Android Apps and User Feedback: A Dataset for Software Evolution and Quality Improvement

Grano, Giovanni; Di Sorbo, Andrea; Mercaldo, Francesco; Visaggio, Corrado A; Canfora, Gerardo; Panichella, Sebastiano

Posted at the Zurich Open Repository and Archive, University of Zurich
ZORA URL: [https://doi.org/10.5167/uzh-139426](https://doi.org/10.5167/uzh-139426)

Originally published at:
Android Apps and User Feedback: A Dataset for Software Evolution and Quality Improvement

Giovanni Grano  
University of Zurich  
grano@ifi.uzh.ch

Andrea Di Sorbo  
University of Sannio  
disorbo@unisannio.it

Francesco Mercaldo  
Ist. di Informatica e Telematica, CNR  
francesco.mercaldo@iit.cnr.it

Corrado A. Visaggio  
University of Sannio  
visaggio@unisannio.it

Gerardo Canfora  
University of Sannio  
canfora@unisannio.it

Sebastiano Panichella  
University of Zurich  
panichella@ifi.uzh.ch

ABSTRACT
Nowadays, Android represents the most popular mobile platform with a market share of around 80%. Previous research showed that data contained in user reviews and code change history of mobile apps represent a rich source of information for reducing software maintenance and development effort, increasing customers’ satisfaction. Stemming from this observation, we present in this paper a large dataset of Android applications belonging to 23 different apps categories, which provides an overview of the types of feedback users report on the apps and documents the evolution of the related code metrics. The dataset contains about 395 applications of the F-Droid repository, including around 600 versions, 280,000 user reviews and more than 450,000 user feedback (extracted with specific text mining approaches). Furthermore, for each app version in our dataset, we employed the Paprika tool and developed several Python scripts to detect 8 different code smells and compute 22 code quality indicators. The paper discusses the potential usefulness of the dataset for future research in the field.

Dataset URL: https://github.com/sealuzh/user_quality

CCS CONCEPTS
- Software and its engineering → Software development process management; Software evolution;

KEYWORDS
Software Quality, App Reviews, Mobile Applications, Software Maintenance and Evolution

ACM Reference format:
https://doi.org/10.1145/nmmmnn.mnmmm

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference’17, July 2017, Washington, DC, USA
© 2017 Association for Computing Machinery.
https://doi.org/10.1145/nmmmnn.mnmmm

1 INTRODUCTION
Mobile app stores, such as Google Play and Apple App Store, represent a rich source of information for software engineering researchers and developers interested in better understanding how mobile software is created and maintained [9], since these markets provide an open access to huge numbers of software applications together with consumers’ feedback [11]. Indeed, previous research showed that data contained in both user reviews and code change history of mobile apps represent a rich source of information for the development of mobile software, for reducing software maintenance effort and increasing customers’ satisfaction [5, 12, 14].

In this context users’ feedback are particularly important since software maintenance and evolution of mobile applications are strictly guided by requests contained in user reviews [3, 6, 11]. For instance, investigating the types of feedback users report on the apps they are using can give valuable information about the features on which they pay more attention or help better understanding the most common issues related to a specific app category [5, 11, 16]. Besides that, a more in-depth analysis on the code changes performed by developers when integrating users’ feedback in the code base of mobile applications can provide key insights on how developers evolve these apps to gain an higher customers satisfaction (e.g., for increasing downloads of a given app). Unfortunately, app stores lack functionalities to organize informative user reviews feedback toward proper software maintenance tasks or filter them according to the treated topics [1].

