

Vaibhav vaibhav

vaibhav_thesis1.0

 M.TECH. THESIS 2

Document Details

Submission ID**trn:oid:::27535:140930402****Submission Date****May 29, 2026, 1:21 PM GMT+5:30****Download Date****May 29, 2026, 3:51 PM GMT+5:30****File Name****vaibhav_thesis1.0.pdf****File Size****282.2 KB****27 Pages****8,195 Words****46,986 Characters**

9% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Small Matches (less than 8 words)

Exclusions

- 14 Excluded Matches

Match Groups

-  **65 Not Cited or Quoted 9%**
Matches with neither in-text citation nor quotation marks
-  **6 Missing Quotations 1%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 7%  Internet sources
- 6%  Publications
- 6%  Submitted works (Student Papers)

Match Groups

- 65 Not Cited or Quoted** 9%
Matches with neither in-text citation nor quotation marks
- 6 Missing Quotations** 1%
Matches that are still very similar to source material
- 0 Missing Citation** 0%
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted** 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 7% Internet sources
- 6% Publications
- 6% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

- 1** Internet
mdpi-res.com <1%
- 2** Publication
Gabriel Kofi Armah, Guangchun Luo, Ke Qin. "Multi_level data pre_processing for ... <1%
- 3** Internet
arxiv.org <1%
- 4** Internet
dokumen.pub <1%
- 5** Publication
Zhou Xu. "Machine-Learning-Assisted Software Defect Prediction", Springer Scien... <1%
- 6** Publication
Cenk Temizel, Salih Tutun, Celal Hakan Canbaz, Ekrem Alagoz et al. "Artificial Inte... <1%
- 7** Internet
www.karunya.edu.in <1%
- 8** Student papers
Liverpool John Moores University on 2025-03-14 <1%
- 9** Student papers
Tilburg University on 2026-05-19 <1%
- 10** Internet
ojs.istp-press.com <1%

11	Internet	psasir.upm.edu.my	<1%
12	Student papers	University of South Wales - Pontypridd and Cardiff on 2025-06-19	<1%
13	Publication	Ruchika Malhotra, Anjali Bansal, Marouane Kessentini. "Chapter 36 A Hybrid Met...	<1%
14	Internet	andaloussi.org	<1%
15	Internet	thesai.org	<1%
16	Internet	tmfv.com.ua	<1%
17	Internet	cdn.iiit.ac.in	<1%
18	Internet	elibrary.tucl.edu.np	<1%
19	Internet	www.mdpi.com	<1%
20	Internet	www.ijritcc.org	<1%
21	Student papers	Delhi Technological University on 2026-05-24	<1%
22	Internet	dirp4.pids.gov.ph	<1%
23	Internet	inass.org	<1%
24	Internet	jisem-journal.com	<1%

25	Internet	jssci.org	<1%
26	Internet	jst-ud.vn	<1%
27	Internet	lhkhiem28.github.io	<1%
28	Internet	pdfs.semanticscholar.org	<1%
29	Internet	ray.yorks.ac.uk	<1%
30	Publication	"Advances in Smart Knowledge Computing", Springer Science and Business Medi...	<1%
31	Student papers	Australian National University on 2023-05-30	<1%
32	Publication	Debasish Pradhan, Debendra Muduli. "Improved prediction of software defect: A ...	<1%
33	Publication	Fernanda C. Orlandi, Julio C. S. Dos Anjos, Juan F. de P. Santana, Valderi R. Q. Leith...	<1%
34	Student papers	Liverpool John Moores University on 2023-11-10	<1%
35	Student papers	University of Melbourne on 2026-05-22	<1%
36	Student papers	University of Wolverhampton on 2026-05-21	<1%
37	Publication	Yousef Abdi, Saeed Parsa, Yousef Seyfari. "A hybrid one-class rule learning appro...	<1%
38	Publication	Zhihao Lin, Wei Ma, Tao Lin, Yaowen Zheng, Jingquan Ge, Jun Wang, Jacques Klein...	<1%

39	Internet	ira.lib.polyu.edu.hk	<1%
40	Internet	pure.rug.nl	<1%
41	Internet	real-j.mtak.hu	<1%
42	Internet	www.cbit.ac.in	<1%
43	Publication	"Proceedings of the Third Congress on Control, Robotics, and Mechatronics", Spri...	<1%
44	Publication	Ahmed A. Abd El-Latif, Mohammed A ElAffendi, Mohamed Ali AlShara, Yassine Ma...	<1%
45	Student papers	GLA University on 2016-05-11	<1%
46	Publication	Oshina Vasishth, Ankita Bansal. "Enhanced software defect prediction using krill ...	<1%
47	Publication	R. Anandan, Souvik Pal, D. Balaganesh, Farshad Badie. "Federated Learning for H...	<1%
48	Publication	Raghunath Dey, Jayashree Piri, Biswaranjan Acharya, Pragyan Paramita Das, Vass...	<1%
49	Student papers	The Robert Gordon University on 2024-11-02	<1%
50	Student papers	University of Edinburgh on 2025-01-23	<1%
51	Student papers	University of Warwick on 2023-09-27	<1%
52	Publication	Yue Jiang, Bojan Cukic, Yan Ma. "Techniques for evaluating fault prediction model...	<1%

53	Internet	dirros.openscience.si	<1%
54	Internet	public-pages-files-2025.frontiersin.org	<1%
55	Internet	publications.polymtl.ca	<1%
56	Internet	www.e-informatyka.pl	<1%
57	Internet	www.internationaljournalssrg.org	<1%
58	Internet	www.preprints.org	<1%

Abstract

The software defect prediction is an important issue in software quality as it helps to detect the defective software modules at early stages of software development. Correct prediction models decrease the cost of maintenance, testing and total project risk. Conventional software defect prediction methods, however, tend to be centralized and use standard machine learning methods, which could have issues with class imbalance, data privacy, heterogeneous data distribution, and lack of model generalization. These limitations have led to a need for advanced and scalable prediction frameworks that can perform effectively in real-world environments.

This thesis introduces a thorough research to apply federated learning and hybrid optimization methods to software defect prediction. The main emphasis of the work is on the three research contributions. The first contribution involves developing a FedProx-based federated learning framework with the aim of enhancing the model's generalization performance in a decentralized setup, whilst simultaneously managing Non-IID data distributions amongst the clients. The second contribution presents a MOON-based model-contrastive federated learning framework to further reinforce the model consistency at local sites while boosting the convergence performance in distributed scenarios. The fourth one introduces an integrated ensemble learning-PSO-GWO optimization framework for better feature selection and hyperparameter tuning and enhance the classification accuracy for defect prediction.

Overall, this thesis highlights the potential of combining federated learning, meta-heuristic optimization, and ensemble learning to build efficient and scalable software defect prediction systems. The findings of this work can contribute toward the development of intelligent and privacy-aware software engineering solutions capable of handling complex and distributed software development environments.

