A Hybrid Multi-Filter Wrapper Feature Selection Method for Software Defect Predictors

Balogun Abdullateef Oluwagbemiga[#,*1], Basri Shuib[#2], Said Jadid Abdulkadir[#3], Ahmad Sobri Hashim[#4]

[#] *Department of Computer and Information Sciences & Software Quality and Quality Engineering (SQE) Research Cluster , Universiti Teknologi PETRONAS,*

32610 Seri Iskandar, Perak, Malaysia.

[1]abdullateef_16005851@utp.edu.my

[2]shuib_basri@utp.edu.my

[3]saidjadid.a@utp.edu.my

[4]sobri.hashim@utp.edu.my

[*]*Department of Computer Science, University of Ilorin, PMB 1515, Ilorin, Nigeria*

[1] balogun.ao1@unilorin.edu.ng

***Abstract*— Software Defect Prediction (SDP) is an approach used for identifying defect-prone software modules or components. It helps software engineer to optimally, allocate limited resources to defective software modules or components in the testing or maintenance phases of software development life cycle (SDLC). Nonetheless, the predictive performance of SDP models reckons largely on the quality of dataset utilized for training the predictive models. The high dimensionality of software metric features has been noted as a data quality problem which negatively affects the predictive performance of SDP models. Feature Selection (FS) is a well-known method for solving high dimensionality problem and can be divided into filter-based and wrapper-based methods. Filter-based FS has low computational cost, but the predictive performance of its classification algorithm on the filtered data cannot be guaranteed. On the contrary, wrapper-based FS have good predictive performance but with high computational cost and lack of generalizability. Therefore, this study proposes a hybrid multi-filter wrapper method for feature selection of relevant and irredundant features in software defect prediction. The proposed hybrid feature selection will be developed to take advantage of filter-filter and filter-wrapper relationships to give optimal feature subsets, reduce its evaluation cycle and subsequently improve SDP models overall predictive performance in terms of Accuracy, Precision and Recall values.**

***Keywords*—** *Data Quality Problem, Feature Selection, High Dimensionality, Software Defect Prediction,*

International Journal of Supply Chain Management
IJSCM, ISSN: 2050-7399 (Online), 2051-3771 (Print)

1. Introduction

Software Defect Prediction (SDP) is an approach used for identifying defect-prone software modules or components. It helps software engineers to optimally allocate limited resources to defective software modules or components in the testing or maintenance phases of SDLC [1, 2]. This will, in turn, helps to assess software quality and also monitor software quality assurance [3, 4]. SDP models make use of information such as software source code complexity, developer's information, and development history to predict software modules or component that may be defective[5, 6]. This information is quantified using software metrics to determine the level of software quality and reliability.

As such, each component or module of the software is depicted by a set of metric values and a label. The label indicates the defective state of software components/modules and the derived scale values are used to build predictive models [7, 8]. In other words, SDP uses data derived from software via software metrics to determine the quality and reliability of software modules or components. SDP predictive results can, therefore, be utilized by software engineers in addressing modules or components that may be defective, the number of defects in a module, or any other details cognate to software defects afore software testing[7, 8]. Machine learning methods are the most mundane and widely used techniques for SDP and successes have been recorded in its application in software engineering for SDP [5, 9, 10]. This has increasingly

attracted the attention of researchers to the field to develop more predictive models with high accuracy[9, 11]. The predictive performance and effectiveness of SDP models depend on the quality of data used for training predictive models and the choice of the predictive model[6, 10]. Recent studies have revealed that software defect data suffer from data quality issues such as the presence of noisy data instances, high dimensionality and class imbalance [5, 9, 10].

High dimensionality in datasets has been seen as an issue with software defect prediction. This is primarily caused by using many software metrics for evaluating the quality of software [10, 12]. These software metrics generate large software features which can be used to characterize software quality. However, these features are often redundant and some are usually irrelevant which only add to the computational overhead cost and also hamper the performance of predictive models in SDP [3, 13]. A plausible approach to address large software features is to deploy feature selection on such data. Feature selection will identify a subset of features that are relevant, non-redundant and provide the best predictive result [10, 12, 14].

