"Analyzing the Effect with Integrated Technique for Feature Selection and Software Defect Prediction"

Raghvendra Omprkash Singh, Blessy Thankachan,

Abstract ***software defect prediction (sdp) technique was projected to designate testing assets sanely, decide the testing want of assorted modules of the software system, and improve programming quality. By utilizing the implications of sdp, programming specialists will fruitfully pass judgment on it that software system modules area unit sure to be blemished, the conceivable range of imperfections in a very module or different information known with software system defects before testing the software system [1]. Existing sdp studies may be divided into four types: (1) classification, (2) regression, (3) mining association rules, (4) ranking. The primary aim of the primary class is to classification of the software system entities like functions, classes, files, etc into completely different levels of severity with the assistance of various applied math techniques like supply regression [2] and discriminant analysis [3] and techniques of machine learning like svm [4] and ann [5]. The second kind aims to assess the amount of imperfections within the components of the software system by victimisation completely different ways, for instance, genetic programming, and support vector regression [6]. The third category utilizes association rule mining approaches, for instance, relative affiliation rule [7], and also the cba2 algorithmic rule, to mine the affiliation between the errors of programming components and programming measurements. The fourth category contemplates to rank the product of the software system as per the amount of errors in components or specifically streamlining the performance of ranking, i.e., faults share average (fpa) as indicated by existing studies of sdp [8]. Sdp distinguishes the modules that area unit imperfect and it needs a large scope of testing. Early recognizable proof of a blunder prompts viable allotment of assets, decreases the time and value of developing software system of high-quality. Hence, associate degree sdp model assumes a vital job in comprehending, assessing and rising the character of a product framework. Consequently, predicting deformity is incredibly basic within the field of reliableness and quality of software system. Predicting the defects is almost a unique analysis space of programming quality planning. By covering key indicators, forms of info to be assembled and also the role of sdp in software system quality, the connection among predictor and defect may be established.***

Keywords: faults percentage average (fpa) sdp, cba2 algorithm etc.

I. INTRODUCTION

It is applicable to specify that computer code defects decrease programming quality, increment prices additionally remit the schedule of development. victimization the techniques of computer code defect prediction, a team of computer code development will conjecture the conceivable bug and their seriousness within the underlying part of developing the computer code. Before the beginning of the testing stage, the method toward finding faulty components within the product is understood as computer code defect prediction. A standout amongst the foremost dynamic regions of analysis in computer code engineering is computer code defect prediction that prompts distended client loyalty, exaggerated responsible ness of computer code, a decrease within the time of advancement and reduction in work on effort and cost-adequacy. The activity of anticipating constant or ordered parameters for given information is understood as prediction. Hence, to attain the standard of computer code and to find out from previous errors, the prediction practices of defects is viewed as extraordinarily vital. Testing the computer code with associate degree aim to sight as several errors or flaw as potential before the computer code is discharged, plays main role for perform in guaranteeing the standard of the computer code. even so, with the event in scale and elaboration of the computer code, price of testing and length of typical computer code testing square measure increasing considerably. a way to improve checking effectiveness with unnatural testing assets to ensure programming quality is an implausible test to specialists and analysts. computer code defect prediction (SDP) technique was planned to designate testing assets sanely, decide the testing want of assorted modules of the computer code, and improve programming quality. By utilizing the implications of SDP, programming specialists will profitably pass judgment on it that computer code modules square measure absolute to be blemished, the conceivable range of imperfections in an exceedingly module or different knowledge known with computer code defects before testing the computer code [1]. Existing SDP studies may be divided into four types: (1) Classification, (2) Regression, (3) Mining association rules, (4) Ranking. The primary aim of the primary class is to classification of the computer code entities like functions, classes, files, etc into totally different levels of severity with the assistance of various applied math techniques like supplying regression [2] and discriminant analysis [3] and techniques of machine learning like SVM [4] and ANN [5]. The second kind aims to assess the amount of imperfections within the components of the computer code by victimization totally different ways, as an example, principles of genetics and natural selection code methods, and support vector regression [6].