Contents

Acknowledgement	ii
Candidate's Declaration	iii
Certificate	iv
Abstract	iv
Content	vi
List of Tables	vii
List of Figures	viii
List of Symbols and Abbreviations	ix
1 INTRODUCTION	10
1.1 Background of Software Defect Prediction	10
1.2 Challenges in Traditional Approaches	10
1.3 Motivation for Federated and Hybrid Approaches	11
1.4 Objectives of the thesis	11
1.5 Organization of the thesis	12
2 LITERATURE REVIEW	13
2.1 Traditional Software Defect Prediction	13
2.2 Machine Learning & Ensemble Approaches	13
2.3 Federated Learning in SDP	14
2.4 Optimization Techniques (GWO, PSO, GA)	14
3 METHODOLOGY	15
3.1 Overall framework	15
3.2 Federated learning approaches	15
3.2.1 FedProx framework	16
3.2.2 MOON-Based Federated Learning	16
3.2.3 Hybrid Optimization Approach (GWO-PSO-GA)	16
3.2.4 Ensemble Learning Strategy	17
3.2.5 Dataset Description	19
3.2.6 Evaluation Metrics	19
4 RESULTS and DISCUSSION	20
4.1 Results of FedProx model	20

4.2	Results of MOON-Fed Model	20
4.3	Results of Hybrid GWO–PSO–GA Model	21
4.3.1	Impact of Combining GWO, PSO, GA, and Stacking Ensemble . .	22
4.3.2	Statistical Significance Analysis of Model Performance	23
4.3.3	Friedman Test and Bonferroni Post-hoc Analysis	24
4.4	Comparative Analysis	25

5	CONCLUSION AND FUTURE SCOPE	26
5.1	Conclusion	26
5.2	Limitations	26
5.3	Future Works	27

A	Appendix Title	28
----------	-----------------------	-----------

List of Tables

3.1	Dataset Description	19
4.1	Performance of FedProx based FL approach	20
4.2	MOON-based FL Approach Metrics	21
4.3	Hybrid Approach Performance Across Multiple Datasets	21
4.4	Accuracies of combination of GWO, PSO and GA	22
4.5	Precision of combination of GWO, PSO and GA	23



List of Figures

3.1	Federated Learning Architecture	16
3.2	MOON-FED algorithm Architecture	17
3.3	Architecture of hybrid approach	18
4.1	Proposed Approach Performance Metrics	22
4.2	Boxplot analysis of accuracy across model combinations	24
4.3	Boxplot analysis of F1-score across model combinations	24

LIST OF SYMBOLS AND ABBREVIATIONS

SDP	Software Defect Prediction
FL	Federated Learning
FedProx	Federated Proximal Optimization
MOON	Model-Contrastive Federated Learning
GWO	Grey Wolf Optimizer
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
RF	Random Forest
SVM	Support Vector Machine
GB	Gradient Boosting
NB	Naive Bayes
KNN	k-Nearest Neighbors
AUC	Area Under the Curve
IID	Independent and Identically Distributed
Non-IID	Non-Independent and Non-Identically Distributed
SMOTE	Synthetic Minority Oversampling Technique
ADASYN	Adaptive Synthetic Sampling
NASA	National Aeronautics and Space Administration
n	Number of datasets used in statistical testing
k	Number of models compared in statistical testing
R_j	Sum of ranks for the j th model
χ_F^2	Friedman test statistic
df	Degrees of freedom
p	Probability value used for significance testing
α	Significance level used for hypothesis testing
Z	Standardized statistic for Bonferroni post-hoc comparison
H_0	Null hypothesis
H_1	Alternative hypothesis

CHAPTER 1

INTRODUCTION

Software defect prediction focuses on identifying the software modules which are likely to have the defects, helps in optimizing the allocation of testing resources, improves software quality and reduces the testing cost [1, 2]. SDP helps to find the defective modules to minimize the testing costs in a limited budget. A software defect is a deficiency in a software which makes it behave in unexpected ways. The main goal of SDP is to identify these defects in early stages of testing. Due to huge and more sophisticated growth in the software industry, anticipating the reliability of software is an important task in software development lifecycle. Since, practically testing all the software modules for the defects is not possible, there is a need to detect the faulty or defect prone modules from the huge software systems. To anticipate this task there are different techniques to solve software defects problems in early testing. This study includes the use of meta-heuristic optimisation techniques and ensemble learning method for feature selection from the datasets by choosing the most influential features that contribute in predicting or detecting the defective or non-defective software module.

1.1 Background of Software Defect Prediction

SDP is an essential research topic in software engineering that seeks to predict defects in software modules prior to deployment. Defect prediction at an early stage allows proactive quality assurance, lower maintenance costs, and higher software reliability. Existing SDP approaches utilize ML models that have been trained on past defect datasets to predict software components as defective or not [3, 4]. These methods generally use the centralized collection of data due to which issues like ownership, privacy, and security of data arise. Another problem is an imbalanced dataset in which the number of the samples which are defective are very less in comparison to the non-defective ones causing the predictions to be skewed.

Defect prediction is one of the key challenges and the problem is to identify the defective modules with high accuracy. SDP is a complicated task due to various challenges related to software datasets. One of the main problems is that it has a very high-dimensional feature space where software metrics usually encompass many attributes that are either irrelevant or redundant with respect to the output target. This results in increased complexity, as well as overfitting and reduced generalization capability. Imbalanced datasets are also challenging, with defective modules being less in number than the non-defective ones. This misconception leads to a situation where the model learns more about the non-defective classes and learns quite less for defect classes.

1.2 Challenges in Traditional Approaches

However, many existing software defect prediction methods have been found to have substantial limitations for real-world situations. The traditional models are based on

centralization of data collection, posing privacy, security and ownership concerns, particularly when data is stored and accessible in several organizations. In addition, these approaches are generally restricted to the assumption of independent and identically distributed (IID) data, which is often not realistic in the real world, causing issues with generalization to different data sets. Another challenge with traditional machine learning models is the problem of class imbalance, noisy data, and irrelevant or redundant features, which can negatively impact the accuracy of the predictions. Moreover, many classical methods lack tools such as feature selection and hyperparameter optimization, which leads to suboptimal performance of the model.

1.3 Motivation for Federated and Hybrid Approaches

A number of studies stated in this section have employed FL to SDP, confirming its potential in improving defect prediction performance without disclosing data confidentiality. For example, differential perception with inheritance FL has been suggested to effectively capture cross-project variations [5], whereas Docker-based FL frameworks have enabled effective model deployment and monitoring [6]. Some studies have utilized federated meta-learning [7], oversampling techniques [8], and privacy-preserving cross-project learning [9] to enhance model performance in decentralized settings. Even after these developments, FL presents new challenges like client drift, large communication overhead, and particularly non-IID data distributions, which compromise model convergence and stability [10, 11].

The model of contrastive learning which was introduced in MOON (Model Contrastive Federated Learning) states a different solution for this challenge. MOON reflects the local model parameters to the global model parameters under a loss function which is contrastive in nature [12]. In the case of SDP, the representations of the features can be exploited to improve the generalizability across projects and datasets which is a prominent property reflected in the benchmark defect datasets of NASA like CM1, KC1, and PC1.

Optimization and ensemble learning techniques have proven to be an effective means to improve the performance of software defect prediction models. Many traditional models have suboptimal feature selection and poorly-tuned hyperparameters, affecting prediction accuracy directly. To identify the most relevant features and optimize the parameters of the models, metaheuristic optimization algorithms like Grey Wolf Optimizer (GWO), Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) are employed [13, 14, 15, 16].

1.4 Objectives of the thesis

The main goal of this thesis is to create a useful and dependable model for software defect prediction using state-of-the-art machine learning methodologies. One of the main focuses is on the development of federated learning methods to tackle centralized data processing challenges, especially in cases where privacy and data distribution are significant issues. In this context, data heterogeneity and Non-IID distributions are explored by means of approaches like FedProx and MOON, as a way to enhance the generalization of the models on different datasets.