As a general data pre-processing step in data mining tasks, feature selection selects subsets of irredundant and relevant features for predictive processes. It aims at identifying dataset features that are relevant and eliminating features that are redundant [15]. Studies have shown that failure to remove irrelevant and redundant features from SDP datasets can affect the performance of predictive models used on such dataset[10, 15, 16]. In addition, a subset can consist of features that are correlated with the class label and uncorrelated with each other. Thus, there is a need for efficient feature selection method which is able to optimally select relevant and irredundant features as possible and also leading to a good predictive performance [17, 18]. This paper therefore proposes a hybrid multi-filter wrapper feature selection method for software defect predictors.

The rest of this paper is organized as follows: Section II presents a detailed literature review and related studies done by other researchers. Section III describes the methodology which explains the proposed hybrid feature selection method. Section IV gives the conclusion of the study.

.

2. Literature Review

In this section, related literatures and existing studies are categorized and discussed as follows:

2.1 Software Defect Prediction

The reliance on Information technology (IT) cuts across all human endeavors and this makes software systems development imperative. Due to the continuous increase in the number of software users, modern software systems are inherently large and open to continuous upgrades. With the modern advancement in IT and software development as its focal point, there is a need for development and deployment of large software systems. Unfortunately, developing defect-free software systems in this context is difficult due to the increasing requirements based on a large number of users and other constraints. Presence of defects in software systems usually leads to a problem for both software users and enterprises [19, 20]. Therefore, predicting software modules defectiveness is imperative and could help address these problems. This would help software enterprises focus more on software modules or components that could be defective and consequently reducing maintenance cost and deploying better software products [21].

Software Defect Prediction (SDP) is an approach used for identifying defect-prone modules or components in a software system. It uses quality and reliability measurements of software systems for this purpose. Software metrics are used to determine the worth of software systems in terms of quality and reliability[1, 2]. Figure 1, gives a graphical representation of the SDP process.

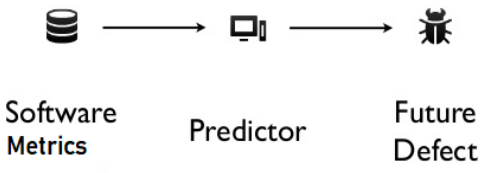

Figure 1. Software Defect Prediction Model (Huda, et al. [22])

2.2 Software Metrics

The prediction of software defects is based on the quantified values derived via software metrics [23].

Software metrics is a measurement tool in software engineering for determining the quality and reliability level of software system [24, 25]. Aside from measuring software quality, software metrics help determines software performance, software planning and so on. This makes it very critical in software engineering as it provides a medium for planning, organizing, control, and improvement of software systems or projects. Software metrics are of different types and be categorized as product, process and project metrics [25].

Product metrics are derived from the software system by measuring its characteristics in term of lines of code (LOC), McCabe metrics and so on. They are easy to understand and shows the depth of a software system[23]. Process metrics in its own case measures and models the software development process[1, 25]. It anchors on the software methodology used, development time, or details about the development team. A detailed process metric analysis could improve the software maintenance processes and its complementary costs[23]. While the project metrics bothers on the productivity of SDLC. The effort and resources used in each phase of the SDLC in the production of the software product[1, 25].

The proliferation of these software metrics in determining the quality of a software system leads to high dimensionality problem in metric values generated. The problem may eventually lead to a high computational time of processing the metrics, low SDP predictive model performance, or difficult understanding of knowledge derived from these metrics [26, 27]. Liebchen and Shepperd [28] and Ghotra, et al. [29] in their studies, advocated that quality of data in SDP must be given considerable attention as most inconsistent research results in SDP are mostly caused as a result of unclean data. Petrić, et al. [16], supported this claim stating that "the quality of the SDP data underpins the confidence that can be placed in the results of studies using such SDP data". Since the performance SDP predictive models are based on software metrics and data collection issues are ineluctable, data quality becomes of high interest. Shepperd, et al. [15] and Gray, et al. [30] proposed an outline for data pre-processing procedures to clean SDP datasets with conflicting values and removal of identical or inconsistent values. However, data quality issues such as high dimensionality, class imbalance, class overlap, and noisy data instances are still present in SDP and these undermine the performance and generalizability of SDP predictive models [10, 15, 16].