Revised Manuscript Received on July 06, 2019.

Mr. Raghvendra Omprkash Singh, Department of Computer Systems and Sciences, Jaipur National University, Jaipur, India.

Dr. Blessy Thankachan, Department of Computer Systems and Sciences, Jaipur National University, Jaipur, India.

The third category utilizes association rule mining approaches, as an example, relative affiliation rule [7], and also the CBA2 algorithmic rule, to mine the association between the errors of programming components and programming measurements. The fourth category contemplates to rank the product of the computer code as per the amount of errors in components or specifically streamlining the performance of ranking, i.e., faults share average (FPA) as indicated by existing studies of SDP [8]. SDP distinguishes the modules that square measure imperfect and it needs a large scope of testing. Early recognizable proof of a blunder prompts viable allotment of assets, decreases the time and price of developing computer code of high-quality. Hence, associate degree SDP model assumes an important job in comprehending, assessing and rising the character of a product framework. consequently, predicting deformity is incredibly basic within the field of responsibleness and quality of computer code. Predicting the defects is almost a completely unique analysis space of programming quality coming up with. By covering key indicators, styles of info to be assembled and also the role of SDP in computer code quality, the link among predictor and defect may be established.

II. LITERATURE SURVEY

Software flaw forecast is to anticipate the flaw structured modules for the consecutive arrival of programming or cross venture programming. varied students used machine learning techniques on computer code defect prediction. In any case, for different realistic knowing's, defect info of all modules within the coaching information assortment might not be accessible. As of late, a couple of students factory-made semi-administered classifiers to foresee the modules with computer code defects. An improved semi-supervised learning approach for predicting the defects including category unbalanced and restricted marked issue of knowledge was introduced in [9]. this system utilizes Associate in Nursing arbitrary under-sampling procedure to resample the initial set of coaching of coaching information and refreshing the training set in every spherical for co-train vogue algorithmic rule. It makes the defect indicator increasingly realistic for real applications, by the mixture of those problems. A key part of those methods is that it simply doesn't modification the coaching info directly, however conjointly fabricates arrangement methods on the equitable info, by under sampling the new coaching information in every refreshing levels. In examination with customary AI approaches, this strategy significantly performed higher. alpha outcomes in addition incontestable that with the projected learning approach, it's conceivable to tack together higher technique to handle the category unbalanced issue in semi-supervised learning. The authors in [10] investigated the execution of semi-supervised learning for programming flaw forecast. A preprocessing system, flat scaling, is embedded so as to decrease the dimensional unpredictability of programming measurements utilized for the forecast. The outcomes demonstrate that the activity decrease with semi-administered learning algorithmic rule performs basically superior to at least one of the most effective performing arts directed learning algorithmic rule - random forest - in circumstances once some of modules with far-famed fault content area unit accessible. a mixture of semi-supervised learning and also the dimension reduction technique provides necessary advantages to computer code quality prediction. The overwhelming majority of existing methods assume that heaps of tagged recorded info area unit accessible for prediction, whereas within the starting time of the prevailing cycle, tasks might not have the knowledge needed for structuring such indicators. Also, the larger a part of existing systems utilizes static code measurements as indicators, whereas they preclude modification information which will bring dangers in developing the computer code. Considering these problems a semi-supervised primarily based approach to predict the defects within the computer code - extRF was developed [11]. extRF expands the normal Random Forest algorithmic rule by self-training. Also, burst info is utilized for predicting the defects within the computer code. Experiments area unit conducted to assess the extRF against alternative techniques of supervised machine learning with relevance modification burst metrics, code metrics. alpha outcomes demonstrate that extRF ready with a smaller size of dataset that's tagged accomplishes equivalent execution to some administered learning techniques developed with a much bigger size of the marked dataset. A consolidated methodology for computer code defect prediction and detective work the programming bugs area unit introduced in [11]. The projected approach conveys an inspiration of reducing the options and AI wherever the theme of principle component analysis (PCA) reduces the options. this method is any improved by together with maximum-likelihood estimation for reducing the errors whereas reconstructing the PCA information. In the end, the neural system primarily based technique for classifying is employed that indicates forecast results. A structure is planned and dead on NASA programming dataset wherever four datasets i.e., KC1, PC3, PC4 and JM1 area unit thought-about for analyzing the performance utilizing MATLAB replica device. A broad experimental check is administered to gauge the system with relevance the accuracy of classification, recall, precision. therefore, it is often inferred that by utilizing techniques of feature choice the complexities because of time and area for predicting the defects while not poignant the accuracy of prediction.