This work also focuses on using federated learning and hybrid optimization and ensemble learning to improve prediction performance. Metaheuristic optimization techniques such as GWO, PSO and GA are designed to enhance the selection of features and the tun-

ing of the hyperparameters; ensemble methods are used to get the advantages of several models to achieve more accurate and robust results. In general, the thesis aims to give a comparative study of these approaches and show the advantages of using their combination for constructing more efficient and scalable software defect prediction systems.

1.5 Organization of the thesis

This thesis is structured to make it easy to understand the problem, the methods employed and the results obtained. The book opens with a chapter that provides an overview of the software defect prediction concept, identifies major problems with current software defect prediction methods, and outlines the motivation for using novel techniques like federated learning and hybrid optimization. It also explains the main goals of the research and provides the background for the research presented in the subsequent chapters.

The second chapter explores the existing literature on software defect prediction, including traditional machine learning approaches, ensemble and recent advances in federated learning. The third chapter describes the proposed methodology, including the implementation of FedProx and MOON-based federated learning frameworks, as well as the hybrid GWO-PSO-GA optimization approach combined with ensemble learning. The fourth chapter presents the experimental results along with a detailed discussion and comparison of the different models. Finally, the fifth chapter concludes the thesis by summarizing the key findings, discussing limitations, and suggesting possible directions for future research.

CHAPTER 2

LITERATURE REVIEW

 This section of the thesis includes the study of different research that has been carried out in the field of software defect prediction. It includes all the outcomes and findings in respect to traditional approaches to SDP, federated learning approaches and hybrid algorithms approach towards SDP.

2.1 Traditional Software Defect Prediction

The NB classifier works well for small data sets due to its predictive nature, while the SVM uses a margin-based classification strategy, making it very effective for high dimensional data. Decision trees, on the other hand, are fast and interpretable but more prone to overfitting. KNN is useful for non-linear relationships but has a high computational complexity. All these conclusions depict that standard machine learning models are capable enough to show good accuracy, but they really depend a lot on the dataset features and can even be noisy or redundant. Thus, they adapt as evident in the need for feature selection and hyperparameter tuning, which can be done effectively with metaheuristic optimization methods. Predictive capability of SDP models can be enhanced enormously using hybrid metaheuristic methods having the promise of better defect classification with reduced computational. By using the historical data these methods help in the prediction whether the given software module contains defects or not. These models rely on manually designed features obtained after deriving them from source code, process measures, and history of defects. They work well for linear and rule-based relations among the data. However, in general, they fail when it comes to SDP-related high dimensional datasets with an effect of feature redundancy and complicated dependencies among software measures. In general, conventional ML models have a major downside when it comes to dealing with highly imbalanced datasets.

2.2 Machine Learning & Ensemble Approaches

Ensemble learning methods are known to be very effective for SDP, aggregating different classifiers to improve accuracy in prediction. RF, SVM, and gradient boosting exist as the various models of ensemble learning widely applied, that possess better accuracy than individual classifiers [17, 18, 19]. The RF is a blend of many DTs, using the averaging of the prediction from multiple trees to increase stability and decrease overfitting. This greatly helps defect detection by applying sequential model construction, thereby correcting the errors made by the previous step: applying model boosting. The SVM ensembles use many support vectors to maximize the classification boundary for defect-prone modules. In the existing study it has been observed that ensemble methods perform better than individual models by reducing bias and variance and improving generalization; for instance, RF achieved an AUC of 0.929 on the PC1 dataset, outperforming individual models like NB and DT, while an improved RF model with hyperparameter tuning improved accuracy

by 3.9 percent. These results justify the use of ensemble methods for increasing accuracy in defect prediction, solving the class imbalance problem, and increasing the robustness of models, making ensemble methods a strong candidate for application in software defect prediction.

2.3 Federated Learning in SDP

Data privacy and heterogeneity have been principal barriers in defect prediction among organizations. Researchers have therefore initiated the study of FL structures that would enable multiple clients (e.g., software teams or companies) to collaborate on training without sharing raw data. FedAvg and FedProx are a few techniques that have reported promising results by evading issues due to data silos and partial heterogeneity [10, 11]. FedProx, in particular, improved stability of the worldwide model via a proximal term that regularized the local update so as to control client drift in non-identical distributions [10]. Despite the advances, the vast majority of available federated techniques in SDP are still plagued with learning robust and invariant representations, especially where local data significantly differ in structure and defect characteristics.

The recent developments in contrastive learning offer new possibilities for improving model generalization in federated environments. Contrastive learning helps models learn how to differentiate similar and dissimilar feature representations, which leads to more discriminative embeddings even in a federated environment. One extension of FL is known as Model-Contrastive Federated Learning (MOON), where contrastive loss is used in the FL framework [12]. MOON is different from FedProx in that it is based on optimization stability rather than local and global representation by penalizing the difference between the local model and global model or reference model while pushing away old global representations. This helps the model learn invariant features from clients with different data distributions.

2.4 Optimization Techniques (GWO, PSO, GA)

The hybridization metaheuristic methods have gained much attention in the area of SDP, due to their capability of performing better in feature selection and classification accuracy through the combination of the strengths of several optimization methods. A Hybrid GWO and PSO (HGWOPSO) method was proposed for better SDP performance [20]. The work combined HGWOPSO with Gradient Boosting classifiers and reported better AUC values than any of the single optimization algorithms considered. The findings, therefore, reaffirm that hybrid metaheuristic-based feature selection enhances defect prediction models' accuracy and computational efficiency. Related optimization studies also use binary GWO, PSO-based bagging, GA-based feature selection, and neural models optimized using GA and PSO for SDP [21, 22, 23, 24]. Another study investigated hybrid GA and PSO approaches combined with Neural Networks for SDP. The results suggested that GA-based feature selection and PSO hyperparameter optimization noticeably boosted classifier performance especially when working with high-dimensional and imbalanced datasets. The hybrid approach thus improved not only predictive accuracy, but also robustness and generalizability of SDP models. Such studies accentuate the gains from using multiple metaheuristic-based hybrid algorithms, thus providing the justification for using a GWO-PSO-GA hybrid in this study.

CHAPTER 3

METHODOLOGY

The approach to SDP in order to get more robust findings and results were made in the direction of federated learning and hybridisation of different metaheuristics algorithms. Different approaches and methods were tried and the best result giving methods were taken into account to solve the challenges of SDP.

3.1 Overall framework

The proposed framework which uses the part of federated learning includes the decentralized learning strategy for SDP with a privacy-preserving base in FL. Multiple clients, that is, corresponding to different software projects jointly train global models collectively, without sharing raw data. Each client learns the global model parameters and local trains on its defect-tagged dataset, then sends back only a new model weight to the central server. Afterwards the server combines these updates via a weighted average approach and iteratively sends the new global model back to the clients.

In the MOON-Fed approach we make efforts to address data privacy, non-IID heterogeneity, and insufficient generalization in SDP within a decentralized FL paradigm assisted by the Model-Contrastive (MOON) algorithm. The framework developed under that name, referred to as MOON SDP, employs a contrastive learning process at the process level of a model so that local clients benefit from a stable, consistent representation without model drift across non-identical datasets. By applying contrastive regularization at the process level of federated optimization, the framework allows clients to learn discriminative and globally consistent representations to conduct SDP.

