Chapter 7

Discussion

In this chapter, we elaborate on an inside discussion or summary of the thesis based on proposed SBP models. After that, we provide the response to common RQs introduced in thesis objective Section 1.6.

This thesis has undertaken an extensive exploration of SBP, presenting four distinct approaches while also delving into the multifaceted challenges inherent in this domain. We conducted an experimental analysis of these challenges, revisiting existing solutions and critically assessing their efficacy. Notably, we introduced a classification-based supervised SBP model and subsequently expanded upon it by venturing into classification-based unsupervised SBP. This progression extended into regression-based unsupervised SBP. Furthermore, our work encompassed the extension of classification-based unsupervised SBP to functional programming datasets. Below, we provide a concise summary of all four research endeavors.

WMV: Classification-based supervised SBP- WMV is a classification-based supervised software bug prediction (SBP) model, and its development incorporated a unique weight-evaluating scheme in conjunction with SMV to establish the innovative WMV ensemble technique. This technique harnessed the power of heterogeneous ensemble learning, leveraging five distinct base classifiers as weak learners

and employing a weighted majority voting mechanism in the ensemble learning. Our empirical study revealed that ensemble learning played a pivotal role in mitigating class imbalance and over-fitting issues, resulting in the creation of more versatile and resilient SBP models. Our rigorous validation process encompassed 28 public datasets drawn from five different groups. Our findings underscored WMV's exceptional performance, surpassing the majority of existing state-of-the-art methods across various metrics such as accuracy, FM, and MCC. Furthermore, we conducted a comprehensive k-fold cross-validation test to assess the model's stability, and the results attested to WMV's high degree of stability and reliability.

TCL/TCLP: Classification-based unsupervised SBP - The proposed TCL/TCLP method involves two major steps: TCL for automatic labeling based on metrics thresholds and TCLP for metrics and instance selection, machine learning training, and prediction. Different combinations of metrics and instance selections are analyzed to optimize the approach. TCLP aims to minimize the number of selected metrics and instances through heuristics. It demonstrates superior performance when selecting a minimum number of metrics based on MVR and a minimum percentage of instances with IVR, ensuring the inclusion of at least two buggy instances. Additional instances are selected progressively if fewer buggy instances are initially identified. This approach is used to reduce complexity and identify essential metrics. The study shows that the proposed TCLP labels more data points correctly compared to other methods, exhibiting its effectiveness in SBP across various datasets. Comparing automated labeling with expert labeling, the automated approach based on metrics threshold is faster and offers respectable accuracy, making it a valuable alternative. The study highlights the significance of selecting metrics based on MVR and a varying percentage of instances based on IVR. The results suggest that selecting a minimum number of metrics (2 to 5) with minimum MVR and a minimum percentage of instances (85% to 95%) results in better performance without significantly increasing model complexity.

MTB/MTBP: Regression-based unsupervised SBP - The discussion highlights the promising applications and performance of the proposed MTB/MTBP SBCV prediction model based on software metric threshold calculation. The model demonstrates its effectiveness in indexing software modules with predicted bug counts, particularly for modules with a maximum of seven bugs. Additionally, MTBP is found to be more effective for software projects with bug percentages below 48%, while MTB outperforms it when bug percentages exceed this threshold, indicating potential use cases in different contexts. Furthermore, the model exhibits flexibility by being applicable to software projects of various sizes without affecting its performance. However, a limitation of MTB/MTBP is its SBP capability, constrained by the maximum number of software metrics selected for prediction. It's essential to acknowledge the assumptions underlying MTB/MTBP's applicability, particularly the alignment between software metrics and the bug proneness concept. These discussions open avenues for future research, exploring specific scenarios and further refining the model's capabilities.

UMV: Classification-based unsupervised SBP for FP - In this discussion, it's evident that the proposed UMV approach represents a pioneering step in the realm of SBP in functional programming, particularly within the Haskell domain. This novel approach leverages six software metrics (SMs) to identify bugs in functions based on metrics extracted from Haskell packages. It's worth noting that IND is identified as an unsuitable SM for SDP models, as illustrated in the bug-wise boxplot analysis. Furthermore, the transformation techniques of sqrt and cbrt have shown promise in enhancing SDP. UMV, a generalized and robust SDP model, demonstrates notable improvements over individual threshold-based models, positioning it as the best model among both threshold-based and unsupervised ML approaches. The silhouette analysis suggests UMV excels at creating well-defined clusters for binary classification, with its performance surpassing that of threshold-based and unsupervised ML models. However, it's essential to explore more advanced ensemble models that can significantly enhance the performance of threshold-based

techniques. While these experiments were conducted on a limited dataset of six SMs and four projects, further investigation is needed, especially for datasets like Etherium. Additionally, the comparison with the ACL method highlights the importance of effective threshold values and the need to evaluate threshold-based SDP methods concerning dataset-specific statistical characteristics.

Response to common RQs introduced in Thesis Objectives (Section 1.6)

1. RQ-1: Is the prediction performance of the proposed supervised approach comparable to that of standard supervised ML models?
   Response of RQ-1: YES, the prediction performance of the proposed supervised approach WMV is significantly comparable to that of standard supervised ML models?

2. RQ-2: Is the performance of unsupervised approaches comparable to the performance obtained using the standard supervised / unsupervised learning approaches?
   Response of RQ-2: YES, the performance of unsupervised approaches TCL/TCLP, MTB/MTBP, and UMV is comparable to the performance obtained using the standard supervised / unsupervised learning approaches

3. RQ-3: Are the results obtained by proposed techniques comparable to those obtained by best-performing SOTA advanced techniques?
   Response of RQ-3: YES, the results obtained by proposed techniques WMV, TCL/TCLP, MTB/MTBP, and UMV are comparable to those obtained by best-performing SOTA advanced techniques.

4. RQ-4: Is the performance of proposed approaches without sampling techniques comparable to that of proposed approaches with sampling techniques?
   Response of RQ-4: YES, the performance of proposed approaches without

sampling techniques is comparable to that of proposed approaches with sampling techniques.

5. RQ-5: Is the prediction performance of the proposed regression-based SBP comparable to that of standard supervised regression ML models?
   Response of RQ-5: YES, the prediction performance of the proposed regression-based SBP MTB/MTBP is comparable to that of standard supervised regression ML models.

6. RQ-6: Are the extracted SMs capable of predicting defect proneness in the functions of Haskell?
   Response of RQ-6: YES, the extracted SMs are capable of predicting defect proneness in the functions of Haskell.