App Miscategorization Detection: A Case Study on Google Play

Abstract:

App markets, where app developers can make their apps available to potential users, have been created by mobile device and platform manufacturers. The two dominant app markets, APPLE App Store and GOOGLE Play Store, each hosts over one million apps spread over 15+ categories [1], [2]. Now other players, including mobile operators, are also launching their own app markets. With the increasing popularity arises a concomitant downside that the app markets are becoming increasingly difficult to navigate and app categories harder to distinguish. As a result, app developers struggle to get visibility for their products. App search is not as advanced as web search and technology for "app search optimization" is still nascent. **An example of a miscategorization:** 2048 is a popular number puzzle game1. Several implementations of this game are available in GOOGLE Play Store. This example shows a scenario where a developer has published one implementation under Words category, which is not relevant.

Existing System:

An ongoing challenge in the rapidly evolving app market ecosystem is to maintain the integrity of app categories. At the time of registration, app developers have to select, what they believe, is the most appropriate category for their apps. Besides the inherent ambiguity of selecting the right category, the approach leaves open the possibility of misuse and potential gaming by the registrant. Periodically the app store will refine the list of categories available and potentially reassign the apps. However, it has been observed that the mismatch between the description of the app and the category it belongs to, continues to persist. Although some common mechanisms (e.g. a complaint-driven or manual checking) exist, they limit the response time to detect miscategorized apps and still open the challenge on categorization.

Proposed System:

We proposed FRAC+, (FR)amework for (A)pp (C)ategorization, to infer app categories and detect

miscategorized apps. The key ideas include: (i) expressing the app descriptions as normalized word frequency counts which are modeled using a topic model based on directional distributions, (ii) integrating existing app categories with inferred categories. We have shown that our topic model is able to effectively separate small number of apps that have different word distributions from the rest apps.

Modules:

- A framework for app categorization and miscategorization detection.
- Miscategorization Detection.

SYSTEM REQUIREMENTS

H/W System Configuration:-

Processor	-	Pentium –III
RAM	-	256 MB (min)
Hard Disk	-	20 GB
Key Board	-	Standard Windows Keyboard
Mouse	-	Two or Three Button Mouse
Monitor	-	SVGA

S/W System Configuration:-

Operating System	:	Windows95/98/2000/XP
Application Server	:	Tomcat5.0/6.X
Front End	:	HTML, Jsp
Scripts	:	JavaScript.
Server side Script	:	Java Server Pages.

Database : MySQL 5.0

Database Connectivity : JDBC