In the hybridisation approach we introduces the hybrid metaheuristic modeling optimization method combining GWO, PSO, and GA methods for selecting the features and tuning the hyperparameters in SDP. Selection of right features is important to the research of SDP because high-dimensional datasets contain redundant or irrelevant features that diminish classification accuracy. The hybrid model takes advantage of the GWO good exploration characteristic, PSO good convergence speed, and genetic diversity of GA to create the most appropriate feature subset. The fitness function assesses feature subsets that confirm only the most important features into the end model. It also involves the outlier elimination technique of isolation forest and class balancing technique of ADASYN (Adaptive Synthetic Sampling) to enhance predictive performance.

3.2 Federated learning approaches

FL based decentralized approaches helps in SDP to a very great extent. The proposed scheme establishes the setting for the joint model training over several clients without exposing sensitive information regarding software repositories.

3.2.1 FedProx framework

The proposed framework features a decentralized learning strategy for SDP with a privacy-preserving base in FL. Multiple clients, that is, corresponding to different software projects jointly train global models collectively, without sharing raw data. Each client learns the global model parameters and local trains on its defect-tagged dataset, then sends back only a new model weight to the central server. Afterwards the server combines these updates via a weighted average approach and iteratively sends the new global model back to the clients. To enhance generalization and stabilize local training across non-identically distributed (non-IID) client datasets, the model includes the application of the FedProx optimization method used in the local training phase. The built-in preprocessing functions at the client level in the learning pipeline are ADASYN-based class balancing, polynomial feature augmentation, and normalization scaling. The federated training process propagates over a fixed number of communication rounds to produce a strong privacy-protective global model that learns the defect-prone patterns in heterogeneous software repositories.

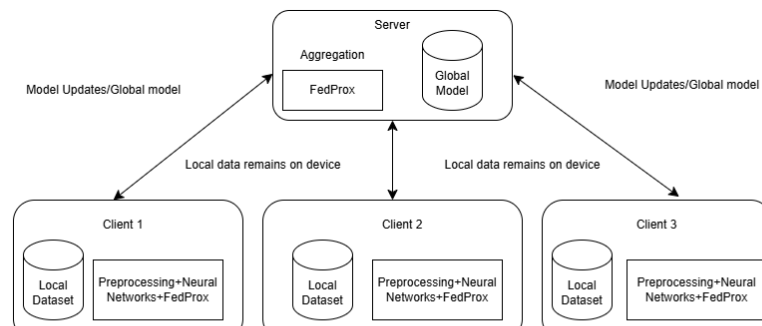


Figure 3.1. Federated Learning Architecture

3.2.2 MOON-Based Federated Learning

The framework developed under that name, referred to as MOON SDP, employs a contrastive learning process at the process level of a model so that local clients benefit from a stable, consistent representation without model drift across non-identical datasets. By applying contrastive regularization at the process level of federated optimization, the framework allows clients to learn discriminative and globally consistent representations to conduct SDP.

Among the key reasons behind this federated scheme is the latter's native ability to maintain confidentiality of information. In this arrangement, no bare defect raw information is permanently held on any central server, with just learned model parameters or gradients communicated between a client device and a central server. In this way, privacy requirements are met in shared environment configurations, e.g., inter company software developmental partnerships, where information exchange is otherwise not possible.

3.2.3 Hybrid Optimization Approach (GWO-PSO-GA)

The study introduces the hybrid metaheuristic modeling optimization method combining GWO, PSO, and GA methods for selecting the features and tuning the hyperparameters in SDP. Selection of right features is important to the research of SDP because high-dimensional datasets contain redundant or irrelevant features that diminish classification accuracy. The hybrid model takes advantage of the GWO good exploration characteristic,

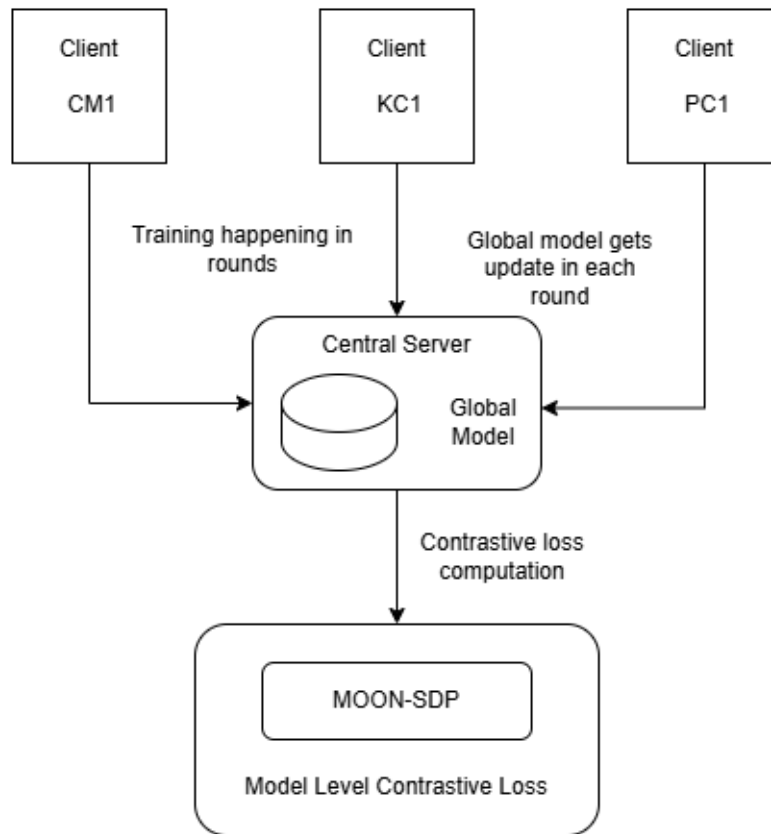


Figure 3.2. MOON-FED algorithm Architecture

PSO good convergence speed, and genetic diversity of GA to create the most appropriate feature subset. The fitness function assesses feature subsets that confirm only the most important features into the end model. It also involves the outlier elimination technique of isolation forest and class balancing technique of ADASYN (Adaptive Synthetic Sampling) to enhance predictive performance.

For enhancing accuracy in defect prediction, the final subset of features is input to an ensemble learning model comprising RF, SVM and gradient boosting. RF provides robustness and does not overfit; SVM enables one to enhance boundary-based classification; and Gradient Boosting makes incremental adjustments for error. Hybrid GWO-PSO-GA optimization presents an ensemble learning approach, and it produces a generalized high-performance SDP model sufficient to deal with challenges such as high dimensionality, imbalanced datasets, and noisy features and make it suitable to deal with real-world SDP issues.

3.2.4 Ensemble Learning Strategy

In this work, ensemble learning was used to enhance software defect prediction software model performance and stability. The ensemble approach is a combination of several machine learning models that will generate more accurate and reliable result compared to the single classifier. For classifiers, the information they are able to extract from the data is different; when multiple classifiers are combined, the limitations of individual classifiers can be overcome, including overfitting, lack of generalization ability, and biased

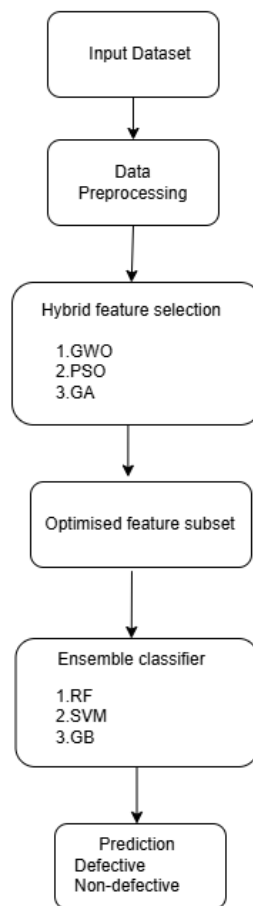


Figure 3.3. Architecture of hybrid approach

prediction. This study employed the ensemble framework in order to benefit from the advantages of several base learners, enhancing the ability to classify defects over various software datasets.