Sl no.	Paper title	Findings / conclusions
1	An improved semi-supervised learning method for software defect prediction [9]	They improved the semi-supervised learning method to build software defect prediction model. They also resolved the problem of class imbalance in a semi-supervised learning by combining the method of random under sampling along with Tri-training algorithm.
2	A semi-supervised approach to software defect prediction [10]	In this study proved that the algorithm performed better than the random forest supervised learning algorithm in the cases where known fault content of some modules are available.
3	Software defect prediction using semi-supervised learning with change burst information [11]	This study developed an extended random forest algorithm for software defect prediction. This algorithm also employed the change burst information for enhancing the accuracy of software defect prediction.
4	Applying feature selection to software defect prediction using multi-objective optimization [12]	This study proposed application of feature selection techniques for resolving the problem of curse of dimensionality in predicting software defects. They formalized the feature selection for predicting software defects as a multi-objective optimization problem. This method had the advantage of lesser number of features being selected and achieved better accuracy when compared to other techniques.
5	Automated parameter optimization of classification techniques for defect prediction models [13]	This study investigated the performance of defect prediction models by applying Caret - an automated parameter optimization technique. They ended that the parameter settings have a big impact on the defect prediction model. The study counseled the students to explore the automatic parameter improvement rather than counting on the default parameter settings with an assumption that the other parameter settings do not lead to considerable improvements.
6	Software defect prediction using ensemble learning on selected features [14]	The primary objective of this study was to study the positive effects of combining ensemble learning and feature selection on the performance of classifying the software defects. From the results it could be inferred that the features of the software dataset must be carefully chosen for precisely classifying the defects.

7	A learning - to - rank approach to software defect prediction [14]	This study introduced a learning to rank technique for constructing the software defect prediction model by the direct optimization of the ranking performance. The comparison of the developed algorithm with the existing ones showed that the proposed algorithm had higher accuracy of classification.
8	Software defect prediction using cost-sensitive neural network	This study combined the ANN and the novel Artificial Bee Colony (ABC) algorithm for classifying the software defects. The proposed approach was applied to five publicly available datasets from the NASA Metrics Data Program repository. Accuracy, probability of false alarm, probability of detection, probability of false alarm, balance, Area Under Curve (AUC), and Normalized Expected Cost of Misclassification (NECM) were the primary indicators of the performance of the developed model.
9	MAHAKIL: Diversity based oversampling approach to alleviate the class imbalance issue in software defect prediction	This study developed a simple and effective synthetic oversampling technique MAHAKIL for for software defect datasets which were based on chromosomal theory of inheritance. The experiments indicated that the MAHAKIL improved the prediction accuracy and achieved better and considerable values of pf in comparison with the other oversampling approaches.
10	MULTI: Multi-objective effort-aware just-in-time software defect prediction	This study proposed a multi-objective optimization based supervised approach MULTI to develop Just-in-time software defect prediction (JIT-SDP) models. The objectives were designed such that one maximized the number of identified buggy changes while the other minimized the efforts in the activities of software quality assurance. It could be inferred that the MULTI performed better than the state-of-the-art supervised and unsupervised methods thus confirming that the techniques of supervised learning are still a feasible option in effort-aware JIT-SDP.

