FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 9 No. 3, March, 2025, pp 140 - 153
DOI: https://doi.org/10.33003/fjs-2025-0903-3085

SOFTWARE DEFECT PREDICTION MODELS FOR MITIGATING THE RISK OF BUG RECURRENCE IN LINES OF CODES: A SYSTEMATIC REVIEW

***[1]Shamsuddeen Muhammad Abubakar, [2]Abdulmajid Babangida Umar and [1]Abdullahi M. Ibrahim**

[1]Faculty of Computing, Federal University Dutse, Jigawa - Nigeria.
[2]Faculty of Computing, Yusuf Maitama Sule University, Ado Bayero House Kofar Nassarawa, Kano - Nigeria.

*Corresponding authors' email: salsabil012@gmail.com; abumar@yumsuk.edu.ng

ABSTRACT

The custom of recent research in the field of software defect prediction is to display frameworks that provide software quality assurance teams with the ability to concentrate purely on software defect-prone codes. This has enabled software development time and maintenance activities to be managed more efficiently. Previous studies involved intricate software defect prediction datasets and methods. However, more detailed and inclusive insight into the state-of-the-art software defect prediction research method is needed. This systematic literature review aimed at evaluating the trends in the research field, the algorithms deployed, the methods utilised, the datasets used, the feature selection techniques mostly applied, the software metrics used, and the evaluation measures used in software defect prediction studies between 2017 and June 2023. Upon the application of selection and rejection criteria, 104 studies published on software defect prediction from January 2017 to June 2023 were evaluated. It was shown that 82 published articles on software defect prediction applied classification techniques, or about 78% of the total study. Meanwhile, the estimation algorithm accounts for 3% of the published articles on software defect prediction. Association methods have 2%. Clustering methods have four total published articles, with 4%. Finally, data processing has 14 published articles, resulting in 13%. The distribution of datasets used in software defect prediction found that 21 published articles utilised private datasets, a result of about 40%. Public datasets have a total of 32 published articles on software defect prediction, which resulted in 60%. Additionally, it was found that Naïve Bayes dominates the most deployed methods with 24%, followed by Decision Tree with 20%. Also, static code metrics constitute 51% of the metrics used in a primary study, while object-oriented metrics produce 9% of the metrics. Composite metrics generated 23%, and miscellaneous metrics produced 17% of the metrics used in software defect prediction. The distribution of filter-based feature selection techniques utilised from our primary studies shows that information-gained feature selection techniques constitute 22% of the primary studies, followed by correlation-based techniques at 13%. The distribution of wrapper-based feature selection techniques used in software defect prediction has shown that genetic algorithms have 25%, followed by particle swamp algorithms with 16% of the number of published articles.

Keywords: Feature selection techniques, Software defect prediction, Software metrics, Systematic literature review

INTRODUCTION

As is prevalent in the software manufacturing process, defects in software are an inevitable part of the development process. The majority of software organisations struggle to create qualitative, error-free software. End users receive a lot of complex software packages that lack essential capabilities. Because end users require qualitative, defect-free software, software testing has become a popular field of study in recent years.

Despite software testing, it is impossible to completely rule out the possibility of a software module defect reoccurring. When the software product is being tested and deployed to clients, defects may resurface. When found, vulnerabilities in software can negatively affect the whole software's quality, cost, maintenance schedules, and reliability. According to Bergmane *et al.* (2018), software failure typically results in financial loss as well as hazards to people's lives and property. These can be caused by incorrect technique selection, unsteady settings, and inadequate staff training, or missing design requirements.

This research is set to study different software defect prediction models that are used to mitigate the risk of re-occurrence of bugs in subsequent releases of software before reaching the end-user, with the aim of making a critical analysis between them. The systematic literature review will span from 2017 to 2023.

Background and Motivation

In the past thirty years, there have been two widely accepted methods for identifying and resolving software defect issues (Wahono *et al.*, 2015). These two approaches are prediction-based and legacy-based. Up until recently, it was shown that software flaws may be found and fixed using a search-based method (Malhotra *et al.*, 2023). This section presents similar researches conducted as well as distinguishing it from the prior studies.

Hosseini *et al.* (2017) performed state-of-the-art cross-product Cross Product Defect Prediction (CPDP) by examining the previous models, software metrics used, datasets deployed, different techniques employed, and defect prediction model performances. Their research found that the nearest-neighbor and decision tree models proved to present better performance accuracy while using CPDP. Naive Bayes models, on the other hand, have average prediction accuracy.

Malhotra *et al.* (2017) reviewed previous studies from January 1992 to December 2015 by investigating software engineering dimensions such as defect interpretation, maintainability, defect prediction models, and defect-proneness. Seventy-eight (78) research papers were examined upon applying search-based algorithms to evaluate parameter settings, fitness functions, and validation techniques. More reviews are needed to fill in the vacuum left by the research.

Sobrinho *et al.* (2018) investigated the bad smells in the code and the work done by early researchers. What kind of bad smell in the code is causing the issues? The findings also point to future efforts against the bad smell. In the review conducted by Malhotra (2018), sixteen (16) hybrid-based methods were applied to ten (10) machine learning algorithms for the construction of software defect prediction models. Object-oriented metrics were tested on datasets with seventeen (17) inputs. The findings of statistical analysis have shown the predictive efficiency of using search-based and hybrid techniques for classifying software defects.

Son *et al.* (2019) conducted a systematic review of 98 studies between 1995 and 2018. Various approaches to software defect prediction, such as data collection, pre-processing techniques, performance evaluation methods, security-related issues with defect models, and the number of studies conducted on cross-project software defect prediction, address the level of defect intensity. Li *et al.* (2020) performed a systematic literature review on 49 research studies that span 2456 different experimental results. The review was conducted between January 2000 and March 2018. Confusion matrices and the Matthews Correlation Coefficient (MCC) were employed to examine the prediction performance across these studies. Their results proved that unsupervised models can be compared with supervised models for both within-project and cross-project prediction.

Pandey *et al.* (2021) performed a systematic literature review on 154 articles from 1990 to June 2019. Several properties of machine-learning algorithms were investigated. The research presented a summary of different machine learning algorithms, datasets used, feature reduction techniques employed, and performance evaluations between machine learning techniques and statistical techniques used in defect prediction. Malhotra *et al.* (2023) examined seventy-two (72) published research papers between January 2000 and December 2021. The review evaluates several techniques, experimental procedures, fitness functions, performance evaluation functions, threats to validity, and statistical analysis used in the studies that applied hybrid techniques. The aforementioned research has ascertained the use of hybrid techniques and their efficiencies in predicting software defects. However, further research is needed as the study retrained itself only on hybrid-based methods.

Pachouly *et al*. (2023) conducted a systematic review of 74 published articles ranging from 1990 to 2007. The survey examined mainly the datasets used, methods deployed, and software metrics used. The major weaknesses reported have shown that many studies have applied private data sources, which are impracticable and lead to fruitless research. The survey suggested that machine learning techniques are predominantly applied in software defect prediction. Six types of metrics are commonly used: software class, module, file, component process, and quantitative level. The review further proved that machine learning and statistical approaches are coupled together for the construction of software defect prediction models.

Previous studies mainly focused on defect classification using datasets from open-source repositories. The surveys conducted mainly concentrated on differentiating between modern and local approaches for evaluating software defects. One important aspect overlooked by prior research was the computational complexity of the defect models, tuning the appropriate parameter settings, and assessing the severity level of the defective software module. Therefore, an extensive survey is required aimed at the feature selection technique used, datasets applied, data validation technique, defect interpretation and prediction approaches, and tools used, thereby recommending further research. Hence, this systematic literature review will pay more attention to the feature selection technique chosen, datasets used, data validation methods, defect interpretation and prediction propositions, and tools, which we will recommend for future research.

Review Methodology

The approach followed in conducting this systematic literature review presents the methods used in predicting software defect prediction models. Systematic literature review (SLR) has been a proven method in software engineering as a process of evaluating all available research with the goal of providing answers to the research question (Kitchenham & Charters 2007). This review is based on the initial rules proposed by Kitchenham and Charters (2007). The systematic literature review approach is conducted based on three (3) broadways. They are in the planning, conducting, and reporting stages. Initially, the vital part of conducting the systematic review, such as the objectives, is identified in the introduction section. Secondly, the review protocol is drafted to reduce the investigator's partiality. It describes the research questions, data exploration techniques, inclusion and exclusion criteria, data quality evaluation, and finally, data extraction processes. Figure one will show the steps to be followed in generating a systematic literature review as derived by(Wahono, 2015)

Figure 1: Steps to be followed in Conducting a Systematic Literature Review (Source: Wahono *et al.,* 2015)

Research Questions

The research questions (RQ) were designed to guide the review in maintaining its focus, thereby obtaining the aim and objectives. This systematic literature review is targeted to answer the following research questions and motivations shown in Table 1.