The stacking approach was used with an ensemble of several base models, with their predictions being aggregated by a meta-classifier to obtain the final prediction. The dataset was preprocessed before training, which involved normalization, class balancing, and optimum feature selection by hybrid GWO–PSO–GA technique. The feature set was then refined and fed to the ensemble model to achieve a greater learning efficiency and prediction accuracy. This holistic approach not only boosted the classification accuracy, but also strengthened the stability of the model in handling complex and imbalanced software defect datasets.

3.2.5 Dataset Description

Table 3.1. Dataset Description

Project Name	Programming Language	Number of Instances	Defective Instances (%)	Description
CM1	C	498	9.7	NASA spacecraft instrument
JM1	C	10,885	19	Real-time predictive ground system
KC1	C++	2,109	15.4	Storage management system for ground data processing
KC2	C++	522	20	Science data processing system
PC1	C	1,109	6.8	Flight software for an Earth-orbiting satellite

3.2.6 Evaluation Metrics

While accuracy is a measure of the percentage of correctly classified instances, it is not always a good indicator of the overall performance of a model when dealing with imbalanced data. Precision is the ratio of the number of correctly predicted faulty modules to the total number of fault prediction under certain small false positive fraction. Recall or sensitivity reflects how good the model is in picking faulty modules but keeping a minimal false negative. F1-score is a harmonic mean of precision and recall as a balanced value, particularly in cases of class imbalance. These values give a measure of model performance in general and show how robustness and the ability to generalize across datasets are improving defect detection.

CHAPTER 4

RESULTS AND DISCUSSION

This section shows all the results that we derived after doing the extensive research on SDP and finding the ways for a better defect prediction. The results combines the derivations from all three methods that are fedprox, moon-fed and hybridisation approach.

4.1 Results of FedProx model

The predictive performance of the constructed model was evaluated employing four commonly used metrics in classification, namely accuracy, precision, recall, and F1-score. These metrics therefore give a global view of the capability of the model to classify defective and non-defective software modules correctly. The final performance was computed for each dataset, CM1, KC1, and PC1, independently at the end of all federated communication rounds. The following table presents these results.

Table 4.1. Performance of FedProx based FL approach

Dataset/Clients	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
cm1	85.42	93.42	96.67	84.22
kc1	93.58	90.73	97.17	93.84
pc1	95.73	92.36	99.81	95.94

The fedProx-based FL model proposed in this work was tested on three software defect datasets—CM1, KC1, and PC1—and achieved uniformly robust performance, as presented in Table 1. It attained an F1-score of 84.22% on CM1, 93.84% on KC1, and 95.94% on PC1 with recall values very close to and higher than 96% in all datasets. These results validate that the model is robust even when project properties differ. Which means that the model is generalising well across different datasets.

4.2 Results of MOON-Fed Model

Study measures the predictive power of the model by making use of four major metrics—Accuracy, Precision, Recall, and F1-score—the combined effect of which provides the comprehensive analysis of the model's potential in classifying the defective and non-defective software modules. Table 1 shows the results obtained after all the rounds of training.

The MOON-based federated learning model that we created showed promising results in all three datasets, showing the increase in the model's strength and flexibility in dealing with different kinds of data sets. With F1-scores of 85.58% for the CM1 dataset, 91.12% for the KC1 dataset, and 95.23% for the PC1 dataset, the model demonstrates good generalization ability and efficient classification of defects even in heterogeneous data sets. High recall values also show the sensitivity of the model to detect the defect prone

Table 4.2. MOON-based FL Approach Metrics

Dataset/Clients	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
cm1	86.40	92.70	79.48	85.58
kc1	90.91	89.51	92.79	91.12
pc1	95.01	92.11	98.58	95.23

module, balanced precision verifying the reliability to suppress false positives. Such results demonstrate that the MOON based model is not only stably converging across different clients, but it still possesses strong predictive power in privacy preserving de-centralized environments.

4.3 Results of Hybrid GWO–PSO–GA Model

To evaluate the performance of the suggested hybrid GWO-PSO-GA feature selection approach, the Random Forest (RF), Support Vector Machine (SVM), and Gradient Boosting (GB) classifiers were trained individually on the optimized feature subset (i.e., the best subset of features determined by the hybrid optimization procedure). The classifiers were then ensembled in a stacking ensemble approach to leverage their individual strengths and counter prediction bias. The reason behind such an ensemble is to improve prediction accuracy and generalization capability in software defect prediction tasks.

Following common SDP evaluation practice [19, 2], the ensemble method was experimented with four main performance measures—Accuracy, Precision, Recall, and F1-score—on five standard benchmark datasets: CM1, JM1, KC1, KC2, and PC1. The experimentation is important in quantifying the improvements achieved through feature optimization and ensemble learning methods. The results are presented in table , which provides the performance measures of the proposed hybrid model in terms of GWO-PSO-GA stacking methodology.

Table 4.3. Hybrid Approach Performance Across Multiple Datasets

Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
cm1	96	94.16	97.73	96
jm1	92	92.5	90.1	91.63
kc1	92.8	90.12	96.23	93.08
kc2	90.32	89.06	92	90.48
pc1	96.7	94.5	98.65	96.53

All those measurements are compared on all datasets in a graphical form in figure that highlights the impact of the hybrid GWO-PSO-GA feature selection by stacking ensemble learning. The graph presents all comparative study for Accuracy, Precision, Recall, and F1-score which shows the efficiency of the proposed method to enhance the software defect prediction. This visualization adopts understanding the relative performance differences over different datasets and shows the consistency and stability of the stacking ensemble model.