The high dimensionality problem can be addressed by feature selection as its goal is to select relevant and irredundant software metric values that are better than the whole software metric values [31].

2.3 Feature Selection

There are two famous special methods of dimensionality reduction which are the feature selection (FS) and feature extraction (FE) methods. Feature extraction method transforms features of data into new feature representation. That is, it reduces the datasets with high dimensional features to a lower dimensional datasets. While Feature selection method seeks to generate subsets from the original features of a dataset that models the original dataset. Re-presentation of features is not used in this case as the characteristics of the dataset and other measures are used for the feature selection process [31, 32].

In most studies, feature selection methods are used ahead of feature extraction since it retains the nature and characteristics of the datasets [14, 31]. The idea behind feature selection is to reduce the effect of tricky features in the dataset. Such features are regarded to as Irrelevant and redundant features. Irrelevant features are features that do not add to the improvement of predictive models' performances. The accuracy of a predictive model is how close a measured value is to the actual or true value. However, the predictive model may mistakenly include them in the model. Removal of these irredundant features from datasets lowers the dimensions and consequently reduces the computational time of the model [32, 33]. Redundant features are those that can replace or be replaced by other features in a dataset. This is defined based on the correlation between or among features in a dataset [32, 33]. According to Cai, et al. [31], a successful feature selection reduces the dimensionality of the feature space, speeds up and reduces the cost of a predictive model, and obtains the feature subset which is the most relevant to the predictive process.

Feature selection approaches can be divided into two types: filter feature selection and wrapper feature selection [32].

- Filter feature selection method uses the general properties of datasets to assess, rank and filter its features. Consequently, making filter feature selection methods absolute of any predictive technique [17]. It uses data properties such as the measure of spread, skewness, probability distribution and so on, to assess the dataset features without depending on any particular classification algorithm. This makes filter feature selection method fast, scalable, and low computational complexity [18]. Unfortunately, filter methods assess and measure features individually which means it does not consider interaction and dependencies of features [17, 32].

- Wrapper feature selection method in its own case measures the relevance of subsets by assessing its predictive performance. It based on the underlining classification processes. This makes it better than the filter method in terms of their respective influence on predictive models. However, they often lead to high computational complexity [18]. These are in line with its typical lack of generality since the resulting subset of features is tied to the bias of the underlining classification technique used as the evaluation function and choice of the search strategy used for optimal subset selection[32]. The feature subset generated will be more in line with the classification process. Hence, the bias and lack of generalizability of wrapper feature selection methods[9, 34].

In summary, wrapper methods measure the relevancy of a feature subset using a classification algorithm while filter methods depend on the properties of the dataset [18]. This makes filter method simple and less computational complexity, but the accuracy of classification algorithm on such filtered data cannot be guaranteed. While wrapper methods with good predictive performance but with high computational complexity and lack of generalizability [9, 34].

Finding a way to hybridize filter and wrapper methods into a single process to utilize their respective advantages while avoiding their disadvantages are expected to improve the performance of predictive models [32]. The hybrid feature selection method attempts to utilize the advantages of both filter and wrapper methods by combining their complementary strengths [9, 34]. It uses different evaluation criteria in different search stages to improve the efficiency and prediction performance with better computational performance. However, most of the studies which hybridize wrapper with filter methods are usually based on wrapper methods which inherit most of the wrapper feature selection problems [14, 31, 32].

Based on these reasons, this study proposes a hybrid multi-filter wrapper feature selection method for software defect prediction. The proposed feature selection will be developed to take advantage of filter-filter and filter-wrapper relationships to give optimal feature subsets with high predictive performance and also to reduce its evaluation cycle and subsequently improve performance with low computational cost.

