1 | 11-40 | 435 |
| 108 | 1 | 2-10 | 10,286 |
| 71-100 | 15 | 1 | 1,896,588 |
| 41-70 | 33 | | |

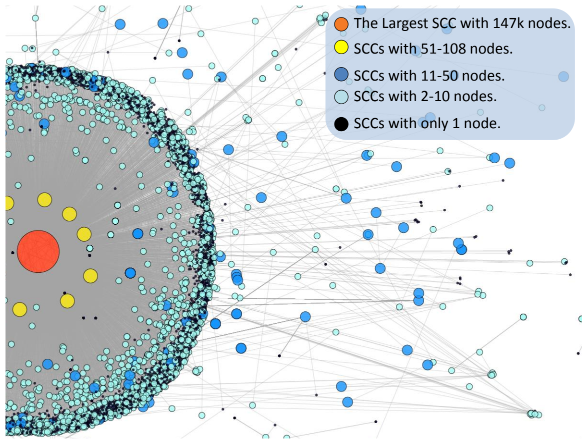


**Figure 8: Visualization of AppNet (part). Note that each SCC is concentrated into one node.**

**SCC partitioning.** AppNet can be partitioned into multiple SCCs, including only one large SCC with 147k nodes, about 10 thousand SCCs smaller than 110 and 1.9 million SCCs with only one node (cf. Table 4). To our surprise, this distribution also exhibits a *power law*. This feature is very similar to the structure of the web network, which was demonstrated by Broder *et al.* [3] that structure of the Web network is a "bow-tie" shape, consisting of a large SCC, two groups named IN and OUT. All the nodes of the group IN have a directed path to the largest SCC, and the group OUT contains the nodes that have directed paths from the largest SCC to any nodes in the group. The biggest difference is that two sides of these "bow-tie" shape in the Web network are balanced. While the size of the group IN in AppNet is much lager than that of the group OUT. Besides, the percentage of nodes in this central core of AppNet is much lower than that in the Web network.

The number of nodes in the largest SCC is 147k (7%), which is not very big compared with the size of the whole network. However, when considering the edges included in the largest SCC, 97% of edges in AppNet can be found in this SCC, among which 9.286% are internal edges, 0.002% are out-edges, and the remaining 87.731% are in-edges. This further indicates a serious imbalance between the group IN and group OUT in AppNet. After concentrating SCCs into nodes, we color the nodes according to the different sizes of the SCCs. Then we use tool Gephi to draw a part of AppNet, as shown in Figure 8. AppNet looks more like an annulus structure, with the largest SCC surrounded by nodes in form of small groups.

### 5.3 Characterizing the Largest SCC

We observed that most of the edges in AppNet are related to the largest SCC. Considering the out-edges of nodes in this component, somehow, most of them are links to the largest SCC, only about 3,000 (0.002%) out-edges are links to nodes outside the component.

**Table 5: The largest SCC vs. the overall AppNet.**

| | Overall | The Largest SCC |
|---|---|---|
| # Nodes | 2,084, 946 | 147,437 (7%) |
| # Edges | 131.7m | 127.8m (97%) |
| Average Downloads | 98,033 | 500,984 |
| Average Rating | 3.2 | 4.4 |
| Average In-degree | 63.18 | 866.77 |
| Average Out-degree | 63.18 | 82.99 |
| Average Distance | 5.45 | 5.34 |
| Clustering Coefficient | 0.03 | 0.23 |

For in-edges, they somewhat represent the exposure from the source node to the destination node in AppNet. As we discussed in Section 4.5, a certain number of well-known apps with high downloads only have 0 in-degrees, which is beyond our expectation. Note that these apps with 0 in-degrees cannot belong to this SCC, according to the definition of SCC. In this case, the largest SCC gain over 87% of the exposure in the whole AppNet.

To understand the nature of nodes in such a highly exposed group, we further study the differences between the largest SCC and the overall network. Table 5 compares the larget SCC and the overall network from app metadata and network features.

Even in the case where over 70% highly downloaded apps are not in the largest SCC, the average downloads of this component is five times of the overall network. In addition, we also calculated the average downloads of nodes with 0 in-degrees and the result is 80,961, despite over 62% highly downloaded apps are in this group. This may be due to the fact that the largest SCC gains a significantly high number of in-degrees (i.e., exposure), which is over one order of magnitude higher than the overall average. In terms of out-degrees, although the average out-degree is higher than that the overall network, most of the exposure gained is within the component, which also reflects that there exists an internal mutual promotion. So we positively assume that entering this highly exposed component is helpful for an app to increase its downloads.

From the perspective of the network structure, these two have a similar average distance. However, it should be noted that in the entire AppNet, the probability that exists a shortest path between two randomly chosen nodes is only about 7%, while any two nodes in the largest SCC are 100% reachable.