Table 1: Research Questions on Literature Review

RQ	Research Question	Motivation
RQ1.	What are the machine learning classification algorithms used in software defect construction?	Identify the most predominantly used machine learning algorithms applied in software defect prediction.
RQ2.	What is the distribution of studies that have utilised feature selection techniques for the construction of software defect prediction models?	Identify which of the three feature selection techniques is used in developing a software defect prediction model.
RQ3	At what percentage do reputable journals publish articles on software defect prediction?	Identify the percentage of reputable journals that have published research on software defect prediction.
RQ4	What are the datasets used for software defect prediction?	Identify the datasets frequently used in software defect prediction.
RQ5	What are the software metrics used for software defect prediction?	Identify the various software metrics used in software defect prediction.
RQ6	What are the validation techniques used in software defect prediction?	Identify the commonly used validation techniques used in software defect prediction.

RQ1 will trigger the major classification algorithms used in machine learning for the development of defect prediction models. RQ2 will present the yearly distribution of studies that employed feature selection techniques while constructing the software defect prediction models. RQ3 will examine several datasets used in software defect prediction. RQ5 will showcase different software metrics used in software defect prediction. Finally, this work will address the different validation techniques used in software defect prediction development.

Literature Collection

The literature was fetched using six scientific digital libraries, comprising IEEE eXplore, ACM Digital Library, Science Direct, Arxiv, Wiley, and Springer. Chen *et al*. (2010) proposed the aforementioned digital libraries as per the author's research expertise and criteria. The digital libraries are as follows:

i. IEEE eXplore [*ieeexplore.ieee.org*]
ii. ACM [*dl.acm.org*]
iii. ScienceDirect [*sciencedirect.com*]
iv. Springer [*Springerlink.com*]
v. Wiley [*onlinelibrary.wiley.com*]
vi. Axiv [*arxiv.org*]

The following phrase were used to search for a better result:

i. Software x {defect + fault + error + flaw + bug + failure + search-based + legacy-based + prediction-based}
ii. Feature Selection Techniques x {Filter + Wrapper + Embedded} x {prediction + proneness}
iii. Software Metrics x {process + product + object-orientated} x {defect prediction}
iv. Correlated Software Metrics x {defect prediction}

The search string was conducted and kept alternating, but the original keyword was retained. This is because the adapted search string will increase the number of irrelevant searches. The search combinations were adjusted so as to get the required results from each database. The search digital library databases were checked using title, keyword, and research abstract. The search was restricted to only the research conducted between January 2017 and June 2023. Two types of publications were considered during the search: conference proceedings and journal papers. Finally, only articles published in English are considered. Figure 3 shows a flowchart of how we conducted the search for our primary research.

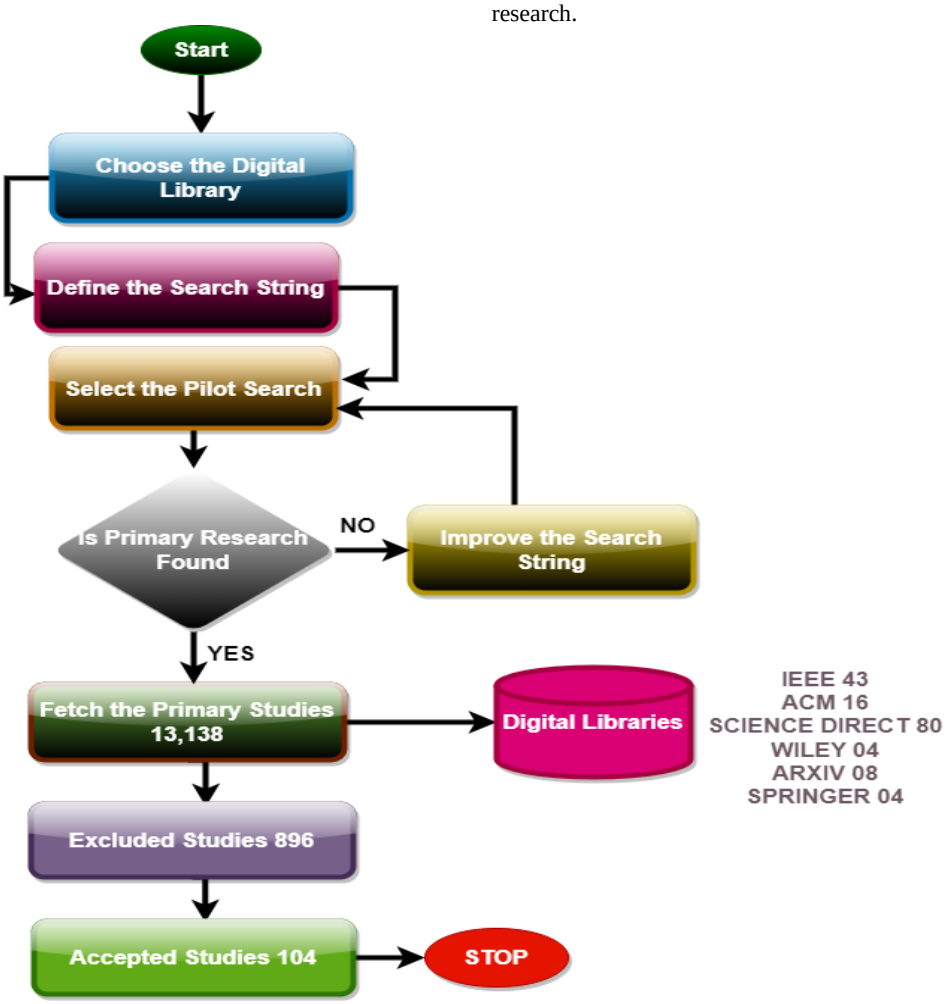

Figure 2: The Search Strategy for the main Research

Statistical Analysis of the Search

Different searches were made using each of the aforementioned digital libraries with the phrases. It has been shown that IEEE and Science Direct constitute about 68% of the total literature generated for the research. It is further to note that it doesn't mean the two contain the highest number of software defect prediction literatures; rather, they have shown the results based on the restricted phrases above. Table 2 below presents a summary of the statistical analysis of the searched literature from the digital libraries.

Table 2: Summary of statistical analysis

SNO.	Library	Fetched Result	Articles with Identical Methodology	After Removing Identical Methodology	Within the range of study literature
1	IEEE	3,450	610	472	43
2	ACM	1,825	100	93	16
3	Science Direct	3,978	253	200	28
4	Wiley	300	14	14	4
5	Arxiv	1,250	95	76	8
6	Springer	2,335	50	41	4
	Cumulative	13,138	1,222	896	104

Based on the search conducted using our criteria, in IEEE we generated 54% of the reviewed literature, ACM has 9%, Science Direct has 23%, Wiley has 1%, Arxiv has 9%, and Springer has 4%. Figure 4 presents a pictorial representation in percentage of the libraries for the research's reviewed literature.

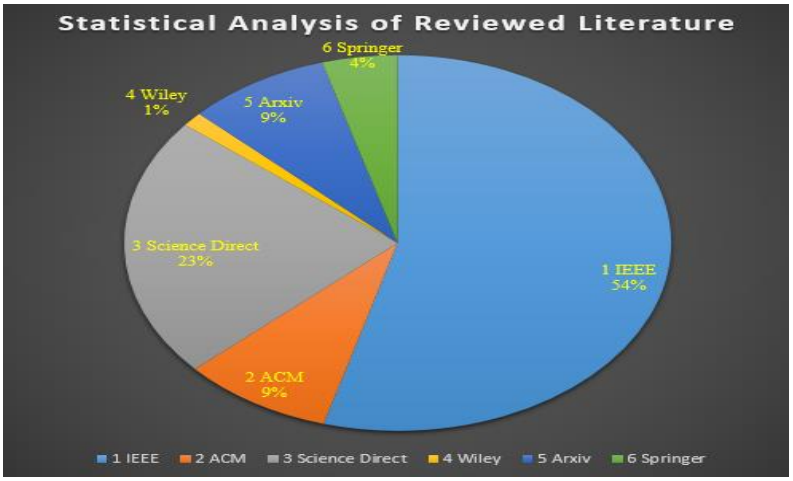

Figure 3: Statistical Analysis of the Literature

Criteria in Choosing the Literature

In order to have a better concise and get scope of the related literature, some certain criteria have been putting into consideration geared towards achieving only the required literatures. In table 2 we divided it into selected literature and rejected.

