Quality Matters: Usage-Based App Popularity Prediction

Eric Malmi
Verto Analytics and Aalto University
Espoo, Finland
eric.malmi@aalto.fi

Abstract
In recent years, mobile application (app) economy has grown to a huge market but it is only the top apps that are able to turn this boom into significant revenues. In this paper, we study how the quality of an app, as reflected in how people start to use it, is linked to the popularity of the app. We show that features extracted from the Device Analyzer dataset, describing the aggregate usage of the app, can be used to predict its popularity. We also look at the connection between app popularity and the past popularity of other apps from the same publisher and find a surprisingly small correlation between the two.

Author Keywords
mobile applications, popularity prediction, Device Analyzer

ACM Classification Keywords
H.5.m [Information interfaces and presentation (e.g., HCI)]: Miscellaneous.

Introduction
There are currently over 1 million mobile apps in the Google Play Store but only a small fraction of these ever get discovered by the public. Indeed, according to some estimates the relationship between app downloads and their ranks is a power law [4]. If however, an app manages...
to climb to the top ranks, it has the potential of making huge revenues. For instance, the mobile game development company Supercell has been reported to make revenues of $2.4 Million per day with only two games in the market\(^1\). Therefore, numerous app developers are trying to find out how to get their apps to the top positions in the app ranking lists.

In this paper, we address the question of how well does the initial usage activity, which can be seen to reflect the quality of the app, predict the app’s popularity. To our knowledge, this is the first attempt in trying to quantify the relation between the quality and popularity of mobile apps. Additionally, we study the relation between the popularity of an app and the past popularity of other apps from the same publisher.

In order to analyze the initial usage of different apps, we use the Device Analyzer dataset \([11]\), which originates from a metering app that has collected fine-grained smartphone usage data from more than 17,000 volunteering participants. We extract 13 aggregate features for each app, describing, e.g., how long the app has been in the foreground. To measure the popularity of an app, we use data about top ranked apps in the Google Play Store, this information being collected directly from Google Play Store through publicly available application programming interfaces (APIs).

**Related Work**

App usage has been the topic of many studies in the past few years. Contextual usage patterns were studied, e.g., by Verkasalo \([10]\) and Böhmer et al. \([2]\). In the latter, the data was recorded by an Android app polling the list of recently started apps twice a second. Another dataset which is similar to the Device Analyzer dataset but with fewer participants is the Lausanne Data Collection Campaign (LDCC) dataset \([7]\). Using this dataset, Do and Gatica-Perez \([3]\) studied the prediction of the user’s next app given the context of the user.

Also some work has been done on the popularity of apps. Garg and Telang \([4]\) proposed a method for inferring app downloads based on the rank of the app, whereas Girardello and Michahelles \([5]\) presented AppAware, a platform for discovering mobile apps based on their current popularity. Our aim is to bring together the usage and the popularity aspect of apps.

Popularity prediction has been studied in several other domains such as news articles, Youtube videos, blog posts, and Tweets (see, e.g., \([1, 8, 12]\)). Typically, in this type of works the prediction of future popularity is done based on the early popularity \([1]\).

**Key Findings**

The key findings of this work are the following: (1) Device Analyzer data can be successfully used to measure various aspects regarding app usage. (2) We can predict the popularity of an app to some extent based on its first week initial usage. (3) The correlation between the popularity of an app and the popularity of other apps from the same publisher appears to be surprisingly small.

First, the measuring of how actively a given app is used is achieved by looking at the list of running processes that is polled every 5 minutes. The list contains the importances of the app processes which tell, e.g., whether an app is on the foreground or background. A limitation of this approach is that due to the relatively long polling interval, some app session are to be missed as an average app.
session lasts only 72 seconds [2].

Second, despite the limitations in the extracted usage features, we can show that they help us predict the app’s popularity, defined as the best rank obtained by the app. If we only limit ourselves to the apps between ranks 1–20 and 419–500, we can distinguish the popular ones with 70 % accuracy.

Third, we measure the popularity of other apps from the same publisher based on their best rank (best_other) obtained before the current app reached its best rank. We show that the app’s best rank and best_other are correlated but, quite surprisingly, the correlation \( \tau \) is only 0.14.

**Data Processing**

In this section, we describe how to extract app usage and popularity features from the Device Analyzer dataset and a rank dataset, respectively.

*Device Analyzer Data*

The Device Analyzer dataset [11] contains roughly around 100 billion records from 17 000 Android devices regarding various things such as app usage, phone calls, and Bluetooth data. For certain events, like incoming calls, the Device Analyzer app receives notifications from the system while other metrics, such as the list of running processes with their importances, are collected by polling them every 5 minutes\(^2\). A log file for each data collecting device is produced separately.