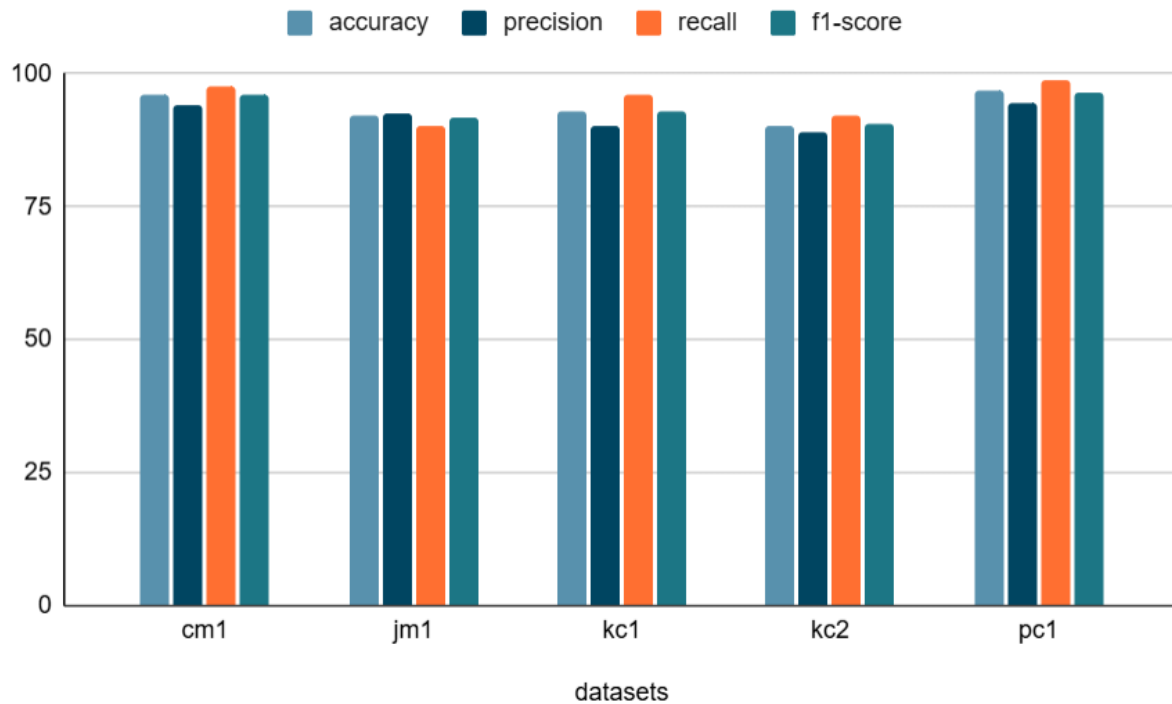


Figure 4.1. Proposed Approach Performance Metrics

4.3.1 Impact of Combining GWO, PSO, GA, and Stacking Ensemble

Where earlier models were dependent entirely on Random Forest (RF) for classification, the incorporation of a stacking ensemble of RF, SVM, and Gradient Boosting increases predictive stability as well as accuracy. Comparative analysis of GWO+PSO+RF, GWO+PSO+GA+RF, and the suggested model GWO+PSO+GA with Stacking Ensemble model is carried out to determine their performance in various datasets through Accuracy, Precision, Recall, and F1-score.

Table 4.4 also validates the accuracy comparisons of different combinations of GWO, PSO, and GA in connection with software defect prediction on different data samples. Also, the findings indicate that although adding an overall GA in the model has the potential of not always translating into accuracy increase, like in KC1 where GWO+PSO has 92% but again GWO+PSO+GA slipped a bit to 91.25%. However, the stacking-based ensemble learning incorporation appeared to give most often higher accurate results proving its ability to make predictions more accurate. The combination of all three optimizers using the stacking ensemble recorded a better 92.8% accuracy. Similar increasing trends were reported in CM1 (96%) and PC1 (96.7%), highlighting the scalability and robustness of the proposed framework.

Table 4.4. Accuracies of combination of GWO, PSO and GA

Dataset	GWO+PSO	GWO+PSO+GA	GWO+PSO+GA with stacking ensemble
cm1	94.2	94.8	96
jm1	90.87	90.89	92
kc1	92	91.25	92.8
kc2	89.95	90.21	90.32
pc1	95.1	96	96.7

Feature selection is a significant factor in improving software defect prediction by re-

moving redundant and irrelevant features, resulting in better generalization of the model. In this subsection, we describe how the hybrid GWO-PSO-GA feature selection and stacking ensemble model improves performance compared to standalone classifiers (Random Forest, SVM, and Gradient Boosting) on various datasets (CM1, JM1, KC1, KC2, PC1). The objective is to determine that the new hybrid feature selection method improves the global performance of the standalone classifiers by improving accuracy, precision, recall, and F1-score, but also minimizes overfitting.

Table 4.5 is a comparison of the precision values of the three approaches—GWO+PSO, GWO+PSO+GA, and proposed GWO+PSO+GA with stacking ensemble—highlighting the effectiveness of the model in classifying defective software modules with high precision. Incorporation of GA with the GWO+PSO approach generally results in enhanced precision, particularly in cases like CM1 (91.03%) and KC2 (88.5%), even resulting in more accurate feature selection and enhanced classification. Utilization of a stacking ensemble also further enhances these gains to produce the highest precision values in CM1 (94.16%) and JM1 (92.5%). These results show the benefit of the utilization of classifier diversity to reduce the number of false positives. Although precision in PC1 is relatively lower compared to some datasets, the consistency of the ensemble model across all datasets is a testament to its strength and usability in real-world software defect prediction tasks.

Table 4.5. Precision of combination of GWO, PSO and GA

Dataset	GWO+PSO	GWO+PSO+GA	GWO+PSO+GA with stacking ensemble
cm1	90.21	91.03	94.16
jm1	90.95	91.11	92.5
kc1	91.39	89.66	90.12
kc2	87.97	88.5	89.06
pc1	97.42	94	94.5

4.3.2 Statistical Significance Analysis of Model Performance

Boxplots of accuracy and F1-score give a holistic view of performance variability across datasets. The results show that GWO+PSO+GA with a stacking ensemble performs better consistently compared to the other two setups, as its median values are higher and its variability is lower. The stacking ensemble method increases model stability with better predictive performance. Furthermore, the hybrid GWO+PSO+GA model, even without stacking ensemble, exhibits enhancements over the GWO+PSO setup, supporting the efficacy of combining several metaheuristic algorithms. These results further support the statistical significance of the proposed method in software defect prediction.