Adoption Criteria

To successfully absorb a paper, the following criteria were applied:

i. Only a paper written in English is considered
ii. The paper was published between January 2017 and June 2023.
iii. Research that applied either search-based, legacy-based, or prediction-based approaches to software defect prediction
iv. New methods have been explored in software defect prediction.
v. Research that includes prediction performance on each dataset
vi. Research that used meta-analysis to showcase their results
vii. Research with the acronym Software defect, bug, and fault/failure prediction
viii. Research on software metrics such as product process
ix. Research on Correlated Software Metrics Analysis

Table 3: Criteria for Selecting Reviewed Literatures

S/No.	Selected	Rejected
1	Software Defect/Bug/Fault/Failure Prediction-based/search-based/legacy-based/	Consider Noise Data or Pre-processing
2	Software Metrics: Product and Process	Published in Journal and Conference
3	Feature Selection Techniques	Survey or Reviewed Literatures
4	Correlated Software Metrics Analysis	Preliminary Research Findings

RESULTS AND DISCUSSION

Figure 4 presents the yearly distribution of the most noticeable software defect prediction published articles from 2015 to 2023.

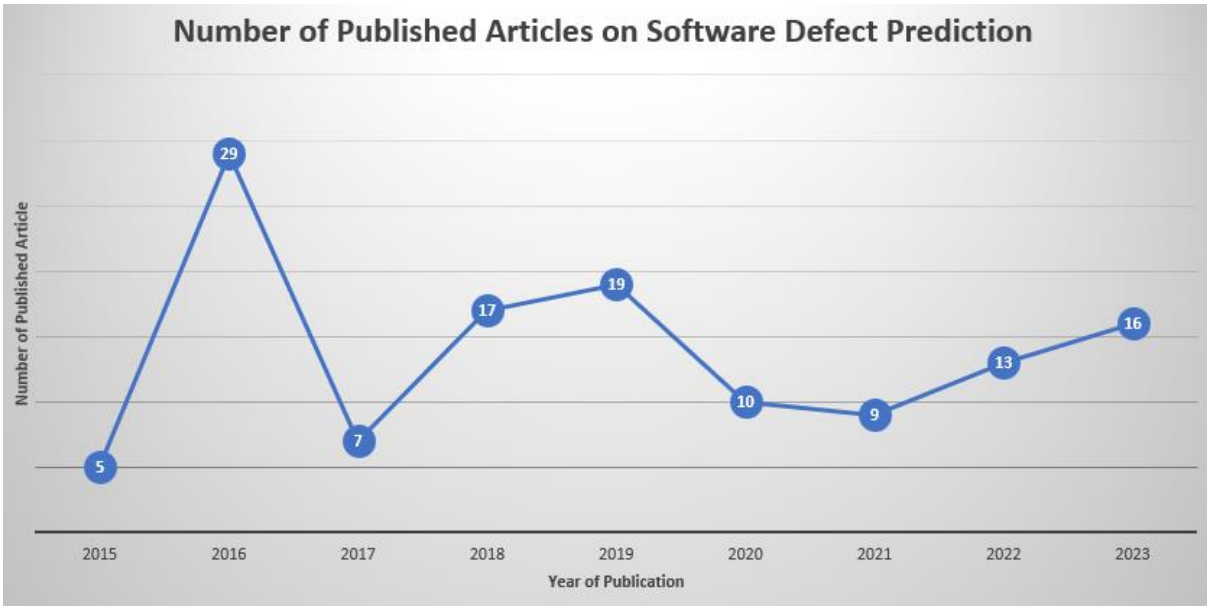

Figure 4: Yearly distribution of published articles on Software Defect prediction

Active Researchers Contributing to the Field

It was observed from our chosen primary literature that prominent researchers who are making significant contributions and are still active in the field of software defect prediction are identified and evaluated. Figure 5 shows researchers with respect to their number of publications. The most leading researchers based on our selected studies are Bowes D., Bhalaji N., Chatterjee S., Ghotra B., Huda S., Ladarji I., Li J., Liu C., Madeyski L., Menzies T., Nam J., Ndenga K., Okutan A., Osman H., Tantithamthavorn C., Thiruvathukail G., Xu Z., and Yathish S.

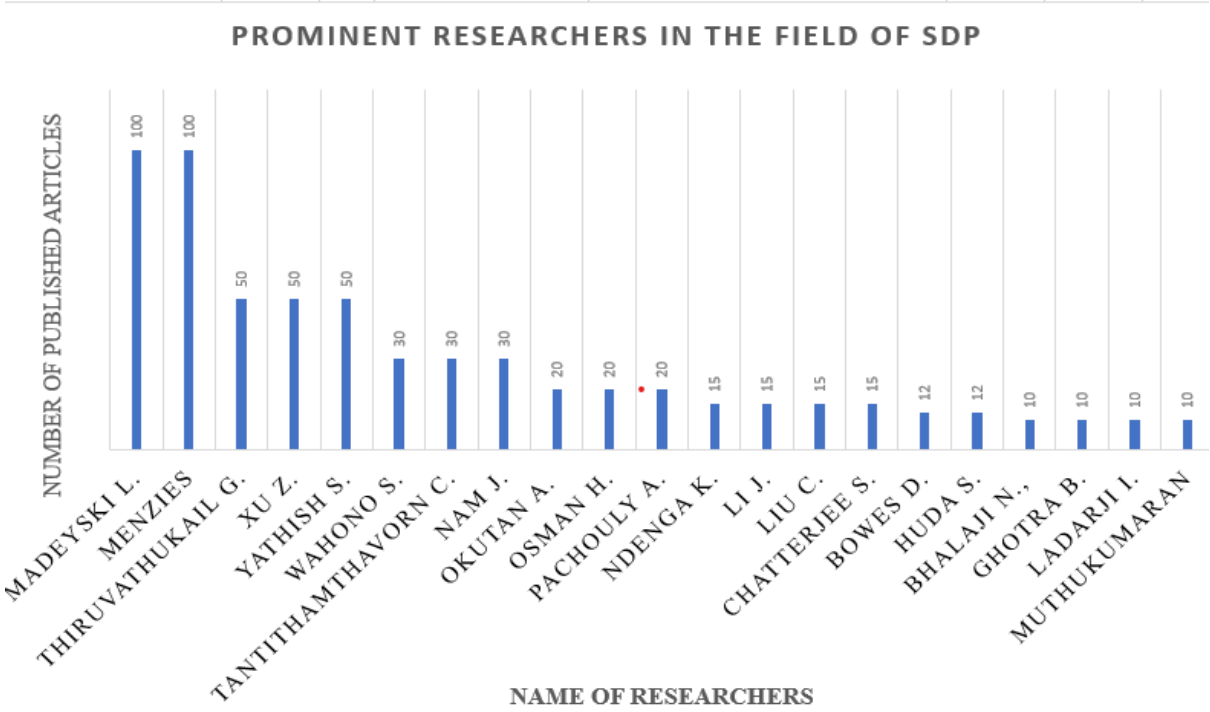

Figure 5: Prominent Researchers in Software Defect Prediction Field

Trending Area of research in Software Defect Prediction

In the past thirty years, there have been two widely accepted methods for identifying and resolving software issues (Pachouly et al., 2022). These two approaches are prediction-based and legacy-based. Up until recently, it was shown that software flaws may be found and fixed using a search-based method (Malhotra et al., 2023).

From the studied primary data, it has shown that there were 82 published articles on software defect prediction that applied classification techniques which accumulates to 78%

of the total study from 2017-2023. Meanwhile, Estimation algorithm has 3% of the published articles on software defect prediction from 2017-2023. Association algorithm has 2% within the said period. Clustering algorithm has 4 total published articles within 2017-2023 from our studied articles amounting to 4%. Finally, Data Processing has 14 published articles resulting in 13% of the total studied researches from 2017-2023 for software defect prediction. Figure 7 below shows the distribution of the studied primary researches from 2017-2023.

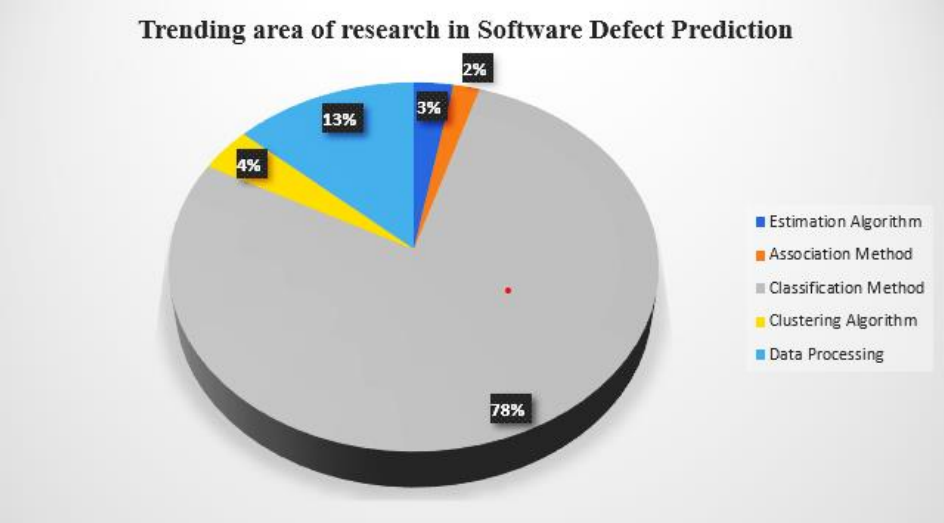

Figure 6: Trending area of research in Software Defect Prediction

The distribution of datasets used in software defect prediction over the range of 2017-2023 were found to be 21 published articles utilizes private datasets resulting of about 40%. Public datasets have a total of 32 published articles on software defect prediction within the time frame that results in 60%. Figure 8 below shows the distribution of datasets used from our primary studied articles within 2017-2023 on software defect prediction.