In this paper we propose a dataset containing 288,065 reviews extracted from Google Play related to 395 open source apps mined from F-Droid. Every review is connected with a specific version of the app and then split into atomic sentences. Each of the obtained sentences is labelled with the intention and the topics it deals with, relying on the two-dimension URM taxonomy proposed by Di Sorbo et al. [5]. Moreover, for each of the app versions, 22 code quality metrics (e.g., object-oriented and Android-oriented metrics) and 8 different code smells have been computed. The goal of this work is to provide data that researchers may promptly use to conduct experiments aimed at better (i) understanding how specific aspects related to code quality could affect app reviews and star-ratings, (ii) comprehending how developers react to specific user review feedback when evolving their mobile applications. To the best of authors’ knowledge, this is the first attempt to create

1https://play.google.com/store/apps
2https://f-droid.org
we analyzed the F-Droid repository and the Google Play store for (ii) combines such information with software quality indicators.

scripts/tools and labeled the extracted reviews through the use of package (i.e., the apk) of the mined apps using several static analysis collecting the app versions data and the information related to their community and, nowadays, there are already over 180 papers devoted to their study[9]. As a consequence, several datasets involving a quite high numbers of apps with structured (e.g., source code) and unstructured information (e.g., commits messages) have been proposed in the literature. For instance, the paper by Krutz et al.[8] provided a dataset that reports results obtained by several static analysis tools on 4,416 different versions of 1,179 open-source android applications combined with data of version control commits related to these applications. Collections containing huge amounts of app reviews have also been published for pursuing different research goals. For example, the Data Set for Mobile App Retrieval5 includes 1,385,607 user reviews of 43,041 mobile apps and it has been mainly used to run experiments about accuracy improvements in mobile app retrieval[15]. The SoftWare Marketplace (SWM) review dataset6 contains 1,132,373 reviews from 15,094 apps and has been involved in research works aimed at detecting spam or fake reviews[2, 18, 19]. Other existing public available data7 could be used to build and test sentiment analysis algorithms, since they contain reviews clustered according to the sentiment expressed in them (i.e., negative and positive sentiment). Nevertheless, to the best of our knowledge, no previous work provided a comprehensive dataset that, at the same time, (i) sheds the light on the types of feedback users report for different versions of several apps and, (ii) combines such information with software quality indicators computed on the app versions they are referring to.

2 RELATED WORK

Despite app stores represent a relatively recent phenomenon, they immediately captured the interest of the software engineering community and, nowadays, there are already over 180 papers devoted to their study[9]. As a consequence, several datasets involving a quite high numbers of apps with structured (e.g., source code) and unstructured information (e.g., commits messages) have been proposed in the literature. For instance, the paper by Krutz et al.[8] provided a dataset that reports results obtained by several static analysis tools on 4,416 different versions of 1,179 open-source android applications combined with data of version control commits related to these applications. Collections containing huge amounts of app reviews have also been published for pursuing different research goals. For example, the Data Set for Mobile App Retrieval5 includes 1,385,607 user reviews of 43,041 mobile apps and it has been mainly used to run experiments about accuracy improvements in mobile app retrieval[15]. The SoftWare Marketplace (SWM) review dataset6 contains 1,132,373 reviews from 15,094 apps and has been involved in research works aimed at detecting spam or fake reviews[2, 18, 19]. Other existing public available data7 could be used to build and test sentiment analysis algorithms, since they contain reviews clustered according to the sentiment expressed in them (i.e., negative and positive sentiment). Nevertheless, to the best of our knowledge, no previous work provided a comprehensive dataset that, at the same time, (i) sheds the light on the types of feedback users report for different versions of several apps and, (ii) combines such information with software quality indicators computed on the app versions they are referring to.

3 DATASET CONSTRUCTION

Our dataset was built in two phases: (i) in the data collection phase we analyzed the F-Droid repository and the Google Play store for collecting the app versions data and the information related to their user reviews; (ii) in the analysis phase we examined the Android package (i.e., the apk) of the mined apps using several static analysis scripts/tools and labeled the extracted reviews through the use of two automated classifiers.

