Research Summary

Safwat Hassan, Assistant Professor
Thompson Rivers University, BC, Canada
shassan@tru.ca

Mobile applications (*apps*) are becoming an integral part of our daily activities. The mobile market is continuously growing, with an expected $156 billion in revenue by 2023 [3]. With the mobile market's highly competitive nature, app developers need to avoid deploying buggy releases and continuously deploy high-quality releases that satisfy user needs. Over the years, the growing mobile apps market with millions of apps and billions of yearly downloads generates comprehensive data. Mining mobile app market data (e.g., analyzing user reviews) can help app developers improve their apps' perceived quality.

My research focuses on applying data mining techniques to the emerging software repositories (e.g., the app market data) to improve the perceived quality of software products. In the last six years, I have been working on providing techniques that can help app developers and store owners improve mobile apps' perceived quality in the following directions.

1. Analyzing Users' Feedback

We proposed approaches that help app developers proactively spot buggy releases and deploy fixes for such buggy releases [9]. Moreover, we proposed approaches to spot user complaints that may lead to user churn (i.e., migrating to competitor products) [7] and prioritize the required features based on the app's competitors [4]. We also proposed approaches to help app developers design mobile apps with a high-quality user interface (UI) [5].

2. Analyzing Developers' Common Practices

We studied the common practices of delivering main features in mobile apps, such as the patterns of writing release notes [14] and the patterns of integrating ad libraries [1, 2] and machine learning libraries [13]. We also identified the common mistakes that lead to emergency fixes in mobile apps [10]. This work highlighted the importance of analyzing the store data on a large scale to automatically identify the deployed issues and recommend solutions to the identified issues.

3. Studying the Interactions Between Users and Developers

Responding to user reviews can improve the ratings of an app. However, with many daily user reviews, it is not practical to respond to all reviews. Hence, we proposed an automatic approach to predict reviews that are most likely to get app developers' responses [11]. We also studied developers' interactions using development communication platforms, such as the Gitter platform [6]. We proposed an approach that identifies discussion threads in communication platforms. Project maintainers can benefit from our work to spot the commonly repeated discussions and use such discussions to improve the documentation of their projects. To ease the replication of our study, we shared our dataset in our replication package [6].

4. Improving the Performance of Software Systems

We are collaborating with IBM to provide approaches to enhance the performance of software systems [15, 8]. We also proposed different approaches to improve the design of software systems [12].

References

[1] M. Ahasanuzzaman, S. Hassan, C. Bezemer, and A. E. Hassan. A longitudinal study of popular Ad libraries in the Google Play Store. *Empirical Software Engineering*, 25(1):824–858, 2020.

[2] M. Ahasanuzzaman, S. Hassan, and A. E. Hassan. Studying Ad library integration strategies of top free-to-download apps. *IEEE Transactions on Software Engineering*, pages 1–12, 2020.

[3] B. O. Apps. App revenue statistics (2019). `https://www.businessofapps.com/data/app-revenues/`. (Last accessed: March 2019).

[4] M. Assi, S. Hassan, Y. Tian, and Y. Zou. Featcompare: Feature comparison for competing mobile apps leveraging user reviews. *Empirical Software Engineering*, pages 1–43, 2021.

[5] Q. Chen, C. Chen, S. Hassan, Z. Xing, X. Xia, and A. E. Hassan. How should I improve the UI of my app? a study of user reviews of popular apps in the Google Play. *ACM Transactions on Software Engineering and Methodology*, 30(3):1–38, 2021.

[6] O. Ehsan, S. Hassan, M. E. Mezouar, and Y. Zou. An empirical study of developer discussions in the Gitter platform. *ACM Trans. Softw. Eng. Methodol.*, 30(1):8:1–8:39, 2021.

[7] O. El Zarif, D. A. da Costa, S. Hassan, and Y. Zou. On the relationship between user churn and software issues. In *Proceedings of the 17th International Conference on Mining Software Repositories*, MSR '20, pages 339–349. ACM, 2020.

[8] O. El Zarif, S. Hassan, Y. Zou, C. Zuzarte, and V. Corvinelli. Pred-cache: a predictive caching method in database systems. In *Proceedings of the 30th Annual International Conference on Computer Science and Software Engineering*, CASCON '20, pages 93–102. ACM, 2020.

[9] S. Hassan, C. Bezemer, and A. E. Hassan. Studying bad updates of top free-to-download apps in the Google Play Store. *IEEE Transactions on Software Engineering*, 46(7):773–793, 2020.

[10] S. Hassan, W. Shang, and A. E. Hassan. An empirical study of emergency updates for top Android mobile apps. *Empirical Software Engineering*, 22(1):505–546, 2017.

[11] S. Hassan, C. Tantithamthavorn, C. Bezemer, and A. E. Hassan. Studying the dialogue between users and developers of free apps in the Google Play Store. *Empirical Software Engineering*, 23(3):1275–1312, 2018.

[12] S. M. Ibrahim, S. A. Salem, M. A. Ismail, and M. Eladawy. Novel sensitive object-oriented cohesion metric. In *Proceedings of the 22nd International Conference on Computer Theory and Applications (ICCTA)*, ICCTA '12, pages 154–159, 2012.

[13] A. K. McIntosh, S. Hassan, and A. Hindle. What can Android mobile app developers do about the energy consumption of machine learning? *Empirical Software Engineering*, 24(2):562–601, 2019.

[14] A. Z. Yang, S. Hassan, Y. Zou, and A. E. Hassan. An empirical study on release notes patterns of popular apps in the Google Play Store. *Empirical Software Engineering*, pages 1–41, 2021.

[15] G. Zhao, S. Hassan, Y. Zou, D. Truong, and T. Corbin. Predicting performance anomalies in software systems at run-time. *ACM Transactions on Software Engineering and Methodology*, 30(3):1–33, 2021.