Empirical Software Engineering (2020) 25:1533–1535
https://doi.org/10.1007/s10664-020-09811-0

EDITORIAL

Guest Editorial: Special Issue on Predictive Models and Data Analytics in Software Engineering

Ayse Tosun[1] · Shane McIntosh[2] · Leandro Minku[3] · Burak Turhan[4]

Published online: 19 February 2020

Applications of predictive modelling and data analytics in software engineering have been a long term and an established interest among researchers and practitioners. Such models and analyses can be targeted at: planning, design, implementation, testing, maintenance, quality assurance, evaluation, process improvement, management, decision making, and risk assessment in software and systems development. This interdisciplinary research between the software engineering and data mining communities also targets verifiable and repeatable experiments that are useful in practice.

This special section on Predictive Models and Data Analytics in Software Engineering presents extended versions of papers from the 14th International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE 2018). The conference was founded in 2004 as a workshop to bring researchers and practitioners to present, discuss and exchange ideas, results, expertise and experiences in construction and/or application of predictive models in software engineering, then extended its focus to include data analytics, after becoming an international conference in 2009.

This article belongs to the Topical Collection: *Predictive Models and Data Analytics in Software Engineering (PROMISE)*

✉ Ayse Tosun
tosunay@itu.edu.tr

Shane McIntosh
shane.mcintosh@mcgill.ca

Leandro Minku
l.l.minku@bham.ac.uk

Burak Turhan
burak.turhan@monash.edu

[1] Faculty of Computer and Informatics Engineering, Istanbul Technical University, 34467 Maslak, Istanbul, Turkey

[2] Department of Electrical and Computer Engineering, McGill University, Montréal, QC, Canada

[3] School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK

[4] Faculty of Information Technology, Monash University, 25 Exhibition Walk, Clayton, VIC 3800, Australia

Based on the ratings of program committee members and feedback from program chairs, we invited three papers from PROMISE 2018 conference for consideration for inclusion in this special section. Each paper was reviewed by at least three reviewers, following the rigour and quality standards expected from all papers submitted to Empirical Software Engineering Journal. After rounds of major revisions, all three papers were selected for inclusion in this special section:

"Deriving a Usage-Independent Software Quality Metric" by T. Dey and A. Mockus propose a new usage-independent quality measure which captures the usage patterns of thousands of software packages in terms of number of new users, usage intensity and frequency to identify their relationships with the number of post-release exceptions. The authors built Bayesian models to explain this relationship on NPM packages, and found that the exceptions are a result of new users, whereas the extent of usage does not appear to have a direct effect on the exceptions. An empirical comparison between the proposed quality measures and the state-of-the-art code quality measures also indicates that the usage patterns are significant predictors for post-release exceptions even after taking the code complexity metrics into consideration.

"Code Localization in Programming Screencasts" by M. Alahmadi et al. explores the applicability of Convolutional Neural Networks (CNNs) to the localization of code within screencasts. This localization task is an essential first step that must be performed before other operations (e.g., automated code extraction) can be reliably performed. The code localization CNNs are trained and evaluated using data extracted from 450 programming screencasts that cover topics using different programming languages (i.e., Java, C, and Python). The results indicate that the model is highly accurate, localizing the bounding box of a code snippet in the screencasts with an accuracy of 94%. Perhaps more impressively, the approach improves the accuracy of code generated by applying Optical Character Recognition (OCR) to screenshot frames from 31% to 97%.

"Cross-Version Defect Prediction: Use Historical Data, Cross-Project Data, or the Both?" by S. Amasaki challenges the long-lasting discussion of using other, i.e., cross, projects' data to train a defect predictor for a particular project (CPDP), but further extends this discussion by assessing the benefits of using cross-project data to build cross-version defect predictors (CVDP). In the paper, Amasaki empirically compares the models built with a single prior release of a project and with all prior releases of a project, against the models built with cross-projects' data. In total, 23 prior CPDP approaches were selected as baselines, and replicated with the addition of cross version training approaches. Their findings indicate using the latest prior release of the project is the best choice in average. If practitioners do not have CVDP data, the use of CPDP data to build CVDP model would also help.

All three papers in this special section show that the area of predictive models and data analytics in software engineering has been expanding to set a more thorough understanding of the issues surrounding existing topics through employing the advanced techniques of machine learning and deep learning.

We would like to extend our sincere thanks to the authors for their contributions, to reviewers for their invaluable assistance, and to the Editors-in-Chief for making this special section possible. We hope that you will enjoy reading these interesting contributions.

Ayse Tosun is an assistant professor at the Faculty of Computer andInformatics Engineering, Istanbul Technical University, Turkey. Prior to joining ITU, she worked as a post-doctoral research fellow at University of Oulu, Finland. She received her PhD in 2012, and MSc degree in 2008 from Department of Computer Engineering, Bogazici University, Turkey. Her research interests are empirical software engineering, more specifically mining software data repositories, software quality, measurement, process improvement, and applications of AI on building recommendation systems for software engineering.