The dataset shared with the researchers outside the university of Cambridge does not contain any actual app (package) names for privacy reasons but the names are hashed separately for each device. However, for the Device Analyzer Workshop 2014, the organizing team provided an opportunity to write a script which extracts aggregated features from the unanonymized data files, send the script to the organizers to run, and receive back the summary statistics. We used this opportunity as we wanted to study how different people use a given app.

Our focus is on the app usage during the initial period of one week starting from the installation of the app. The installation times are available through the rows of a log file containing key “app|installed” which has the list of installed apps and the respective install times. The apps that have been installed before the first log entry was made were ignored since we could not get the data regarding their initial usage. Then we looked for the rows with key “app|[processID]” to collect the names and the importance values of the apps observed at different polls falling in the app’s initial usage interval. The importance value tells the status of the app process and it can take at least 5 different values\(^3\). In our case, we chose to focus only on the importance values foreground and background as we considered these the most relevant regarding to the user’s interaction with the app.

In Table 1, we have listed 13 features extracted from the data. For example, the first feature \( fg_{avg} \) reports the number of polls at which an app has had the foreground importance during the first week of usage, averaged over all users who had the app installed during their data collection. Multiplying this number with the polling interval of 5 minutes, would give us a rough estimate for how long the users actively used the app during the first

---

\(^2\)The list of collected key-value pairs can be found at: https://deviceanalyzer.cl.cam.ac.uk/keyValuePairs.htm

\(^3\)Detailed explanations for different app process importance values can be found at http://developer.android.com/guide/components/processes-and-threads.html
week after downloading it. Note that the initial periods of one week do not necessarily occur at the same time for different users.

Table 1: List of extracted features.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fg_avg</td>
<td>Average number of foreground importances.</td>
</tr>
<tr>
<td>fg_med</td>
<td>Median number of foreground importances.</td>
</tr>
<tr>
<td>fg_nz</td>
<td>The percentage for whom the app has been in the foreground more than once of the users who have the app installed.</td>
</tr>
<tr>
<td>bg_avg</td>
<td>Average number of background importances.</td>
</tr>
<tr>
<td>bg_med</td>
<td>Median number of background importances.</td>
</tr>
<tr>
<td>bg_nz</td>
<td>The percentage for whom the app has been in the background more than once of the users who have the app installed.</td>
</tr>
<tr>
<td>uniq_avg</td>
<td>Average number of unique process IDs (PIDs).</td>
</tr>
<tr>
<td>uniq_med</td>
<td>Median number of unique PIDs.</td>
</tr>
<tr>
<td>uniq_nz</td>
<td>The percentage for whom the app has had more than 1 unique PID of the users who have the app installed.</td>
</tr>
<tr>
<td>ses_avg</td>
<td>Average number of sessions for the app.</td>
</tr>
<tr>
<td>ses_med</td>
<td>Median number of sessions for the app.</td>
</tr>
<tr>
<td>ses_nz</td>
<td>The percentage for whom the app has had more than 1 session of the users who have the app installed.</td>
</tr>
<tr>
<td>fg_bg</td>
<td>Ratio of foreground/background importances averaged over users.</td>
</tr>
</tbody>
</table>

In addition to the foreground/background status, we computed the number of unique process IDs for an app indicating how many times the app has been killed and started again. Furthermore, the number of sessions for an app was computed by saying that a new session has started if the importance of the app’s process changes to foreground from any other importance. Finally, we calculated the ratio of foreground/background importances and averaged these ratios over the users.

It took about one week to run the feature extraction script in parallel. In total, 45,517 apps with at least one user installing the app during his data collection were found and out of these, 4,045 apps had at least 5 users.

**Rank Data**

Google Play Store publishes lists of top-500 apps in different categories and different countries. We focus on the top free apps as most of the apps nowadays are free to download. Also, we only consider the US ranking list since a significant part of the Device Analyzer users seems to be located in US⁴ and our general observation is that apart from the location and language specific apps, the same apps tend to be popular in all western countries. The popularity of an app is thus defined as the best rank it has obtained on the top free apps on the US ranking list. We have crawled the ranking lists, by doing requests through publicly available APIs directly from Google Play Store, on a daily basis from August 8, 2013 to July 1, 2014 observing 3,495 unique apps. Matching these with the apps discovered from the Device Analyzer data, we find 809 apps for which we have the best rank and at least 5 Device Analyzer users. The apps with less than 5 users were ignored since the aggregate features would be less reliable for them.