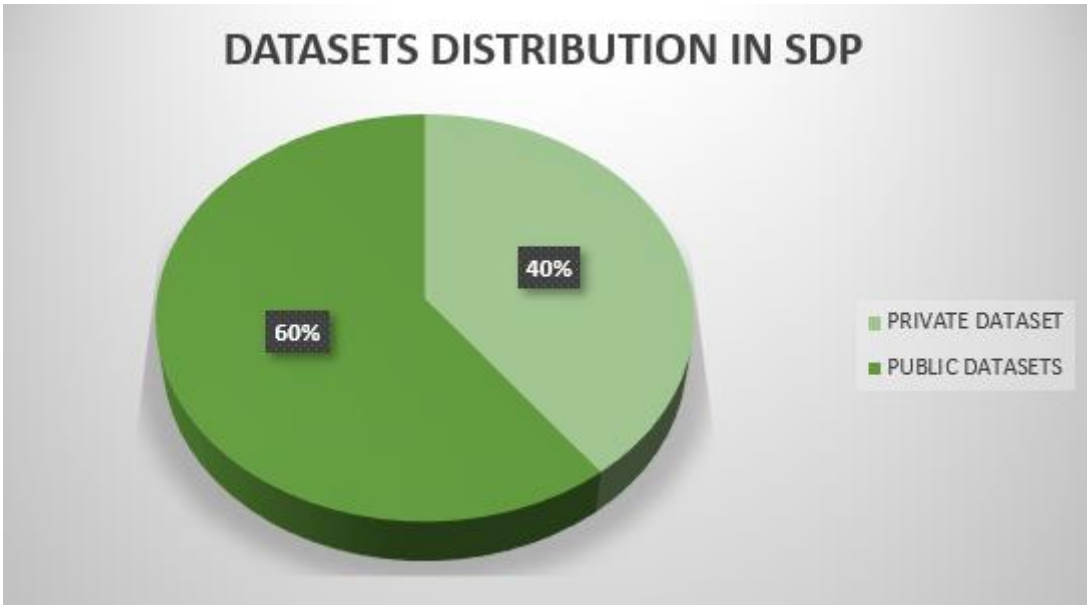

Figure 7: Distribution of applied datasets in Software Defect Prediction

The variation over years of datasets types applied in software defect prediction has shown greater interest in applying open-source datasets over its counterpart proprietary datasets. It has reported by Chatterjee *et al.* (2021) to develop a very good software defect prediction model will depend heavily on the quality of datasets used.

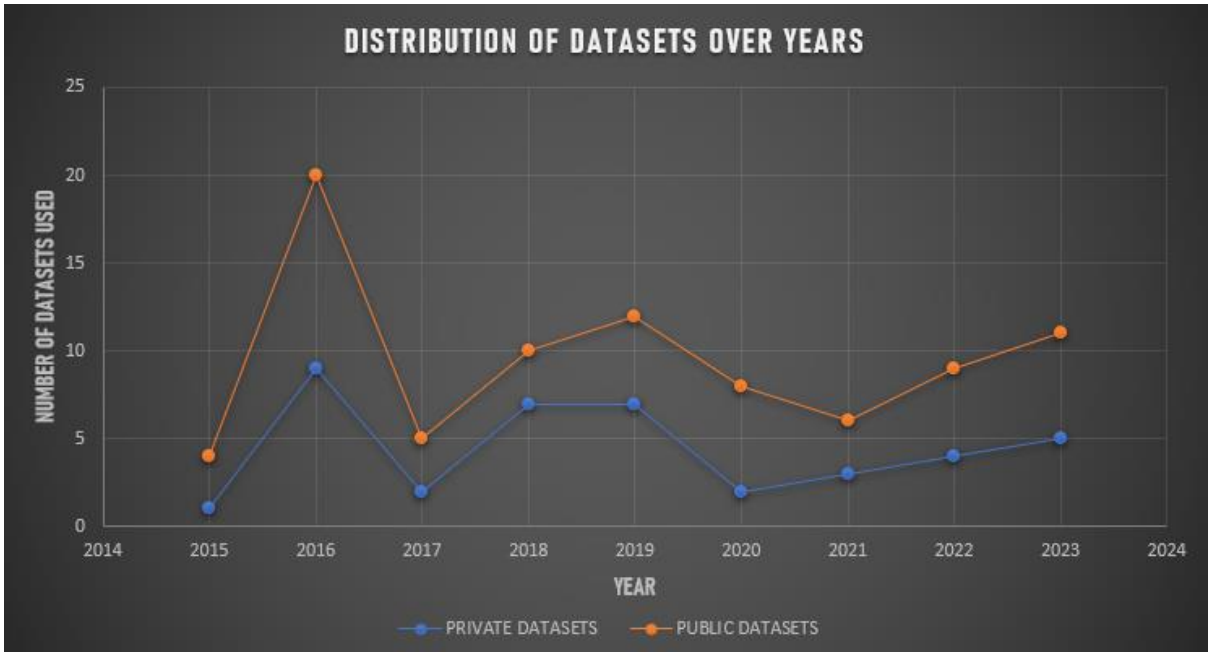

Figure 8: Distribution of Private and Public Datasets over years

Commonly Used Algorithms in Software Defect prediction

From the findings conducted by this literature review, there nine (9) commonly deployed methods in software defect prediction from 2017-2023 out of those mentioned in section 3.5. The methods are:

i. Support Vector Machines (SVMs)
ii. DECISION TREES (DT)
iii. NAÏVE BAYES
iv. LOGISTIC REGRESSION (LR)
v. K-Nearest Neighbours (K-NN)
vi. Random Forest (RF)
vii. Neural Networks (NN)
viii. Ensemble Methods (EM)
ix. Feature Selection Technique (FST)

Figure 9 presents the most commonly deployed method in software defect prediction from 2017-2023 based on our primary studies.

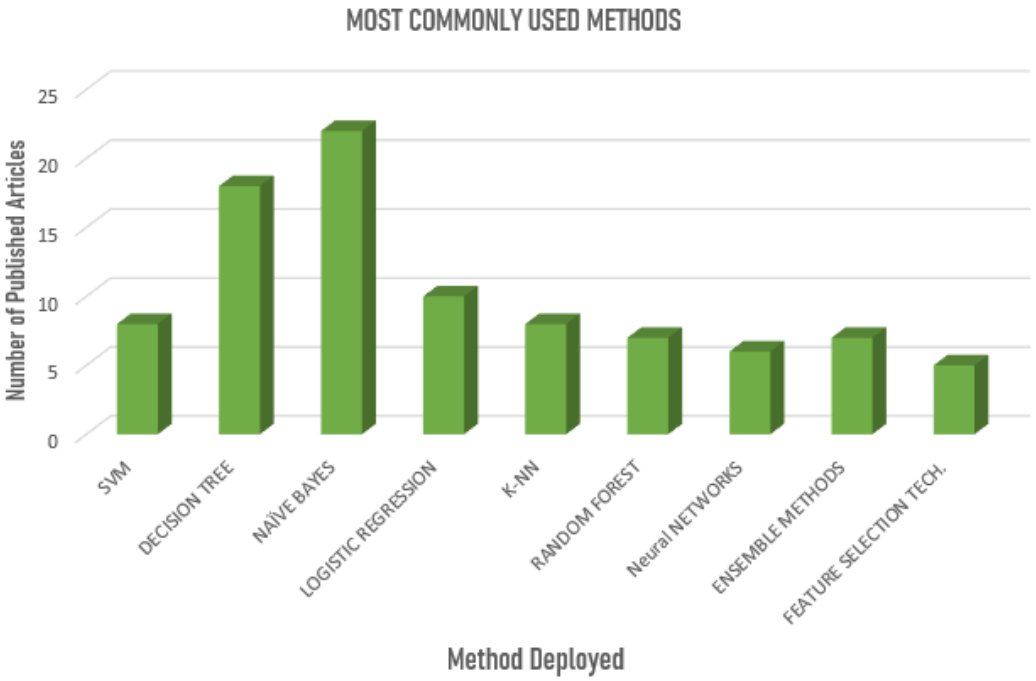

Figure 9: Most Commonly Deployed Methods

Figure 10 shows that Naïve Bayes dominates the most deployed method with 24%, followed Decision Tree with 20%.