3.1 Data Collection Phase

In this phase, we primarily built a web crawler (available in the dataset URL) to collect from the F-Droid repository the meta-data (package name, available versions, release date of each version) and the apks of each app. The crawler initially mined data for 1,929 different apps. The versions of each mobile application have been ordered according to the release date (i.e., from the oldest to the latest version). All the apps (i) not appearing in the Google Play Store and (ii) whose latest version was released before the year 2014 (i.e., this could indicate that the app is no longer maintained) have been discarded. A second scraper tool8 was built to download from Google Play Store all the user reviews related to the remaining 965 apps. It relies on Phantom JS9 and Selenium10 in order to navigate the Play Store web site and extract reviews from the resulting HTML code. We set up a cron job in order to mine new reviews 4 times a week. The tool totally gathered 297,323 app reviews, and for each user comment it also extracted (i) the package name of the app to which the review refers, (ii) the review content, (iii) the related star-rating assigned by the user to the app, and (iv) the posting date of the review. Relying on the release date of each applications’ version and on the review’s posting date of each user comment, we assigned each review to one of the app versions as described below. Given a generic version of an app, Vi and the next version of the same app, Vi+1, the reviews assigned to the version Vi, i.e., Ri, are collected considering the reviews whose posting date occur after the release date of Vi and before the release date of Vi+1. Despite this assumption may produce for some reviews an assignment to a wrong app version, Pagano et Maalej [10] empirically demonstrated that user feedback is mostly triggered by new releases, i.e., usually in the first few days after the download of a new app version. We discarded 8,758 reviews (because their publication date was too old for assigning them to any of the available versions) obtaining a dataset containing 288,656 reviews belonging to 710 different versions. Then we decided to keep in the collection exclusively the app versions having at least 10 reviews assigned (according to previous studies [17]), discarding all the remaining ones. At the end of this filtering process we obtained a dataset of 288,065 reviews related to 629 versions of 395 different apps.

Table 1: Intention Categories Definition

<table>
<thead>
<tr>
<th>Intention</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information</td>
<td>Sentences that inform other users or developers about some aspect of the app.</td>
</tr>
<tr>
<td>Giving</td>
<td>Sentences describing attempts to obtain information or help from other users or developers.</td>
</tr>
<tr>
<td>Information Request</td>
<td>Sentences expressing ideas, suggestions or needs for enhancing the app.</td>
</tr>
<tr>
<td>Problem Discovery</td>
<td>Sentences reporting unexpected behavior or issues.</td>
</tr>
<tr>
<td>Other</td>
<td>Sentences not belonging to any of the previous categories.</td>
</tr>
</tbody>
</table>

Table 2: Topic Definitions

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>App</td>
<td>sentences related to the entire app, e.g., generic crash reports, ratings, or general feedback</td>
</tr>
<tr>
<td>GUI</td>
<td>sentences related to the Graphical User Interface or the look and feel of the app</td>
</tr>
<tr>
<td>Content</td>
<td>sentences related to the content of the app</td>
</tr>
<tr>
<td>Pricing</td>
<td>sentences related to app pricing</td>
</tr>
<tr>
<td>Feature or Functionality</td>
<td>sentences related to specific features or functionality of the app</td>
</tr>
<tr>
<td>Improvement</td>
<td>sentences related to implicit enhancement requests</td>
</tr>
<tr>
<td>Updates</td>
<td>sentences related to explicit enhancement requests</td>
</tr>
<tr>
<td>Resources</td>
<td>sentences dealing with device resources such as battery consumption, storage, etc.</td>
</tr>
<tr>
<td>Security</td>
<td>sentences related to the security of the app or to personal data privacy</td>
</tr>
<tr>
<td>Downloaded</td>
<td>sentences containing feedback about the app description</td>
</tr>
<tr>
<td>Model</td>
<td>sentences reporting feedback about specific devices or app versions</td>
</tr>
<tr>
<td>Company</td>
<td>sentences containing feedback related to the company/team which develops the app</td>
</tr>
<tr>
<td>Other</td>
<td>sentences not fitting any of the previous topics</td>
</tr>
</tbody>
</table>

1https://sites.google.com/site/darhpark/Resources/data-set-for-mobile-app-retrieval
2http://odds.cs.stonybrook.edu/swmreview-dataset/
3https://github.com/amitt001/Android-App-Reviews-Dataset
4https://github.com/sealzhu/user_quality/tree/master/tools
5http://phantomjs.org/
6http://www.seleniumhq.org/
3.2 Analysis Phase

In this phase we classified user reviews’ feedback according to software maintenance and evolution categories and computed software quality indicators for each version of the mined apps. To achieve these goals, we employed some existing tools recently presented in literature. In the following we briefly explain how we performed the two tasks.