Predicting the ranks directly would result in a regression problem but instead, we transform the regression problem into a binary classification problem by splitting the apps into two classes. The apps are first ordered according to their best rank and then the top \( X \) th percentile is labeled as popular apps, the worst \( X \) th percentile as unpopular apps, and the apps in the middle are left out from the analysis. For instance, if we chose to look at the 20th percentiles, the popular apps would correspond to ranks 1–40 and the unpopular ones to ranks 357–500. The drawback of relaxing the problem by decreasing \( X \) is that we will have less data.

⁴https://deviceanalyzer.cl.cam.ac.uk/map/index.html
We are also interested in studying how much the popularity of an app can be explained by the popularity of other apps from the same publisher as it is intuitively easier to get users for a new app if the developer has already published other successful apps. We do not have the data about when different apps have been published but we can compute the best rank any of the other apps received before the current app reached its best rank. This feature is called best_other. Out of the 809 apps, 288 have at least one other app from the same publisher.

**Experimental Results**

**Methods**

Correlation between two ranking lists can be computed using the Kendall tau rank correlation [6] which measures the number of concordantly ordered pairs. The output value \( \tau \) is normalized to interval \([-1, 1]\), where \( \tau = 0 \) indicates no correlation, \( \tau = 1 \) that the lists are in the same order, and \( \tau = -1 \) that the lists are in the opposite order.

To solve the classification problem, we use the Support Vector Machine (SVM) classifier [9] with a Gaussian Radial Basis Function (RBF) kernel. The apps are split into a training set (70\%) and a test set (30\%). With the training set, we do 10-fold cross validation varying the scaling factor \( \sigma \) of the RBF kernel in order to find the optimal \( \sigma \). After this we train a new classifier using the whole training set and the selected value of \( \sigma \), and report the classification accuracy on the test set. Before doing the classification, we apply \( \log(x + 1) \) transform to all average and median features.

**Usage and Popularity**

The correlation between the apps ordered by each individual feature and by the app popularity are shown in Table 2. For each feature, we observe positive correlation which is statistically significant at 0.01 level (i.e., the null hypothesis \( \tau \leq 0 \) can be rejected as \( p < 0.01 \)). Based on the results, we can say that the features related to the foreground time seem to be most strongly related to app popularity but other potentially good predictors are the session features and fg_bg.

Then we study the classification performance using all the extracted features together while varying the percentile \( X \). The results, where performance is measured by accuracy, i.e., the percentage of correctly classified apps, are shown in Table 3. The total number of apps in the two classes is denoted by \( n \). The general trend is that the more we relax the problem by decreasing \( X \), the better classification accuracies we obtain. This supports the conclusion that the usage activity of an app tells about its popularity. For illustration, we show the classification boundary for the 20th percentiles using features fg_nz and \( \log(fg_{bg}) \) in Figure 1.

**Do the Rich Publishers Get Richer?**

Finally, we take a look at how the popularity (best rank) of an app is related to the popularity of the other apps from the same publisher (best_other). The Kendall tau rank correlation between these two variables is \( \tau = 0.13 \). The correlation is statistically significant (\( p < 0.01 \)), but quite surprisingly, it is smaller than the correlation for most of the usage features. Moreover, if we solve the classification problem setting \( X = 20 \) and using best_other as the only feature, we obtain 63\% classification accuracy, whereas fg_avg alone gives an accuracy of 66\%. These results suggest that having other popular apps might help a developer to make it to the top-400 but it is not enough when she wants to reach the best ranks.
Discussion and Future Work

It would be interesting to see how much predictive power could be obtained through other potential features, such as context information. Having more data than the 809 apps studied in this work would also be useful as it would allow us to study popularity prediction for different app categories separately. This should make the prediction task easier since app usage varies over different categories [2]. For example, in Figure 1, the incorrectly classified unpopular app with the highest log(fg,bg) is a personalization app used for creating a custom homescreen. This type of apps will naturally be often in the foreground even though they might not attract lots of downloads.

One of our conclusions is that we can predict the popularity of an app to some extent based on its initial usage. This result could have useful practical implications if we can assume that the initial usage statistics would look similar even if we only considered the users who installed the app right after it was launched. For example, a company that has recently released several apps and wants to understand which of them have the highest potential for success should look at how the apps are being used. The results presented in this paper can be seen as the first step in identifying what would be the most relevant features to look at and how popular the apps can be expected to become given their initial usage.

Acknowledgements

I would like to thank Andrew Rice and the Device Analyzer team for sharing the dataset and running the feature extraction script, and Timo Smura, Aristides Gionis, Hannu Verkasalo, and Matias Kontturi for insightful discussions and comments on the manuscript. I also thank Verto Analytics for providing some of the technical systems and data available to conduct parts of this paper’s empirical analysis.

References