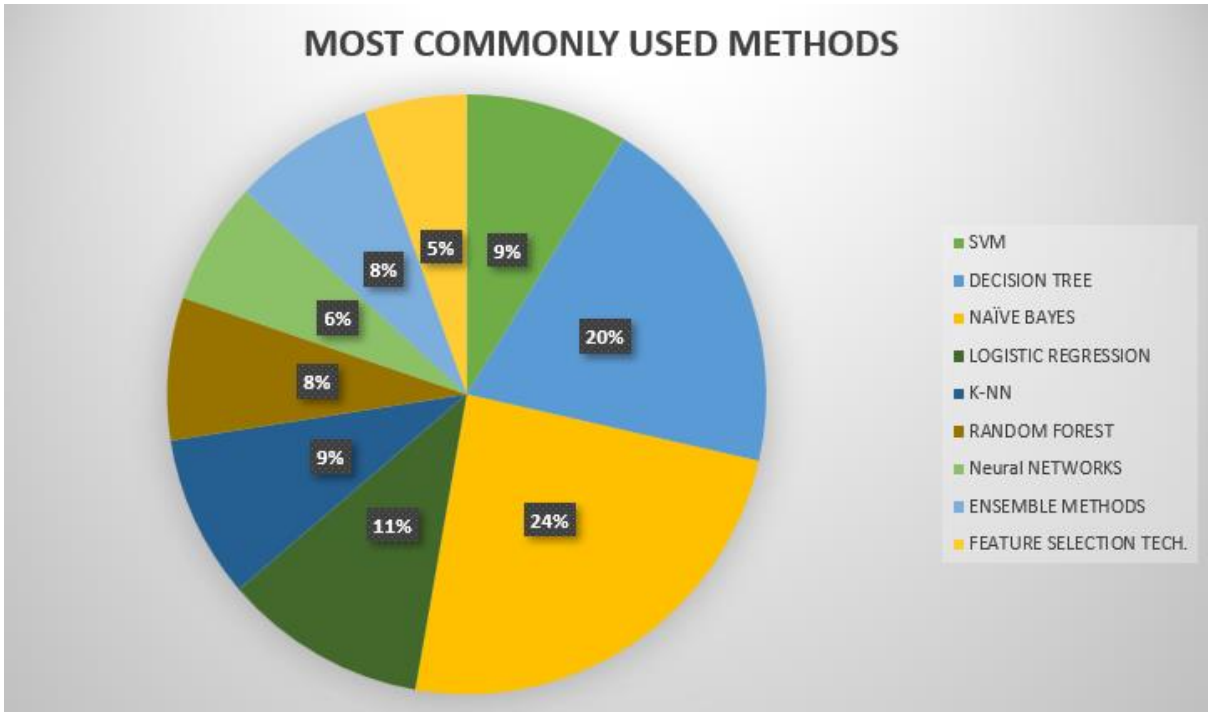

Figure 10: Distribution of most commonly deployed methods in SDP

Figure 11 presents the most commonly used software metrics in software defect prediction.

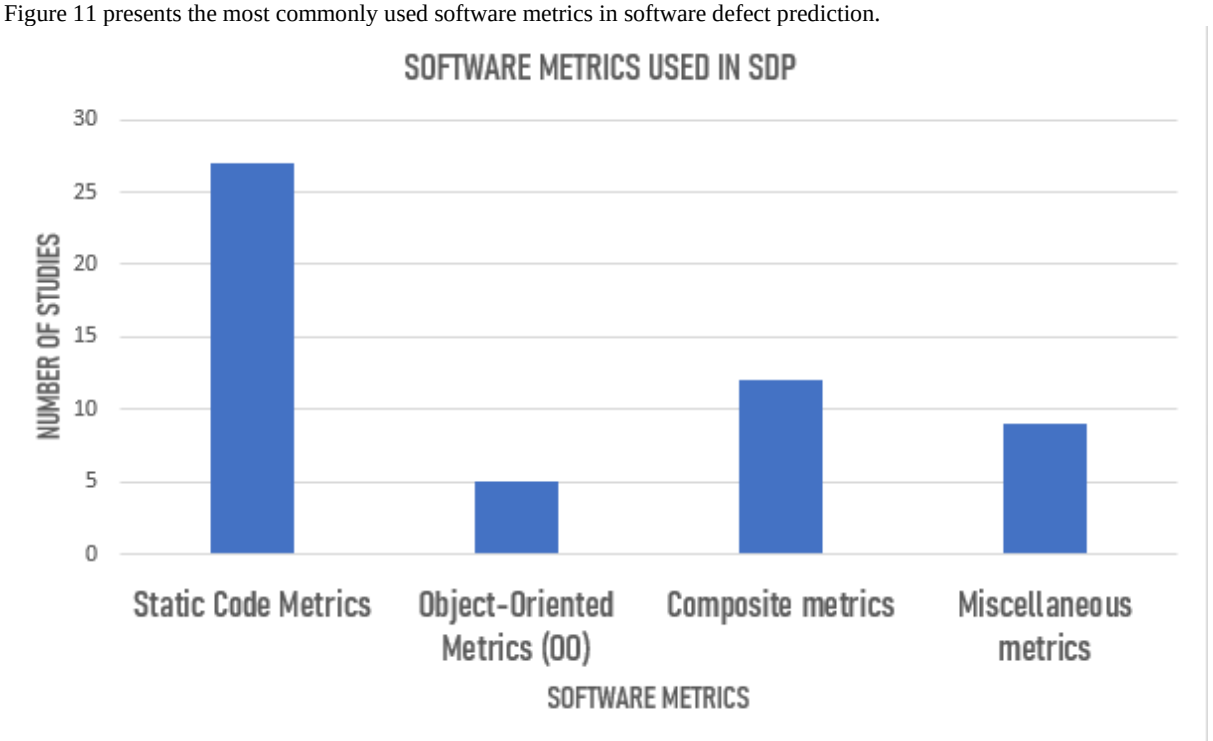

Figure 11: Commonly used software metrics in software defect prediction

Figure 12 shows the distribution of software metrics utilized from 2017-2023. Static code metrics constitutes of 51% of the metrics used in a primary study, while Object-Orientated metrics produced 9% of the metrics. Composite Metrics generated 23% and Miscellaneous Metrics produced 17% of the metrics within the year of review.

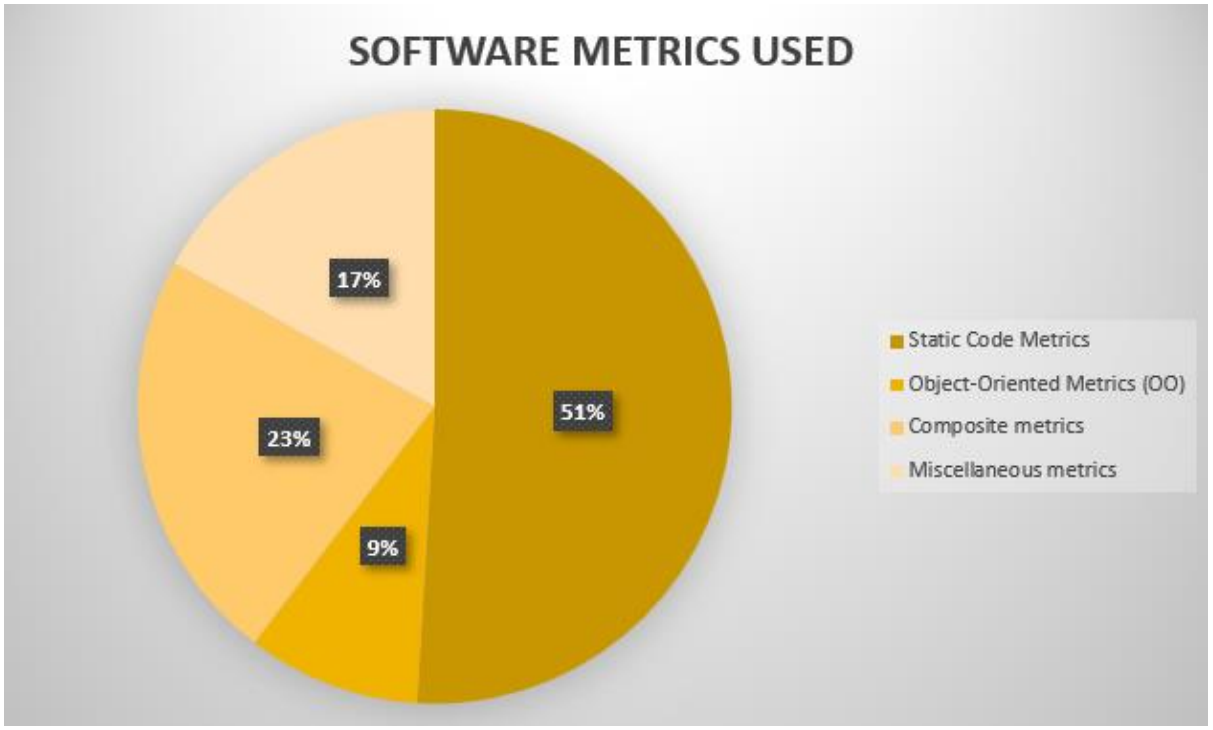

Figure 12: Distribution of Software Metrics used in SDP

Figure 13 presents the commonly applied filter-based feature selection techniques with their corresponding number of studies during the year under review.