3.2.1 User Reviews Classification. In order to classify users’ comments according to software maintenance and evolution categories, we selected as a conceptual framework the URM taxonomy proposed by Di Sorbo et al. [5] which represents a robust and suitable model for representing user comments in meaningful maintenance tasks. This model assigns each review to (i) one of the intention categories showed in Table 1 and (ii) one or more topics detailed in Table 2. For performing the two-dimensions classification encompassed by the model, we employed:

- ARDOC, a reviews classifier previously defined by Panichella et al. [13], which combines Natural Language Processing (NLP), Sentiment Analysis (SA), and Text Analysis (TA) techniques in order to automatically mine intentions in user reviews, according to the categories defined in Table 1. This classifier has shown to achieve high precision (ranging between 84% and 89%) and recall (ranging between 84% and 89%) in categorizing reviews from real-word applications. We used for our purposes the original implementation of the tool, freely accessible as a Java library [13].
- The topic classifier module based on topics-related keywords and n-grams, used in the SURF summarizer tool[5], which is able to assign to each sentence in the review one (or more) of the topics defined in Table 2. This classifier has shown to achieve a classification accuracy of 76%.

3.2.2 Quality Analysis of Applications’ Code. To evaluate the code quality of Android applications, we developed Python scripts that compute a set of code quality indicators (all the quality metrics that our scripts are able to compute are detailed in the Appendix). In particular, the apks of all the mined versions with at least 10 reviews assigned have been disassembled, in order to obtain a set of human readable Dalvik bytecode .smali files from the binary Dalvik bytecode format .dex ones. To accomplish this task we used apktool10, a tool for reverse engineering which allows to decompile and recompile Android applications. Thus, we developed a set of Python scripts11 able to parse the .smali files and automatically compute the suite of code metrics for each of the available apks in our dataset. In our analysis, we compute the metrics by parsing .smali files (and not java ones), in order to consider code optimizations eventually applied by the compiler. In addition, we enrich our analysis by detecting code smells in the selected apks employing Paprika[7], a tooled approach which decompiles the classes (and not java ones), in order to consider code optimizations eventually applied by the compiler. In addition, we enrich our analysis by detecting code smells in the selected apks employing Paprika[7], a tooled approach which decompiles the application with Soot12 and performs the detection of 4 categories of Object-Oriented code smells (i.e., Internal Getter/Setter (IGS), Member Ignoring Method (MIM), No Low Memory Resolver (NLMR), Leaking Inner Class (LIC)). We computed and stored all the code smells above for each of the available versions of apps in our set, except for few ones that Paprika was not able to work with.

