Master thesis proposal

**Evaluation of machine learning methods   
in software defect prediction**

Łukasz Falas

November 29, 2024

# Background:

Software Quality Assurance (SQA) is an important element of the software development process focusing on detection of software defects (informally called software bugs) before releasing software or its new versions. Typically a software defect is defined as an incorrect step, process, or data definition in a computer program that prevents the program from working correctly. Presence of defects in software released into production environment can lead to dire consequences to end users and can result in greatly increased fixing costs in comparison to defects detected in prior stages of software development.

Due to this fact, software defect prediction has been a popular research topic for the last few decades. The main goal of this prediction is to identify defects in codebase before the software is released to the end users. The prediction process identifies potential defects in software and marks them for further analysis by the developers. This prediction process can be conducted at different abstraction levels as module, file, method, line and, as it becomes increasingly popular, in just-in-time manner in which changes introduced by developers (e.g. by commits to repository) are analyzed for potential defects as soon as they are introduced. Currently, state of the art methods focus on utilization of deep-learning models as an effective tool for software defect predictions.

# Thesis project:

The aim of the project is to evaluate the effectiveness of selected exiting machine learning methods for software defect prediction on a new *Defectors* dataset and potentially introduce modifications to this methods aimed at increasing their effectiveness in defect prediction for Python based software projects. Majority of current software defect prediction methods were developed on datasets utilizing projects written in Java and C, and while their authors often claim that these methods are programming language independent, this claims were not verified.

In the process of method implementation and evaluation, it is suggested that the focus should be put either on line-level or just-in-time (JIT) defect prediction. The final results of the project should indicate if current defect prediction methods can be utilized in Python projects and also propose potential method modifications if they are needed to increase their effectiveness.

The starting point for this project is the analysis of the proposed *Defectors* dataset, its structure and data that it supplies. This analysis should be followed by literature review aimed at selection of most promising machine-based defect prediction methods that can be utilized, their adaptation to data present in the dataset and potential modification aimed at their improvement. The next step should focus on the design of the experimental research and the evaluation of the utilized methods, followed by critical discussion regarding the experimental results.

# Data:

The project should utilize the *Defectors* dataset [1], which is a fairly new dataset for software defect prediction, which distinguished feature is its focus on software projects written in Python. The dataset can be found online. The link is supplied below:

<https://doi.org/10.5281/zenodo.7708984>

# References:

1. P. Mahbub, O. Shuvo, and M. Masudur Rahman, *“Defectors: A Large, Diverse Python Dataset for Defect Prediction,”* in 2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR), May 2023, pp. 393–397. doi: 10.1109/msr59073.2023.00085.
2. J. Pachouly, S. Ahirrao, K. Kotecha, G. Selvachandran, and A. Abraham, *“A systematic literature review on software defect prediction using artificial intelligence: Datasets, Data Validation Methods, Approaches, and Tools,”* Engineering Applications of Artificial Intelligence, vol. 111, p. 104773, 2022, doi: <https://doi.org/10.1016/j.engappai.2022.104773>.
3. C. Pornprasit and C. K. Tantithamthavorn, *“DeepLineDP: Towards a Deep Learning Approach for Line-Level Defect Prediction,”* IEEE Transactions on Software Engineering, vol. 49, no. 1, pp. 84–98, 2023, doi: 10.1109/TSE.2022.3144348.
4. T. Hoang, H. Khanh Dam, Y. Kamei, D. Lo, and N. Ubayashi, *“DeepJIT: An End-to-End Deep Learning Framework for Just-in-Time Defect Prediction,”* in 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), 2019, pp. 34–45. doi: 10.1109/MSR.2019.00016.
5. T. Hoang, H. J. Kang, D. Lo, and J. Lawall, *“CC2Vec: distributed representations of code changes,”* in Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering, 2020, pp. 518–529. doi: 10.1145/3377811.3380361.