3. Methodology

In general, this study proposes a hybrid multi filter-wrapper feature selection method. A hybrid method capable of selecting relevant and irredundant features from SDP dataset and which improves the performance of SDP predictive models with a generalizable result. The proposed hybrid feature selection is divided into 3 stages as shown in Figure 2

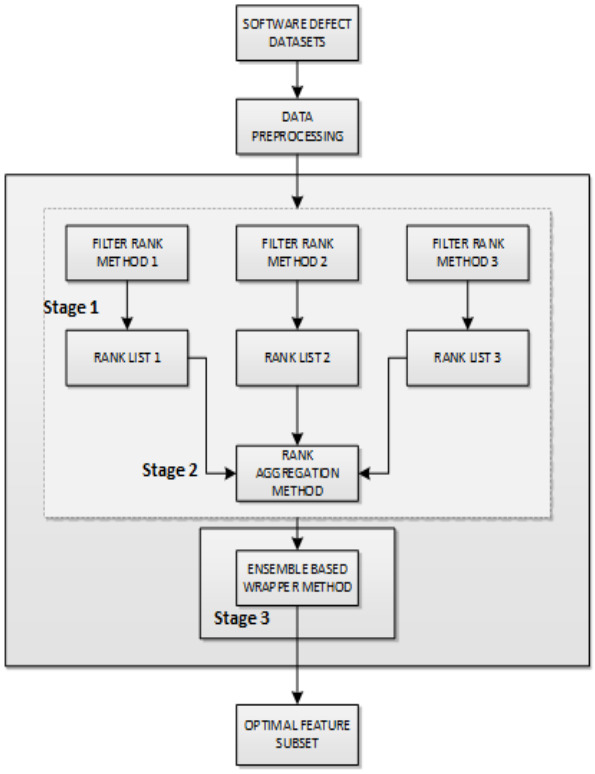

Figure 2. Proposed Hybrid Multi-Filter Wrapper Feature Selection Method for Software Defect Predictors

3.1 Stage 1:Multi-Filter Feature Selection Method

This stage consists of multi-filter feature selection methods. These filter methods will be selected based on different computational characteristics such as probability and statistical functions to introduce diversity and increase the regularity of feature selection process [10]. Each of the filter methods will be used to rank features from the software defect dataset. The essence of this stage is to resolve the selection problem of filter methods by using multiple filter methods of diverse computational characteristics to generate feature rank list of software defect datasets.

3.2 Stage 2:Rank Aggregation Method

Thereafter, individual rank lists from Stage 1 will be aggregated using a rank aggregation method. The aim of the rank aggregation method is to find the best rank list which would be closest as possible to individual ordered lists. The rank aggregation process will produce a more stable (non-disjoint) and complete feature rank list better than individual filter-based feature selection methods.

3.3 Stage 3: An Ensemble based Wrapper Method

In this stage, the aggregated rank list produced from Stage 2 will be further processed using an ensemble based wrapper method. This stage involves the deployment of search strategy into the feature selection process of the ensemble based wrapper method by using a meta-heuristic search strategy. This will lessen the wrapper evaluation cycle but without decreasing the predictive performance of the subsets obtained. Lastly, the subset feature list will be evaluated by an ensemble model. The choice of an ensemble of classifiers against single classifier method is to resolve the bias issue with using single classifier and the lack of generality of wrapper methods. This may end up improving the predictive performance of SDP predictive models since it has been proven that ensemble methods are superior to individual classification algorithm [35].

At the end, the proposed hybrid method is expected to generate optimal subsets of features. The subsets will then be used to training predictive models in SDP.

4. Conclusion

Finding a way to hybridize filter and wrapper-based feature selection is still an open research issue. The reason for this is to maintain high performance and a generalizable result with simultaneous low computational cost. Based on these reasons, this study proposes a hybrid multi-filter wrapper feature selection method for software defect prediction. The proposed hybrid feature method is expected to be capable of selecting relevant and irredundant features from SDP datasets which improves the predictive performance of SDP models. In future, the researchers intend to explore other data quality problems such as class imbalance in SDP.