Table 3: Applications and Versions for each App Category

<table>
<thead>
<tr>
<th>App Category</th>
<th>Apps</th>
<th>Total apks</th>
<th>Total reviews</th>
</tr>
</thead>
<tbody>
<tr>
<td>Books &amp; Reference</td>
<td>19</td>
<td>35</td>
<td>15,892</td>
</tr>
<tr>
<td>Business</td>
<td>1</td>
<td>1</td>
<td>1,172</td>
</tr>
<tr>
<td>Comics</td>
<td>4</td>
<td>5</td>
<td>2287</td>
</tr>
<tr>
<td>Communication</td>
<td>33</td>
<td>64</td>
<td>31,219</td>
</tr>
<tr>
<td>Education</td>
<td>12</td>
<td>14</td>
<td>1,291</td>
</tr>
<tr>
<td>Entertainment</td>
<td>5</td>
<td>5</td>
<td>2,584</td>
</tr>
<tr>
<td>Finance</td>
<td>7</td>
<td>10</td>
<td>621</td>
</tr>
<tr>
<td>Games</td>
<td>30</td>
<td>44</td>
<td>20,378</td>
</tr>
<tr>
<td>Health &amp; Fitness</td>
<td>3</td>
<td>4</td>
<td>1,149</td>
</tr>
<tr>
<td>Libraries &amp; Demo</td>
<td>3</td>
<td>5</td>
<td>990</td>
</tr>
<tr>
<td>Lifestyle</td>
<td>4</td>
<td>6</td>
<td>246</td>
</tr>
<tr>
<td>Maps &amp; Navigation</td>
<td>10</td>
<td>15</td>
<td>1,411</td>
</tr>
<tr>
<td>Music &amp; Audio</td>
<td>17</td>
<td>32</td>
<td>3,025</td>
</tr>
<tr>
<td>News &amp; Magazines</td>
<td>6</td>
<td>9</td>
<td>1,988</td>
</tr>
<tr>
<td>Personalization</td>
<td>18</td>
<td>26</td>
<td>12,037</td>
</tr>
<tr>
<td>Photography</td>
<td>7</td>
<td>10</td>
<td>4,275</td>
</tr>
<tr>
<td>Productivity</td>
<td>45</td>
<td>80</td>
<td>8,361</td>
</tr>
<tr>
<td>Shopping</td>
<td>2</td>
<td>2</td>
<td>2,647</td>
</tr>
<tr>
<td>Social</td>
<td>7</td>
<td>11</td>
<td>6,146</td>
</tr>
<tr>
<td>Tools</td>
<td>139</td>
<td>214</td>
<td>151,509</td>
</tr>
<tr>
<td>Travel &amp; Local</td>
<td>9</td>
<td>12</td>
<td>984</td>
</tr>
<tr>
<td>Video Players &amp; Editors</td>
<td>12</td>
<td>22</td>
<td>15,352</td>
</tr>
<tr>
<td>Weather</td>
<td>2</td>
<td>3</td>
<td>2,501</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>395</strong></td>
<td><strong>629</strong></td>
<td><strong>288,065</strong></td>
</tr>
</tbody>
</table>

4 ANALYTICS & DATA SHARING

To offer a more complete overview of the dataset, in this section we provide some statistics about the collected data, as well as information about its final structure. In detail, Table 3 reports respectively the number of apps collected, the apks available and the reviews mined for each of the covered Google Play categories.

Analyzed apks contain a total of 100,638,277 byte-code instructions, 832,347 classes, and 6,375,906 methods. Each apk ranges from a minimum of 10 to a maximum of 103,535 reviews assigned. On average, each apk has about 458 reviews assigned and in turn each user comment is composed generally by 1.57 sentences. Furthermore, every one of them belongs to one of the intention categories (see Table 1), and, on average, deals with about 1.34 of the maintenance topics (see Table 2). To provide a more detailed description of the reviews’ dataset, Table 4 reports, for each of the maintenance topics, the amounts of (i) sentences discussing the topic, (ii) sentences of the Feature Request (FR) intention category dealing with the topic, (iii) sentences of the Problem Discovery (PD) intention category treating the topic, (iv) sentences of the Information Seeking (IS) intention category dealing with topic, (v) sentences of the Information Giving (IG) intention category discussing the topic, and (vi) sentences of the Other intention category treating the specific topic. Moreover, for the 629 apks of our dataset, we were able to detect a total of (i) 3,263 Blob Classes, (ii) 44,834 Long Methods, (iii) 432 Swiss Army Knives, (iv) 8,640 Complex Classes, (v) 9,012 Internal

---

2. https://ibotpeaches.github.io/Apktool/
conclusion of topics and their corresponding categories. Each table presents a set of related data points, with columns for App, Contents, Download, Company, Feature/Functionality, Improvement, Pricing, Resources, Update/Version, Model, Security, and Other. The sentences are categorized into 13 different topics, each with a specific number of sentences. The topics include App, Contents, Download, Improvement, Pricing, Resources, Update/Version, Model, Security, Other, and TOTAL.

The total number of sentences discussing each topic is as follows:

- App: 117,409
- Contents: 18,819
- Download: 7,853
- Company: 1672
- Feature/Functionality: 173,847
- Improvement: 8,281
- Pricing: 4,016
- Resources: 3071
- Update/Version: 21,669
- Model: 22,604
- Security: 2,392
- Other: 137,784
- TOTAL: 606,477

The dataset we provide comprises 395 different apps from F-Droid in a faster and more efficient way, increasing users' satisfaction. The data provided are useful for understanding potential correlations between the various collected data metrics, not only looking into individual apps, but also analyzing them in aggregation.

REFERENCES