Since the average distance is smaller in the largest SCC and clustering coefficient is much more higher, the small-world phenomenon is more obvious. In other words, if an app can enter this SCC, it will significantly increase its in-degree and obtain higher exposure, which makes it more likely to be reached by users.

## 6 CONCLUDING REMARKS

In this paper, we take the first step to characterize and understand the app recommendation network formed by app recommending relations in Google Play. We first create AppNet, a recommendation network consisting of over 2 million apps and over 130 million app recommendation relations. We have observed the ineffectiveness of app recommendations in Google Play, and found that the scheme could be exploited by developers to intentionally increase the exposures of their apps. We further characterized the detailed network structure, and found several interesting observations. For example, AppNet is a scale-free network and it also has small-world effects.

As Google Play is always removing apps and introducing new apps in the market everyday [11], the main threat to validity in our work is that the constructed AppNet may change all the time. Furthermore, it is quite possible that app updates (e.g., modifying app descriptions) will affect the recommendation relations. Nevertheless, we believe the main observations (i.e., the structure and characteristics of AppNet) in this paper should be valid and will not change because of the volatile nature of the network itself.

We believe that our efforts and the revealed insights can contribute to different stakeholders of the mobile app ecosystem. *Market maintainers* should pay more attention the outliers (e.g., "star apps"), and improve the recommendation algorithm to make more apps gain more meaningful exposures. Our findings also suggested that some apps with no exposures are introduced by careless and unprofessional *app developers* (e.g., no or short app descriptions). Our work will help identify the weakness of their apps and further increase their exposures. Besides, as we have disclosed some fraudulent app promotion behaviors in this paper, these developers should pay more attention to their behaviors, as those behaviors should not be allowed by app markets and should be removed.

## ACKNOWLEDGEMENT

This work is supported by the National Key Research and Development Program of China (grant No.2017YFB0801903) and the National Natural Science Foundation of China (grants No.61702045 and No.61772042). Guoai Xu is the corresponding author.

## REFERENCES

[1] Réka Albert, Hawoong Jeong, and Albert-László Barabási. 1999. Diameter of the world-wide web. *nature* 401, 6749 (1999), 130.

[2] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

[3] Andrei Broder, Ravi Kumar, Farzin Maghoul, S Raghavan, P Rajagopalan, R Stata, A Tomkins, and J Wiener. 2000. Graph structure inthe Web: Experiments and models. In *Proceedings of the 9th World Wide Web Conference.*

[4] Ning Chen, Steven CH Hoi, Shaohua Li, and Xiaokui Xiao. 2015. SimApp: A framework for detecting similar mobile applications by online kernel learning. In *Proceedings of WSDM '15.* ACM, 305–314.

[5] Hawoong Jeong, Bálint Tombor, Réka Albert, Zoltan N Oltvai, and A-L Barabási. 2000. The large-scale organization of metabolic networks. *Nature* 407, 6804 (2000), 651.

[6] Alexandros Karatzoglou, Linas Baltrunas, Karen Church, and Matthias Böhmer. 2012. Climbing the App Wall: Enabling Mobile App Discovery Through Context-aware Recommendations. In *Proceedings of CIKM '12.*

[7] Bin Liu, Deguang Kong, Lei Cen, Neil Zhenqiang Gong, Hongxia Jin, and Hui Xiong. 2015. Personalized Mobile App Recommendation: Reconciling App Functionality and User Privacy Preference. In *Proceedings of WSDM '15.* 315–324.

[8] Stanley Milgram. 1967. The small world problem. *Phychology Today* 1, 1 (1967), 61–67.

[9] Mark EJ Newman. 2001. The structure of scientific collaboration networks. *Proceedings of the national academy of sciences* 98, 2 (2001), 404–409.

[10] Haoyu Wang, Hao Li, and Yao Guo. 2019. Understanding the evolution of mobile app ecosystems: A longitudinal measurement study of google play. In *Proceedings of the World Wide Web Conference.* ACM, 1988–1999.

[11] Haoyu Wang, Hao Li, Li Li, Yao Guo, and Guoai Xu. 2018. Why are Android apps removed from Google Play?: a large-scale empirical study. In *Proceedings of the 15th International Conference on Mining Software Repositories.* ACM, 231–242.

[12] Haoyu Wang, Zhe Liu, Yao Guo, Xiangqun Chen, Miao Zhang, Guoai Xu, and Jason Hong. 2017. An explorative study of the mobile app ecosystem from app developers' perspective. In *Proceedings of WWW '17.* 163–172.

[13] John G White, Eileen Southgate, J Nichol Thomson, and Sydney Brenner. 1986. The structure of the nervous system of the nematode Caenorhabditis elegans. *Philos Trans R Soc Lond B Biol Sci* 314, 1165 (1986), 1–340.

[14] Soon-Hyung Yook, Hawoong Jeong, and Albert-László Barabási. 2002. Modeling the Internet's large-scale topology. *Proceedings of the National Academy of Sciences* 99, 21 (2002), 13382–13386.