Machine-Learning for Optimal Fitness Function Selection in Automated Testing

Author: Daniela Toader
Supervisor(s): Annibale Panichella, Pouria Derakhshanfar, Mitchell Olsthoorn

1. INTRODUCTION

Testing
- essential step in assuring the quality of software
- goal is finding potentially dangerous faults in code

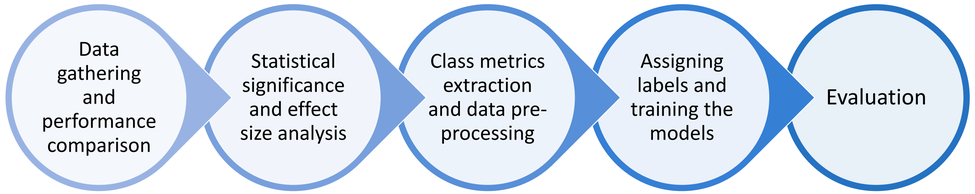
EvoSuite [1]
- SOTA tool that generates unit-level test suites
- uses a genetic algorithm, which optimises for multiple **criteria** (objectives) simultaneously [2]
- ran with criteria **default, branch** (coverage) and branch coverage & weak mutation **(bcwm)**

Research gap
- lack of information that relates objectives to class properties and in turn to coverage and fault-finding capabilities

2. RESEARCH QUESTIONS

- To what extent does **bcwm** affect structural coverage and fault detection capabilities for different search budgets?
- What is the relationship between class static code metrics and the structural coverage and fault detection capabilities when using **bcwm**?

3. METHODOLOGY



4. RESULTS

Figure 1. Decision Tree path with **bcwm** label (branch coverage - 60 seconds)

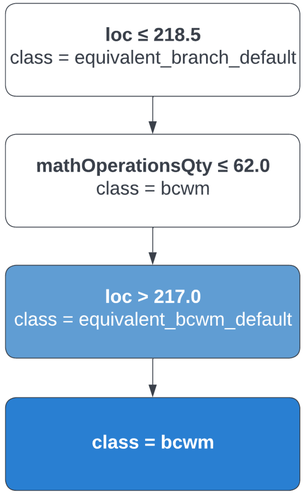
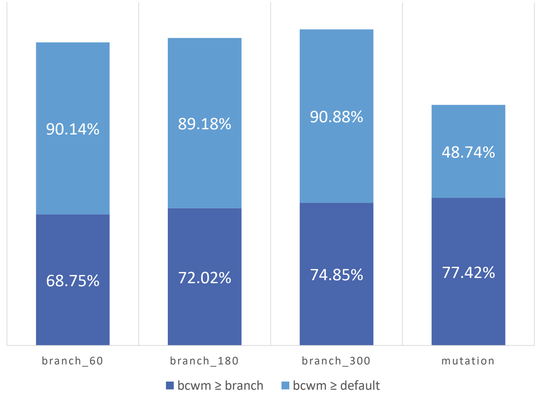


Table 1. F1 Scores of the models

	DT	SVC	RF	LR
branch_60	0.89	0.76	0.93	0.57
branch_180	0.91	0.74	0.95	0.57
branch_300	0.94	0.71	0.97	0.49
mutation	0.90	0.71	0.93	0.63

Table 2. Performance comparison of **bcwm** vs **branch** and vs **default**



5. CONCLUSION

Performance
- **bcwm ≤ branch** in 86.5% (branch coverage)
- **bcwm ≥ default** in 90.0% (branch coverage)
- **bcwm ≥ branch** in 77.4% (mutation score)
- **bcwm ≤ default** in 84.2% (mutation score)

Models
- Random Forest performs best, Decision Tree is second
- Logistic Regression has lowest performance, Support Vector Classifier is second to last
- Most frequent features – loc, wmc, mathOperationsQty, cbo, fanout, rfc

6. LIMITATIONS & FUTURE WORK

- **Computational power –** more extensive Grid Search for hyperparameter tuning
- **Time frame –** more search budgets, optimisation criteria, and models, more data balancing, feature selection and extraction techniques

7. REFERENCES

[1] Gordon Fraser and Andrea Arcuri. Evosuite: automatic test suite generation for object-oriented software. In Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering, pages 416–419, 2011

[2] Annibale Panichella, Fitsum Meshesha Kifetew, and Paolo Tonella. Automated test case generation as a many-objective optimisation problem with dynamic selection of the targets. IEEE Transactions on Software Engineering, 44(2):122–158, 2017