Automated Software Engineering (2022) 29:62
https://doi.org/10.1007/s10515-022-00366-6

Guest Editorial: Deep Learning in Open-Source Software Ecosystems

Honghao Gao[1] · Zijian (Alex) Zhang[2] · Ramón J. Durán Barroso[3] · Xiong Luo[4]

Accepted: 26 September 2022 / Published online: 14 October 2022

The number of available open-source software projects has boosted the software engineering communities and platforms. Deep learning can effectively learn the intrinsic laws of data and mine the semantic and structural features of codes, such as CNN, RNN, and GAN. It brings new possibilities for code representation, users' behavior modeling and intelligent software development. This special issue accepted 6 papers for publication from open submissions. A summary of these accepted papers is outlined below.

In the paper entitled "Towards the Identification of Bug Entities and Relations in Bug Reports" by Bin Li et al., the authors define 8 types of relations between the bug entities and propose a neural joint model. They aim to effectively extract the bug entities and relations in the bug report. The proposed model contains two neural network modules: the first module RNN (Recurrent Neural Network) is used for bug entity recognition, while the second module RNN incorporates SDP (Shortest Dependency Path) of bug entity pairs to extract relations between bug entities. Through the joint learning, the relation information is utilized to improve the bug entity recognition. An empirical study is conducted on the open source project Mozilla and Eclipse, and

✉ Honghao Gao
gaohonghao@shu.edu.cn

Zijian (Alex) Zhang
swordstar1984@gmail.com

Ramón J. Durán Barroso
rduran@tel.uva.es

Xiong Luo
xluo@ustb.edu.cn

1 Shanghai University, Shanghai, China

2 University of Auckland, Auckland, New Zealand

3 Universidad de Valladolid, Valladolid, Spain

4 University of Science and Technology Beijing, Haidian, China

experiment results demonstrate that their method can effectively improve the recall rate, F1 value of named entity recognition and relation extraction tasks.

In the paper entitled "Improving Generality and Accuracy of Existing Public Development Project Selection Methods: A study on GitHub Ecosystem" by Can Cheng et al., the authors test the general PDPs (Public Development Projects) methods on the Standard Dataset, and find that these methods have a low F-measure and poor generality. They aim to understand the difficulties in selecting PDPs from large-scale open-source software ecosystem. To this point, the authors design two new methods called Enhanced_RFM and Fusion_DL_RFM, and conduct two experiments on the GHTorrent dataset. The experiment results show that Enhanced_RFM outperforms on the accuracy and stability, compared with machine learning-based methods.

In the paper entitled "Towards Automatic Detection and Prioritization of Pre-logging Overhead: A Case Study of Hadoop Ecosystem" by Chen Zhi et al., the authors focus on the problems of missing logging guards (MLG). They empirically validate the wide existence of MLG issues in the Hadoop ecosystem and reveal a high demand for automatic detection and prioritization of pre-logging statements(PLSs). The authors propose an accurate algorithm to detect PLSs that over 95% in precision and recall, and find out 16 problematic partially guarded logging calls that 10 of them are confirmed and fixed by developers. The authors also investigate two metric-based ranking approaches with six software metrics to prioritize PLSs. The results show that the execution frequency of PLSs achieves the best performance and the performance on multiple software metrics is improved 7.8% on average.

In the paper entitled "Code Comment Generation based on Graph Neural Network Enhanced Transformer Model for Code Understanding in Open-Source Software Ecosystems" by Li Kuang et al., the authors propose a Graph neural network enhanced Transformer model (GTrans) to generate code comment by learning code representation. Frist, the code graph is built from the parsed AST and augmented with control-flow and data-flow edges. Then, Transformer encoder is utilized to capture the global representation from the code sequence and a Graph Neural Network (GNN) encoder is employed to capture the local details in the code graph. Finally, they leverage a modified decoder to combine both global and local representations by attention mechanism in each layer to generate a code summary. Experiment results demonstrate that GTrans outperforms the state-of-the-art models up to 3.8% in METEOR metrics on the code comment generation, and outperforms the state-of-the-art models by margins of 5.8-9.4% in ROUGE metrics on the method name generation.

In the paper entitled "AppSPIN: Reconfiguration-based Responsiveness Testing and Diagnosing for Android Apps" by Zhanyao Lei et al., a tool named AppSPIN is proposed to diagnose App responsiveness bugs and explore configuration-sensitive bugs. The authors introduce a systematic performance-testing methodology, including the initial and reconfigured test runs, to automatically identify user-perceptible and configuration-sensitive responsiveness bugs. Experiments on 30 real-world Apps show that AppSPIN can detect 123 unique responsiveness bugs and successfully diagnose the cause for 87% cases with an average of 15-minute test time and negligible overhead.

In the paper entitled "HQLgen: Deep Learning Based HQL Query Generation from Program Context" by Ziyi Zhou et al., a novel model named HQLgen is proposed to automatically generate HQL queries from Java program context. HQL-gen is a data-driven approach that combines deep learning with template filling. In detail, it employs recurrent neural networks (RNN) to learn the contextual information of the program context and predicts the key elements within Hibernate Query Language(HQL) clauses via the attention mechanism. The authors use BOA to extract projects with HQL queries from the full mirror of GitHub by locating the createQuery method calls, and build a large dataset containing HQL queries with their related context. Experiment results show that the proposed approach achieves an accuracy of 34.52% in predicting the simple HQL queries.

The Guest Editors would like to express their deep gratitude to all the authors who have submitted their valuable contributions, and to the numerous and highly qualified anonymous reviewers. We think that the selected contributions, which represent the current state of the art in the field, will be of great interest to the community. We also would like to thank the Automated Software Engineering publication staff members for their continuous support and dedication. We particularly appreciate the relentless support and encouragement granted to us by Prof. Tim Menzies, the Editor-in-Chief of Automated Software Engineering.