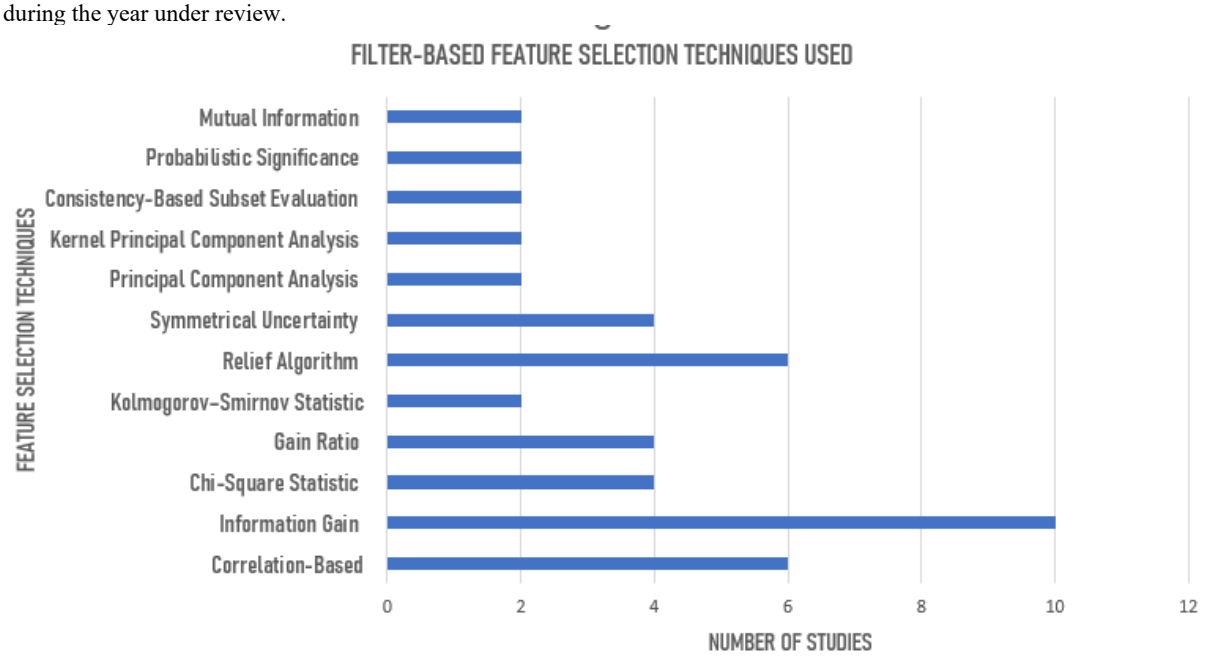

Figure 13: Commonly used filter-based feature selection techniques

Figure 14 presents the distribution of filter-based feature selection techniques utilized from our primary studies within 2017-2023. It has shown the information-gained feature selection techniques constitutes of 22% of the primary studies followed by correlation-based of 13%.

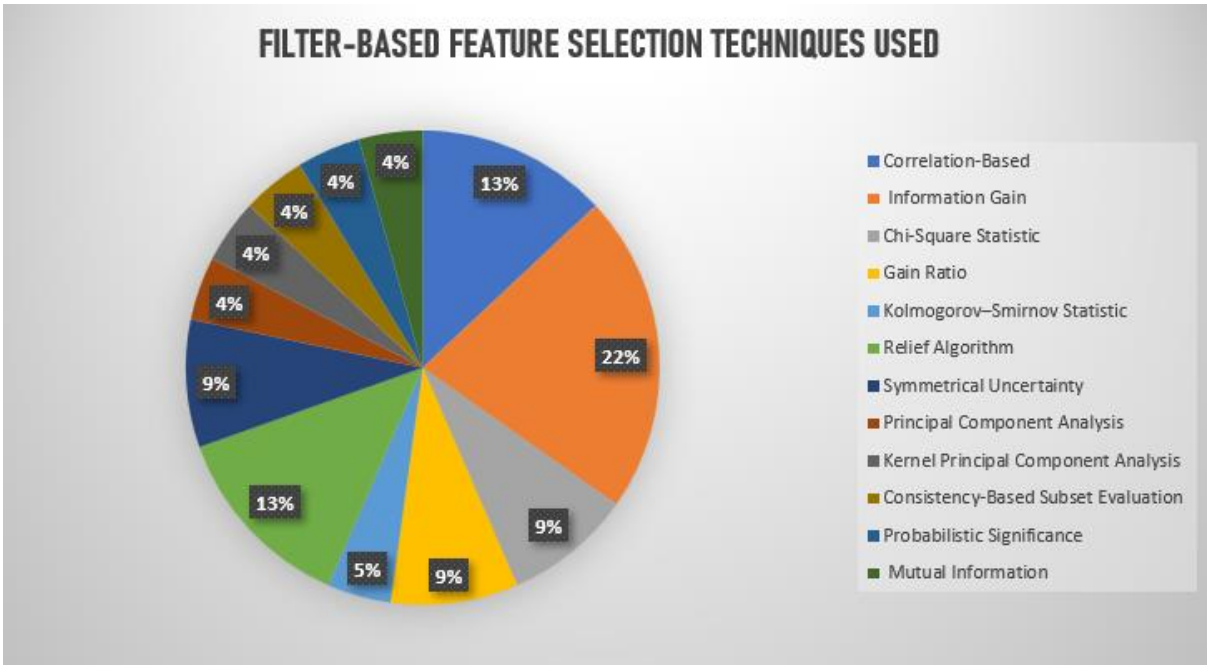

Figure 14: Distribution of Filter-Based Feature Selection Techniques used in SDP

Figure 15 presents the commonly used wrapper-based feature selection techniques with respect to the number of studies from our primary studies.

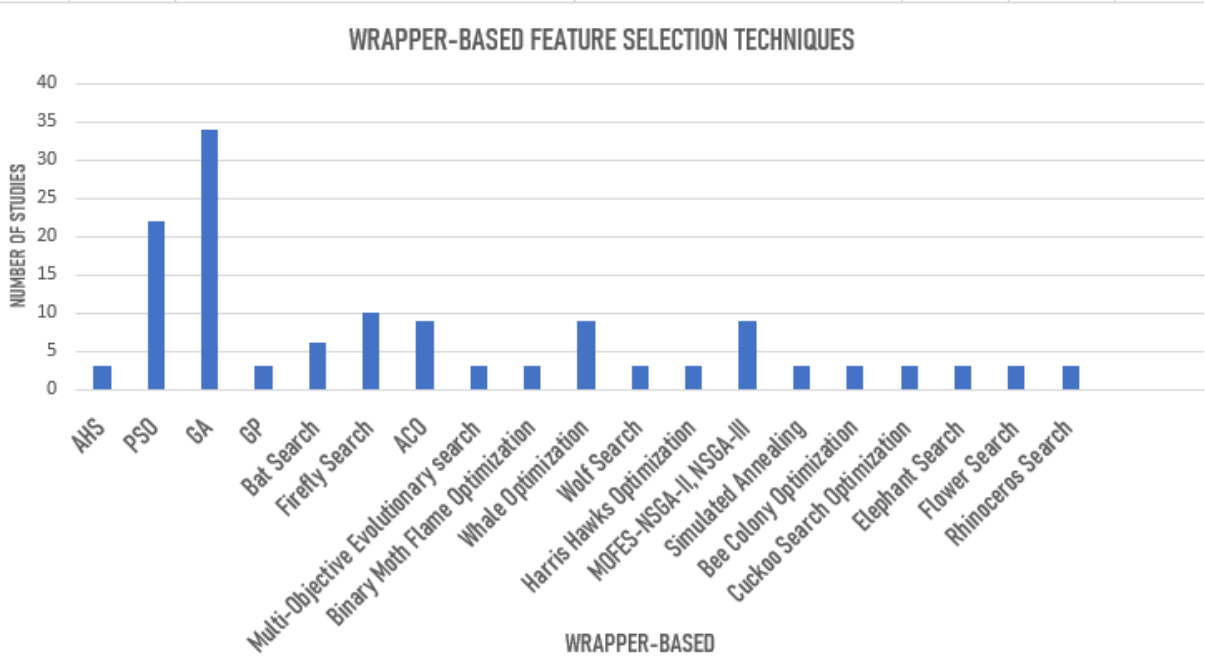

Figure 15: Commonly used Wrapper-based Feature Selection Techniques

Figure 16 shows the distribution of wrapper-based feature selection techniques used in software defect prediction with genetic algorithms having 25% followed by Particle swamp algorithms with 16% of the number of published articles within the time of the review.