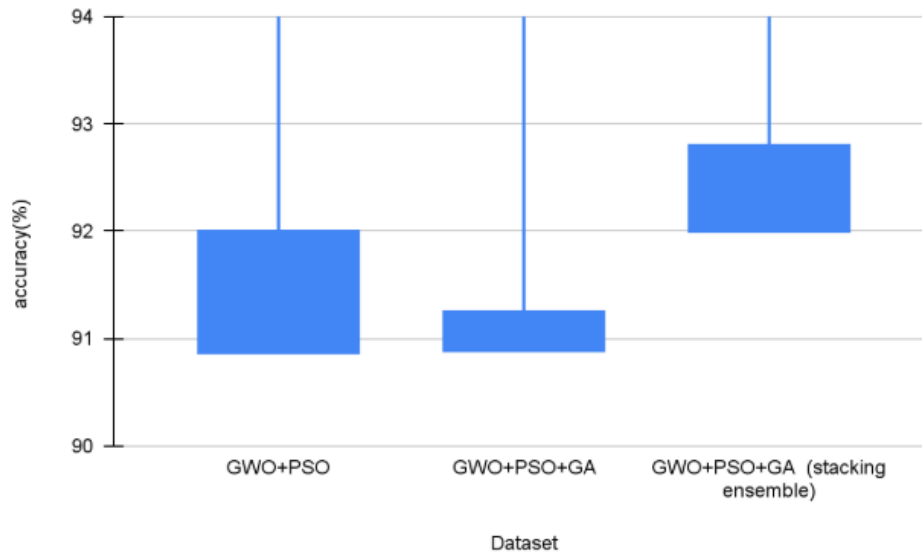


Figure 4.2. Boxplot analysis of accuracy across model combinations

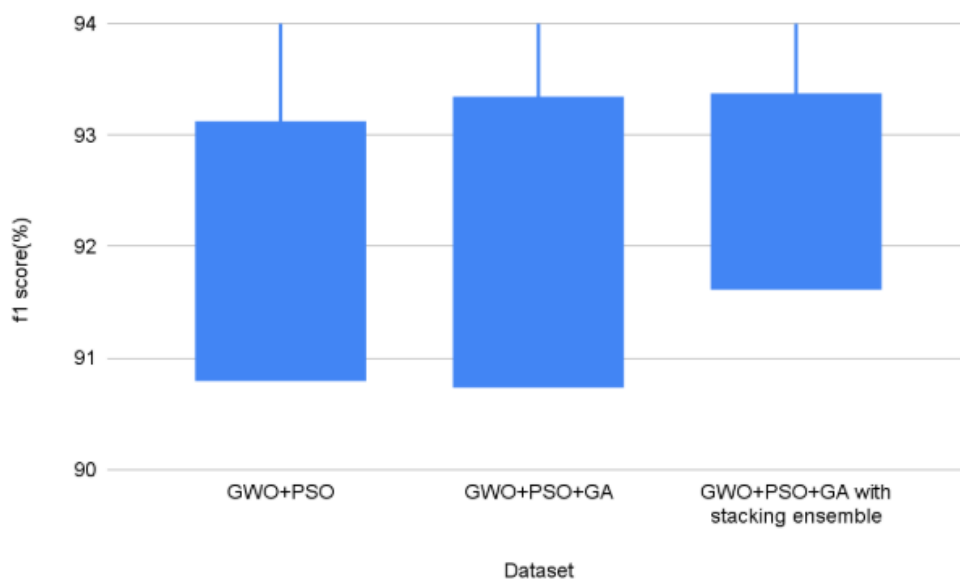


Figure 4.3. Boxplot analysis of F1-score across model combinations

4.3.3 Friedman Test and Bonferroni Post-hoc Analysis

To determine whether the accuracy differences among the three model variants are statistically significant, the Friedman test was applied as a non-parametric comparison method across the five datasets CM1, JM1, KC1, KC2, and PC1. The three models considered were GWO+PSO using Random Forest, GWO+PSO+GA using Random Forest, and the proposed GWO+PSO+GA with Stacking Ensemble. The accuracy values reported in Table 4.4 were ranked for each dataset, where the model with higher accuracy received the higher rank. The rank sums obtained were 6 for GWO+PSO, 9 for GWO+PSO+GA, and 15 for GWO+PSO+GA with Stacking Ensemble, showing that the proposed model consistently achieved the strongest ranking across the datasets.

The Friedman test evaluates the null hypothesis that all three models perform similarly and that any observed difference in accuracy is due to chance. In this analysis, the number of datasets was $n = 5$ and the number of models was $k = 3$. Using the Friedman statistic, the computed value was $\chi_F^2 = 8.40$ with $df = k - 1 = 2$. The corresponding p-value was $p = 0.01499$. Since this value is less than the common significance level of $\alpha = 0.05$, the null hypothesis was rejected. This indicates that at least one of the compared models has a statistically significant difference in accuracy, supporting the claim that the proposed stacking ensemble approach improves software defect prediction performance.

After the Friedman test confirmed a significant overall difference, a Bonferroni post-hoc analysis was used to identify which model pairs differed significantly while controlling for multiple comparisons. The pairwise comparison between GWO+PSO and GWO+PSO+GA produced $Z = 0.79$ and $p = 0.428$, indicating no significant difference. The comparison between GWO+PSO and the proposed GWO+PSO+GA with Stacking Ensemble produced $Z = 2.53$ and $p = 0.0113$, which is significant under the Bonferroni-adjusted threshold of 0.0167. The comparison between GWO+PSO+GA and the proposed stacking model produced $Z = 1.74$ and $p = 0.0818$, which was not statistically significant. These results suggest that adding GA alone does not produce a major statistical improvement, but combining GWO, PSO, GA, and stacking ensemble learning provides a meaningful and statistically supported improvement over the basic GWO+PSO approach.

The suggested GWO+PSO+GA with stacking ensemble method has proved to show remarkable enhancements in predicting software defects. Nevertheless, like any other optimization-based complex model, there are some aspects that need to be addressed. Computational complexity is one of the main factors. The confluence of three meta-heuristic algorithms, namely GWO, PSO, and GA, and ensemble learning makes the overall computation more demanding. Though it leads to even better accuracy, the processing time may take longer than for a basic model.

4.4 Comparative Analysis

The three techniques which were evaluated in this chapter have varying levels of effectiveness for software defect prediction. The FedProx-based federated learning model shows strong privacy-preserving performance in terms of both accuracy and recall on all three datasets, CM1, KC1, and PC1, suggesting that it can accurately identify defect-prone modules. The MOON-Fed model also delivers good performance in the decentralized setting and enhances representation learning by ensuring uniformity of the client models, however, it is not uniform across the datasets, particularly in terms of recall and F1-Score. The Hybrid GWO-PSO-GA model with the stacking ensemble outperforms the proposed models due to the optimal selection of features and diversity of classifiers, which helps to remove irrelevant features and enhance the stability of predictions.

The comparative results show that the hybrid GWO-PSO-GA stacking ensemble approach is the best of the 3 approaches discussed in Sections 4.1, 4.2 and 4.3. FedProx and MOON-Fed are useful for privacy-preserving and federated settings, but they heavily rely on the distribution of client data and model convergence. The proposed hybrid approach is more consistent in various datasets like, CM1, JM1, KC1, KC2 and PC1 and shows better accuracy, precision, recall and f1 score values. Hence, the hybrid of metaheuristic feature selection and ensemble learning approaches gives a more powerful and scalable approach than federation based optimization approaches for software defect prediction.

17

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

In this thesis, privacy-aware and performance-driven approach to software defect prediction was investigated by examining federated learning models and a hybrid metaheuristic ensemble model. This work was mainly concentrated on three major techniques, Federated Learning (FL) using FedProx, MOON-Fed federated learning, and the proposed Hybrid GWO-PSO-GA feature selection with stacking, which is a federated learning technique. The results demonstrate that federated learning models are useful when data privacy and decentralized training are of importance, while the proposed hybrid approach results in a better predictive performance, by leveraging optimized feature selection and classifier diversity.

The FedProx model successfully performed well in all the three datasets (CM1, KC1 and PC1) and was able to deal with heterogeneous project data in a federated environment. The other model, MOON-Fed, also had encouraging results, enhancing the representation learning at the client level and being generally stable on other datasets. But, the Hybrid GWO-PSO-GA model with stacking ensemble was best in all the five models namely: (CM1), (JM1), (KC1), (KC2), and (PC1). The proposed method achieved better accuracy, precision, recall and F1-score by employing GWO, PSO and GA for feature selection and stacking Random Forest, SVM and Gradient Boosting.

The comparative and statistical analyses further support the effectiveness of the proposed model. The Friedman test showed a statistically significant difference among the model variants, and the Bonferroni post-hoc analysis confirmed that the proposed GWO-PSO-GA with stacking ensemble significantly improves performance over the basic GWO+PSO approach. Therefore, the proposed framework can be considered a robust and scalable solution for software defect prediction. It reduces the effect of irrelevant features, improves classification stability, and provides better generalization across different datasets.

5.2 Limitations

While the proposed approaches are promising, they are not without its limitations. The evaluation was performed on three datasets used from the NASA PROMISE repository, and these might not capture the complexity of real industrial software systems. The framework, to that extent, is yet to be deployed within a live environment, and scalability in terms of communication cost and client heterogeneity is yet to be proven so far.