Acknowledgment.

This paper/research was fully supported by Ministry of Higher Education, Malaysia, under the Fundamental Research Grant Scheme (FRGS) with Ref. No. FRGS/1/2018/ICT04/UTP/02/04.

References

[1]. Ali, M. M., Huda, S., Abawajy, J., Alyahya, S., Al-Dossari, H., & Yearwood, J. (2017). A parallel framework for software defect detection and metric selection on cloud computing. *Cluster Computing*, 20(3), 2267-2281.

[2]. Yadav, H. B. & Yadav, D. K. (2015). A fuzzy logic based approach for phase-wise software defects prediction using software metrics. *Information and Software Technology*, 63, 44-57.

[3]. Huda, S., Alyahya, S., Ali, M. M., Ahmad, S., Abawajy, J., Al-Dossari, H., & Yearwood, J. (2018). A Framework for Software Defect Prediction and Metric Selection. *IEEE access*, 6, 2844-2858.

[4]. Li, Z., Jing, X.-Y., & Zhu, X. (2018). Progress on approaches to software defect prediction. *IET Software*.

[5]. Jing, X.-Y., Wu, F., Dong, X., & Xu, B. (2017). An improved SDA based defect prediction framework for both within-project and cross-project class-imbalance problems. *IEEE Transactions on Software Engineering*, 43(4), 321-339.

[6]. Tantithamthavorn, C., McIntosh, S., Hassan, A. E., & Matsumoto, K. (2017). An empirical comparison of model validation techniques for defect prediction models. *IEEE Transactions on Software Engineering*, 43(1), 1-18.

[7]. Arar, Ö. F. & Ayan, K. (2015). Software defect prediction using cost-sensitive neural network. *Applied Soft Computing*, 33, 263-277.

[8]. Zhang, F., Zheng, Q., Zou, Y., & Hassan, A. E. (2016). Cross-project defect prediction using a connectivity-based unsupervised classifier. Paper presented at the Proceedings of the 38th International Conference on Software Engineering.

[9]. Chen, R., Sun, N., Chen, X., Yang, M., & Wu, Q. (2018). Supervised Feature Selection With a Stratified Feature Weighting Method. *IEEE Access*, 6, 15087-15098.

[10]. Huda, S., Islam, R., Abawajy, J., Yearwood, J., Hassan, M. M., & Fortino, G. (2018). A hybrid-multi filter-wrapper framework to identify run-time behaviour for fast malware detection. *Future Generation Computer Systems*, 83, 193-207.

[11]. Bashir, K., Li, T., Yohannese, C. W., & Mahama, Y. (2017). Enhancing software defect prediction using supervised-learning based framework. Paper presented at the Intelligent Systems and Knowledge Engineering (ISKE), 2017 12th International Conference on.

[12]. Ghotra, B., McIntosh, S., & Hassan, A. E. (2017). A large-scale study of the impact of feature selection techniques on defect classification models. Paper presented at the Mining Software Repositories (MSR), 2017 IEEE/ACM 14th International Conference on.

[13]. Ni, C., Liu, W.-S., Chen, X., Gu, Q., Chen, D.-X., & Huang, Q.-G. (2017). A Cluster Based Feature Selection Method for Cross-Project Software Defect Prediction. *Journal of Computer Science and Technology*, 32(6), 1090-1107.

[14]. Jović, A., Brkić, K., & Bogunović, N. (2015). A review of feature selection methods with applications. Paper presented at the Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on.

[15]. Shepperd, M., Song, Q., Sun, Z., & Mair, C. (2013). Data quality: Some comments on the nasa software defect datasets. *IEEE Transactions on Software Engineering*, 39(9), 1208-1215.

[16]. Petrić, J., Bowes, D., Hall, T., Christianson, B., & Baddoo, N. (2016). The jinx on the NASA software defect data sets. Paper presented at the Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering.