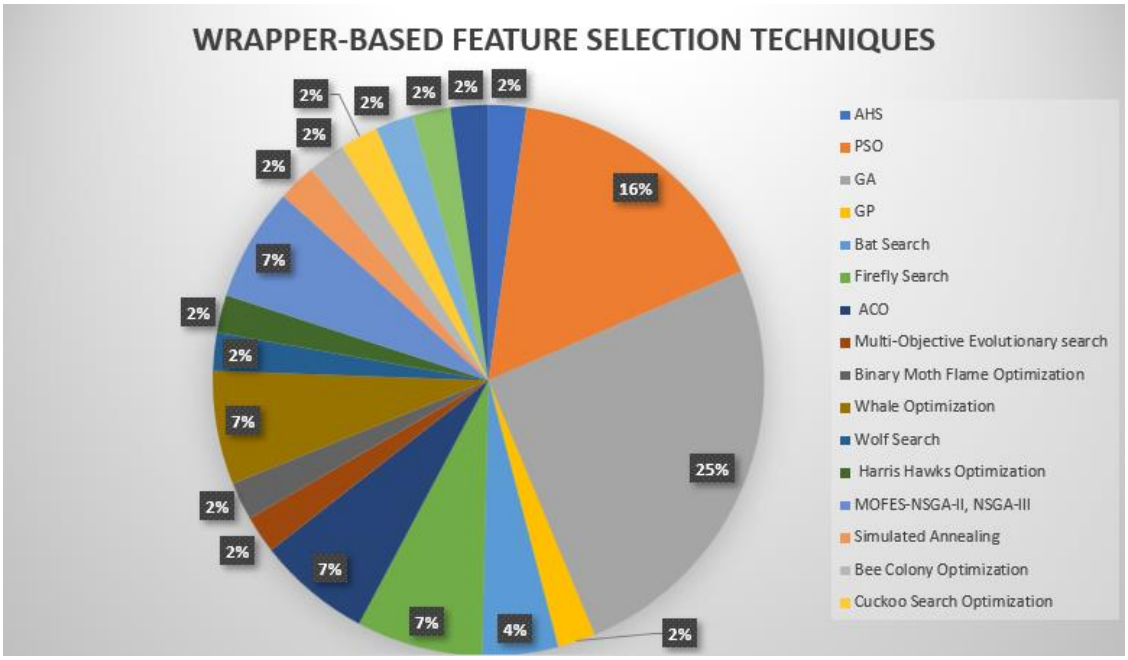

Figure 16: Distribution of Wrapper-based feature selection techniques

CONCLUSION

The goal of this literature review is to evaluate and examine the most current datasets, algorithms and methods applied in software defect prediction researches from 2017 to 2023. This was done based on the criteria led for inclusion and rejection which has yielded to come up with 104 published articles on software defect prediction from 2017 to 2023. Based on the analysis conducted on our primary selected researches disclose that the most trended areas of research in software defect prediction are data Pre-processing, classification methods, estimating the number of defects, association methods and clustering methods. From our studied primary data, it has shown that there were 82 published articles on software defect prediction that applied classification techniques which accumulates to 78% of the total study. Meanwhile, Estimation algorithm has 3% of the published articles on software defect prediction. Association methods has 2%. Clustering methods has 4 total published articles with 4%. Finally, Data Processing has 14 published articles resulting to 13%. The distribution of datasets used in software defect prediction found that 21 published articles utilized private datasets resulting of about 40%. Public datasets have a total of 32 published articles on software defect prediction which resulted in 60%. From the findings conducted by this literature review, there nine (9) commonly deployed algorithms in software defect prediction. The methods are: Support Vector Machines (SVMs), DECISION TREES (DT), NAÏVE BAYES (NB), LOGISTIC REGRESSION (LR), K-Nearest Neighbours (K-NN), Random Forest (RF), Neural Networks (NN), Ensemble Methods (EM) and Feature Selection Technique (FST). Many techniques have been proposed by various researchers to improve the machine learning with different hybrid models. Yet there are more gaps to fill in terms issues relating to computational complexity such as CPU time required to perform the prediction, the manual selection of correlated metrics which was predominantly used, issues of dimensionality reduction.

REFERENCES

Abubakar, S, M., Sufyanu Z. and Miyim, A., M. (2019). Proposed Defect Modelling for Mitigating Correlated Software Metrics. Dutse Journal of Pure and Applied Sciences. Vol. 5 (2b): 76-86.

Abubakar, S., M., Sufyanu Z., and Garko A., B. (2021). Impact of Correlated Software Metrics on Embedded Feature Selection Techniques. International Journal of Information Processing and Communication (IJIPC) Vol. 11 (2) 118-134

Arisholm, E., L. C. Briand, and E. B. Johannessen, (2010) "A Systematic and Comprehensive Investigation of Methods to Build and Evaluate Fault Prediction Models," Journal of Systems and Software, vol. 83, no. 1, pp. 2–17

Batool, S., & Khan, S. U. (2016). A systematic review of software defect prediction using deep learning techniques. Journal of Systems and Software, 118, 86-107.

Bowes, D., Tracy, H., and Jean, P. (2019). Software defect prediction: do Different Classifiers find the same defects. This article is published with open access at Springerlink.com.

Bunescu R, Ruifang G, Rohit JK, Marcotte EM, Mooney RJ, Ramani AK, Wong YW (2005) Comparative experiments on learning information extractors for proteins and their interactions. Artif Intell Med (special issue on Summarization and Information Extraction from Medical Documents) 2:139–155

Cai X, Niu Y, Geng S et al (2020) An under-sampled software defect prediction method based on hybrid multi-objective cuckoo search. Concurr Comput. https://doi.org/10.1002/cpe.5478

Cawley G, Talbot, C., and Girolami, M. (2007). Sparse Multinomial Logistic Regression via Bayesian L1 Regularization. In: B. Schölkopf, J. C. Platt, and T. Hoffmann (Eds.). Advances in Neural Information Processing Systems, MIT Press, 209–216, 2007.

Chambers, J. M. (1992.) "Statistical Models in S. Wadsworth," Pacific Grove, California

Chatterjee, S., and Maji B. (2021). A Mahalanobis Distance based Algorithm for assigning rank to the predicted fault prone software modules. IEEE transaction of science.

Cukic, B., & Singh, H. (2004). Robust Prediction of Fault-Proneness by Random Forests. *15th International Symposium on Software Reliability Engineering*, 417–428. http://doi.org/10.1109/ISSRE.2004.35

Dam, H.K., Tran, T., Pham, T.T.M., Ng, S.W., Grundy, J., Ghose, A.: (2018). Automatic feature learning for predicting vulnerable software components. IEEE Transactions on Software Engineering

Das, S. (2019). Filters, Wrappers and Boosting-based Hybrid for Feature Selection. Division of Engineering and Applied Science Harvard University, Cambridge, MA 02B8, USA.

Dash, M., H. Liu, and H. Motoda, (2000) "Consistency based Feature Selection," in Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 98–109.

Fan, Guisheng, et al. (2019). Software defect prediction via attention-based recurrent neural network. Sci. Program. 2019.

Fenton, N. E., & Neil, M. (1999). A critique of software defect prediction models. *IEEE Transactions on Software Engineering*, *25*(5), 675–689. http://doi.org/10.1109/32.815326

Ghotra, B., S. Mcintosh, and A. E. Hassan, (2020) "A large-scale study of the impact of feature selection techniques on defect classification models," in Proceedings of the 14th International Conference on Mining Software Repositories. IEEE Press, pp. 146–157.

Guyon I. and A. Elisseeff, (2003) "An Introduction to Variable and Feature Selection," Journal of Machine Learning Research, vol. 3, pp. 1157– 1182

Gyimothy, T., Ferenc, R., Siket, I. (2005): Empirical validation of object-oriented metrics on Open-source software for fault prediction. IEEE Transactions on Software engineering 31(10), 897–910

Hall, T., Beecham, S. D., Bowes, Gray, D., and Counsell, S. (1999). A Systematic Literature Review on Fault Prediction Performance in Software Engineering. Transactions on Software Engineering, 38(6):1276–1304.