In spite of the promising results, this work is aware of certain limitations that will be useful for guiding future research work. The experimental investigations included only three widely accepted benchmark datasets, which when standardized, do not truly represent the coarseness encompassed in large industrial software systems. Scalability analysis of the proposed framework using additional clients, diverse data modalities (e.g.,

10

58

text-based measures or process traces), and higher non-IID scenarios could be attempted in the future. Beyond that, reduction of the overhead further without the loss of accuracy may be attempted by the running of communication efficient strategies such as quantization or sparse model updates.

The suggested GWO+PSO+GA with stacking ensemble method has proved to show remarkable enhancements in predicting software defects. Nevertheless, like any other optimization-based complex model, there are some aspects that need to be addressed. Computational complexity is one of the main factors. The confluence of three meta-heuristic algorithms, namely GWO, PSO, and GA, and ensemble learning makes the overall computation more demanding. Though it leads to even better accuracy, the processing time may take longer than for a basic model. However, with optimizations and parallel computing techniques, efficiency of execution can be further improved.

The other one is the model's capability to map across different datasets. The framework has been extensively evaluated with NASA datasets with high F1-scores and accuracy. Our hybrid approach has a better generalizability and when we will test it further with the other datasets which are are of software defects we will be clearly able to understand the real world scenarios. In the near future we can test our model on the more complex datasets that contains a variety of characteristics. This will give more accuracy but may be slower than a normal model. But when implemented with optimisation and using parallel computing methods it can be made even more efficient to execute.

5.3 Future Works

In the future, the proposed framework could be tested on a larger and diverse number of software defect datasets since recent SDP research did indicate there is a need for further testing and future challenges [2]. The present work used the benchmark datasets that have been well known, but more testing of the model with industrial, open-source and real-time project datasets would help in the ability of the model to generalize. The model could be experimented with other datasets with different coding defects, different scales of a project, different numbers of defects distributed across the code of a project, and different histories of development.

The other direction to improve is to enhance the computational efficiency of the hybrid GWO-PSO-GA with stacking ensemble approach. Using different strategies like ensemble learning classifiers and algorithms that are metaheuristic in nature there will be an increase in the complexity of training which we can further work on using parallel processing, adaptive stopping criteria, efficient in terms of communication lightweight feature selection and learning strategies. These enhancements can lead to quicker execution time and to the extension of the framework to fit in large scale software engineering environments.

Future research can also integrate the proposed model with more advanced privacy-preserving and explainable artificial intelligence techniques. Federated learning can be combined with secure aggregation, differential privacy, or encrypted model updates to strengthen data protection. In addition, explainability methods can be added to identify the most influential software metrics responsible for defect prediction. This would make the system more transparent, trustworthy, and useful for developers and project managers in real-world software quality assurance.

CHAPTER A

APPENDIX TITLE

BIBLIOGRAPHY

- [1] M. K. Thota, F. H. Shajin, and P. Rajesh, "Survey on software defect prediction techniques," *International Journal of Applied Science and Engineering*, vol. 17, no. 4, pp. 331–344, 2020.
- [2] Z. Li, J. Niu, and X.-Y. Jing, "Software defect prediction: future directions and challenges," *Automated Software Engineering*, vol. 31, no. 1, p. 19, 2024.
- [3] E. N. Akimova, A. Y. Bersenev, A. A. Deikov, K. S. Kobylkin, A. V. Konygin, I. P. Mezentsev, and V. E. Misilov, "A survey on software defect prediction using deep learning," *Mathematics*, vol. 9, no. 11, p. 1180, 2021.
- [4] R. Ponnala and C. R. K. Reddy, "Software defect prediction using machine learning algorithms: Current state of the art," *Solid State Technology*, vol. 64, no. 2, pp. 6541–6556, 2021.
- [5] A. Wang, Y. Feng, M. Yang, H. Wu, Y. Iwahori, and H. Chen, "Cross-project software defect prediction using differential perception combined with inheritance federated learning," *Electronics*, vol. 13, no. 24, p. 4893, 2024.
- [6] R. Malhotra, A. Bansal, and M. Kessentini, "Deployment and performance monitoring of docker based federated learning framework for software defect prediction," *Cluster Computing*, vol. 27, no. 5, pp. 6039–6057, 2024.
- [7] H. Chen, L. Yang, and A. Wang, "Efficient cross-project software defect prediction based on federated meta-learning," *Electronics*, vol. 13, no. 6, p. 1105, 2024.
- [8] X. Hu, M. Zheng, R. Zhu, X. Zhang, and Z. Jin, "Fed-olf: Federated oversampling learning framework for imbalanced software defect prediction under privacy protection," *IEEE Transactions on Reliability*, pp. 1–15, 2025.
- [9] H. Yamamoto, D. Wang, G. K. Rajbahadur, M. Kondo, Y. Kamei, and N. Ubayashi, "Towards privacy preserving cross-project defect prediction with federated learning," in *Proceedings of the 2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2023, pp. 485–496.
- [10] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems*, vol. 2, 2020, pp. 429–450.
- [11] Y. Yang, X. Hu, Z. Gao, J. Chen, C. Ni, X. Xia, and D. Lo, "Federated learning for software engineering: a case study of code clone detection and defect prediction," *IEEE Transactions on Software Engineering*, vol. 50, no. 2, pp. 296–321, 2024.
- [12] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 713–10 722.

- [13] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah, "Metaheuristic algorithms: A comprehensive review," in *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*. Academic Press, 2018, pp. 185–231.
- [14] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.
- [15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.
- [16] S. Mirjalili, "Genetic algorithm," in *Evolutionary Algorithms and Neural Networks: Theory and Applications*. Cham: Springer International Publishing, 2019, pp. 43–55.
- [17] Y. N. Soe, P. I. Santosa, and R. Hartanto, "Software defect prediction using random forest algorithm," in *2018 12th South East Asian Technical University Consortium (SEATUC)*, vol. 1, 2018, pp. 1–5.
- [18] S. Goyal, "Effective software defect prediction using support vector machines (svms)," *International Journal of System Assurance Engineering and Management*, vol. 13, no. 2, pp. 681–696, 2022.
- [19] F. Matloob, T. M. Ghazal, N. Taleb, S. Aftab, M. Ahmad, M. A. Khan, S. Abbas, and T. R. Soomro, "Software defect prediction using ensemble learning: A systematic literature review," *IEEE Access*, vol. 9, pp. 98 754–98 771, 2021.
- [20] A. M. Akbar, R. Herteno, S. W. Saputro, M. R. Faisal, and R. A. Nugroho, "Optimizing software defect prediction models: Integrating hybrid grey wolf and particle swarm optimization for enhanced feature selection with popular gradient boosting algorithm," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, 2024.
- [21] H. Wang, B. Arasteh, K. Arasteh, F. S. Gharehchopogh, and A. Rouhi, "A software defect prediction method using binary gray wolf optimizer and machine learning algorithms," *Computers and Electrical Engineering*, vol. 118, p. 109336, 2024.
- [22] R. S. Wahono and N. Suryana, "Combining particle swarm optimization based feature selection and bagging technique for software defect prediction," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 5, pp. 153–166, 2013.
- [23] M. K. Suryadi, R. Herteno, S. W. Saputro, M. R. Faisal, and R. A. Nugroho, "Comparative study of various hyperparameter tuning on random forest classification with smote and feature selection using genetic algorithm in software defect prediction," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, pp. 137–147, 2024.
- [24] S. I. Ayon, "Neural network based software defect prediction using genetic algorithm and particle swarm optimization," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, 2019, pp. 1–4.