[17]. Sheikhpour, R., Sarram, M. A., Gharaghani, S., & Chahooki, M. A. Z. (2017). A survey on semi-supervised feature selection methods. *Pattern Recognition*, 64, 141-158.

[18]. Wah, Y. B., Ibrahim, N., Hamid, H. A., Abdul-Rahman, S., & Fong, S. (2018). Feature Selection Methods: Case of Filter and Wrapper Approaches for Maximising Classification Accuracy. *Pertanika Journal of Science & Technology*, 26(1).

[19]. Akintola, A. G., Balogun, A. O., Lafenwa-Balogun, F. B., & Mojeed, H. A. (2018). Comparative Analysis of Selected Heterogeneous Classifiers for Software Defects Prediction Using Filter-Based Feature Selection Methods. *FUOYE Journal of Engineering and Technology*, 3(1), 134-137.

[20]. Jimoh, R., Balogun, A., Bajeh, A., & Ajayi, S. (2018). A PROMETHEE based evaluation of software defect predictors. *Journal of Computer Science and Its Application*, 25(1), 106-119.

[21]. Balogun, A. O., Bajeh, A. O., Orie, V. A., & Yusuf-Asaju, W. A. (2018). Software Defect Prediction Using Ensemble Learning: An ANP Based Evaluation Method. *FUOYE Journal of Engineering and Technology*, 3(2), 50-55.

[22]. Huda, S., Liu, K., Abdelrazek, M., Ibrahim, A., Alyahya, S., Al-Dossari, H., & Ahmad, S. (2018). An ensemble oversampling model for class imbalance problem in software defect prediction. *IEEE Access*.

[23]. Batarseh, F. A. & Gonzalez, A. J. (2018). Predicting failures in agile software development through data analytics. *Software Quality Journal*, 26(1), 49-66.
[24]. Braude, E. J. & Bernstein, M. E., *Software engineering: modern approaches*. Waveland Press, 2016.
[25]. Fenton, N. & Bieman, J., *Software metrics: a rigorous and practical approach*. CRC Press, 2014.
[26]. Chen, J., Liu, S., Liu, W., Chen, X., Gu, Q., & Chen, D. (2014). A two-stage data preprocessing approach for software fault prediction. Paper presented at the Software Security and Reliability (SERE), 2014 Eighth International Conference on.
[27]. Liu, W., Liu, S., Gu, Q., Chen, J., Chen, X., & Chen, D. (2016). Empirical studies of a two-stage data preprocessing approach for software fault prediction. *IEEE Transactions on Reliability*, 65(1), 38-53.
[28]. Liebchen, G. & Shepperd, M. (2016). Data sets and data quality in software engineering: eight years on. Paper presented at the Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering.
[29]. Ghotra, B., McIntosh, S., & Hassan, A. E. (2015). Revisiting the impact of classification techniques on the performance of defect prediction models. Paper presented at the Proceedings of the 37th International Conference on Software Engineering-Volume 1.
[30]. Gray, D., Bowes, D., Davey, N., Sun, Y., & Christianson, B. (2011). The misuse of the NASA metrics data program data sets for automated software defect prediction. Paper presented at the Evaluation & Assessment in Software Engineering (EASE 2011), 15th Annual Conference on.
[31]. Cai, J., Luo, J., Wang, S., & Yang, S. (2018). Feature selection in machine learning: A new perspective. *Neurocomputing*, 300, 70-79.
[32]. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6), 94.
[33]. Gui, J., Sun, Z., Ji, S., Tao, D., & Tan, T. (2017). Feature selection based on structured sparsity: A comprehensive study. *IEEE transactions on neural networks and learning systems*, 28(7), 1490-1507.
[34]. Hancer, E., Xue, B., & Zhang, M. (2018). Differential evolution for filter feature selection based on information theory and feature ranking. *Knowledge-Based Systems*, 140, 103-119.
[35]. Sun, Z., Song, Q., Zhu, X., Sun, H., Xu, B., & Zhou, Y. (2015). A novel ensemble method for classifying imbalanced data. *Pattern Recognition*, 48(5), 1623-1637.