Halstead M., H. (1977) Elements of software science. Elsevier Science Ltd., New York Hosseini S, Turhan B, Mantyl M (2017) Search based training data selection for cross project defect prediction. In: ACM international conference proceeding series. Association for Computing Machinery

Hosseini, S. M., Seker, C., & Torkar, G. (2019). A systematic literature review and meta-analysis of cross-project defect prediction studies. Empirical Software Engineering, 21(3), 1027-1062.

Hoque, N., Mihir, S. and Dhruba, K. B. (2018). EFS-MI: an Ensemble Feature Selection Method for Classification. Complex & Intelligent Systems ISSN: 2199-4536.

Jia L. (2019). A Hybrid Feature Selection Method for Software Defect Prediction. IOP Conf. Series: Materials Science and Engineering 394 032035 https://doi.org/10.1088/1757-899X/394/3/032035

Jiang Y, Cukic B, Yan M (2008) Techniques for evaluating fault prediction models. Empir Softw Eng J 13(5):561–595

Jiarpakdee, J., S., Y., Thongtanunam, P., and Tantithamthavorn, C. (2019). Mining Software Defects: Should We Consider Affected Releases? In Proceedings of the International Conference on Software Engineering (ICSE).

Kalsoom, A., Muazzam, M., and Mustansas, A. S. (2021). A Dimensionality reduction- based efficient software fault production using fisher linear discriminant analysis. http://doi.org/10.1007/s11227-018-2326-5.

Kohavi, R. and G. H. John, (1997) "Wrappers for Feature Subset Selection," Artificial Intelligence, vol. 97, no. 1-2, pp. 273–324,

KubatM, Holte RC, Matwin S (1998). Machine learning for the detection of oil spills in satellite radar images. Mach Learn J 30(2–3):195–215

Lal, T., Chapelle, O., Jason, W., and AndrÃl', E. (2006). Embedded Methods in Isabelle Guyon, Masoud Nikravesh, and Steve Gunn, and Lotfi Zadeh, editors, Feature Extraction, volume 207 of Studies in Fuzziness and Soft Computing, 137–165. Springer Berlin / Heidelberg.

Li, L., Lessmann, S., Baesens, B. (2020): Evaluating software defect prediction performance: an updated benchmarking studies. arXiv preprint arXiv:1901.01726

Lewis D, Gale WA. (1994). A sequential algorithm for training text classifiers. In: Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '94, New York, NY, USA. Springer, New York, pp 3–12

Lu, H., E., Kocaguneli, and B. Cukic, (2019) "Defect Prediction between Software Versions with Active Learning and Dimensionality Reduction," in Proceedings of the International Symposium on Software Reliability Engineering (ISSRE), pp. 312–322.

Malhotra R, Chawla, S.Sharma, A. (2023). Software defect prediction using hybrid techniques: a systematic literature review APPLICATION OF SOFT COMPUTING 27:8255–8288 https://doi.org/10.1007/s00500-022-07738-w

Malhotra R, Khanna M, Raje RR (2017) On the application of search-based techniques for software engineering predictive modeling: a systematic review and future directions. Swarm Evol Comput 32:85–109. https://doi.org/10.1016/j.swevo.2016.10.002

Malhotra, R.: (2014). Comparative analysis of statistical and machine learning methods for predicting faulty modules. Applied Soft Computing 21, 286–297

McCabe TJ, Butler CW (1989) Design complexity measurement and testing. Commun ACM 32:1415–1425. https://doi.org/10.1145/76380.76382

Mccabe TJ (1976) A complexity measure. IEEE Trans Softw Eng SE 2:308–320. https://doi.org/10.1109/TSE.1976.233837

McHugh, M., L. (2013) "The Chi-square Test of Independence," Biochemia Medica, vol. 23, no. 2, pp. 143–149

Menzies, T. (2018). "The Unreasonable Effectiveness of Software Analytics," IEEE Software, vol. 35, no. 2, pp. 96–98, March 2018.

Menzies T, Greenwald J, Frank A (2007) Data mining static code attributes to learn defect predictors. IEEE Trans Softw Eng 33(1):2–13

Menzies T, Stefano J, Ammar K, McGill K, Callis P, Davis J, Chapman. (2003). When can we test less? In: Proceedings of 9th international software metrics symposium, pp 98–110

Mitchell, T. M., "Machine Learning," McGraw Hill, 1997.

Ndega, K. M., Ivayto, G., and Mehat, F. (2019). Performance and cost-effectiveness of change burst metrics in predicting software fault. Springer –verlaglando ltd, post of springer Nature.

Olson D (2008) Advanced data mining techniques. Springer, Berlin

Pachouly, A., Sahin, Y., & Aleti, S. (2016). A systematic mapping study on the use of machine learning and deep learning for software defect prediction. Information and Software Technology, 80, 120-144.

Pandey, S. K., Mishra, R.B., Tripathi A., K. (2021). Machine learning based methods for software fault prediction: A survey. Expert Systems with Applications 172, 114595

Qiu, Shaojian, *et al.*(2019). Transfer convolutional neural network for cross-project defect prediction. Appl. Sci. 9 (13), 2660.

Radjenovic D, Hericko M, Torkar R, Zivkovic A (2013) Software fault prediction metrics: a systematic literature review. Inf Softw Technol 55(8):1397–1418

Raukas, H. (2017). Some Approaches for Software Defect Prediction. PhD. Dissertation UNIVERSITY OF TARTU Institute of Computer Science Computer Science Curriculum.

Rathore SS, Kumar S (2019) A study on software fault prediction techniques. Artif Intell Rev 51:255–327. https://doi.org/10.1007/S10462-017-9563-5/FIGURES/8

Sandri, M., and Zuccolotto, P. (2006). Variable Selection Using Random Forests. In Zani, S. Zani, A., Riani, M. and Vichi (eds.), M. Data Analysis, Classification and the Forward Search, Studies in Classification, Data Analysis, and Knowledge Organization, Springer, 263–270.

Sharma, G. S. Sharma, and S. Gujral, (2023). A Novel Way of Assessing Software Bug Severity Using Dictionary of Critical Terms," *Procedia Comput. Sci.*, vol. 70, pp. 632–639, 2015, https://doi.org/10.1016/j.procs.10.059.

Shepperd, M., Song, Q., Sun, Z., and Mair, C. (2022). Data Quality: Some Comments On the NASA Software Defect Datasets. Transactions on Software Engineering (TSE), 39(9):1208–1215.

Son LH, Pritam N, Khari M et al (2019) Empirical study of software defect prediction: a systematic mapping. Symmetry 11:212. https://doi.org/10.3390/SYM11020212

Turabieh H, Mafarja M, Li X (2019) Iterated feature selection algorithms with layered recurrent neural network for software fault prediction. Expert Syst Appl 122:27–42. https://doi.org/10.1016/j.eswa.2018.12.033

Wang, Y., & Hassan, A. E. (2016). A systematic review of software defect prediction using machine learning techniques. Information and Software Technology, 76, 209-235.

Xu, Z. J. Liu, Z. Yang, G. An, and X. Jia, (2020) "The Impact of Feature Selection on Defect Prediction Performance: An Empirical Comparison," in Proceedings of the International Symposium on Software Reliability Engineering (ISSRE), 2020, pp. 309–320

Yang X, Yu H, Fan G, Yang K (2020) A differential evolution-based approach for effort-aware just-in-Time software defect prediction. In: Proceedings of the 1st ACM SIGSOFT international workshop on representation learning for software engineering and program languages. Association for Computing Machinery, Inc, pp 13–16

Yang, X., Lo, D., Xia, X., Zhang, Y., Sun, J.: Deep learning for just-in-time defect prediction. In: 2015 IEEE International Conference on Software Quality, Reliability and Security. pp.17–26. IEEE (2020)

Yatish, Suraj, et al., 2019. Mining software defects: should we consider affected releases? In: 2019 IEEE/ACM 41st International Conference on Software Engineering. ICSE, IEEE.

Yousef W, Wagner R, Loew M., (2004). Comparison of non-parametric methods for assessing classifier performance in terms of roc parameters. In: Proceedings of international symposium on information theory, 2004. ISIT 2004, pp 190–195

Youden WJ (1950) Index for rating diagnostic tests. Cancer 3(1):32–35

Zhang, H., & Xie, T. (2020). A novel software defect prediction model based on deep belief networks and ensemble learning. Information and Software Technology, 76, 130-144.

Zou, H., and Hastie, T. (2005). Regularization and variable selection via the elastic Net Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(2):301–320.