III. PROPOSED SYSTEM:

Defect prediction is incredibly fundamental in the field of software quality and reliability. Developing a suitable training dataset arranged from an expansive number of projects on public software repositories is indeed a challenge for cross-project defect prediction (CPDP). Due to the issue of dataset shift between cross-project software defect data, an ideal execution can't be accomplished by directly adopting AI classifier. It creates the impression that optimizing directly for the expense and not the performance of the forecast model is presently the best way to deliver generally cost-proficient CPDP models. The defect forecast and key metric choice issue as a single optimization issue. To predict defects while not overrunning the calculable price further as while not delaying regular delivery of package. The event of imperfections is inescapable, however, we must always endeavor to confine these deformities to a minimum count. Defect prediction ends up in reduced development time, cost, reduced work on effort, accumulated client satisfaction and additional reliable package. Therefore, defect prediction practices ar necessary to attain package quality and to be told from past mistakes. Size or complexness measures ar statistical procedure models, that unremarkably assume a straightforward relationship between defects and program complexness. These models aren't subjected to the controlled applied mathematics testing needed to line up a causative relationship. WPDP has an obvious drawback when a project has limited historical defect data.

1. Aim and Objectives

The main aim of the study is to develop an effective software defect prediction model using CSKDL approach. The primary objectives of the study are

- To develop a semi-supervised dictionary learning (SDL) technique.
- To enhance the SDL for a cost-sensitive kernelized semi-supervised dictionary learning (CKSDL) approach that can be utilized for CSDP as well as WSDP.
- To optimize the feature selection using multiobjective optimization technique.

2. Proposed methodology

This study proposes cost-sensitive kernelized semi-supervised dictionary learning (CKSDL) approach that can be utilized for CSDP as well as WSDP to predict the defects in a software module. The performance of the system is further improved by optimizing the cost factor with the help of multi-objective optimization technique for generating Pareto optimal solutions. First, the data is collected and preprocessed to form three different sets of defect-free, defective and unlabeled. Then features like cost are extracted using which is data classified using the CSDKL technique. A feature optimization technique is used for improving the performance of the system. If P = [p_1, p_2, . . ., p_N] = [P_1, P_2, P_3] is the set of modules, where a_n represents the nth module in P and P_1, P_2, P_3 are subsets of defective, defect-free, and unlabeled modules, respectively. The structured dictionary C = [C_1, C_2, C_3] with the module set P, where C_1, C_2, and C_3 are sub dictionaries that are associated with defective, defect-free, and unlabeled modules respectively are learnt. Considering the sparse coding coefficient matrix of P over C to be X = [x_1, x_2, . . . , x_N] = [X_1, X_2, X_3] i.e. P≈CX where X_i is the submatrix containing the coding coefficients of P_i over C, and x_n is the representation coefficients of representing p_n with C. the objective function of SDL can be defined as

$$J_{(C,X)} = \underset{(C,X)}{arg\ min}\{r(P,C,X) + \lambda\|X\|_1\} \quad (1)$$

Where r(P, C, X) is a semi-supervised discriminative fidelity term, $||X_1||$ is the sparsity constraint, and the balance factor is λ. The semisupervised discriminative fidelity term can be defined as

$$r(P,C,X) = \sum_{i=1}^{3} r(P_i,C,X_i) \quad (2)$$

$$r(P_i,C,X_i) = \begin{cases} \|P_i - CX_i\|_F^2 + \|P_i - C_iX_i\|_F^2 + \|C_iX_i\|_F^2 \Big|_{\substack{j=1,2,\\ j\neq i}} & i=1,2 \\ \|P_i - CX_i\|_F^2 & i=3 \end{cases}$$

For a query module a_n in P_3 , its label can be predicted using equation 3.

$$label(p_n) = \underset{i}{arg\ min} \quad \{\|p_n - C_i x_n^i\|_2 | i=1,2\} \quad (4)$$

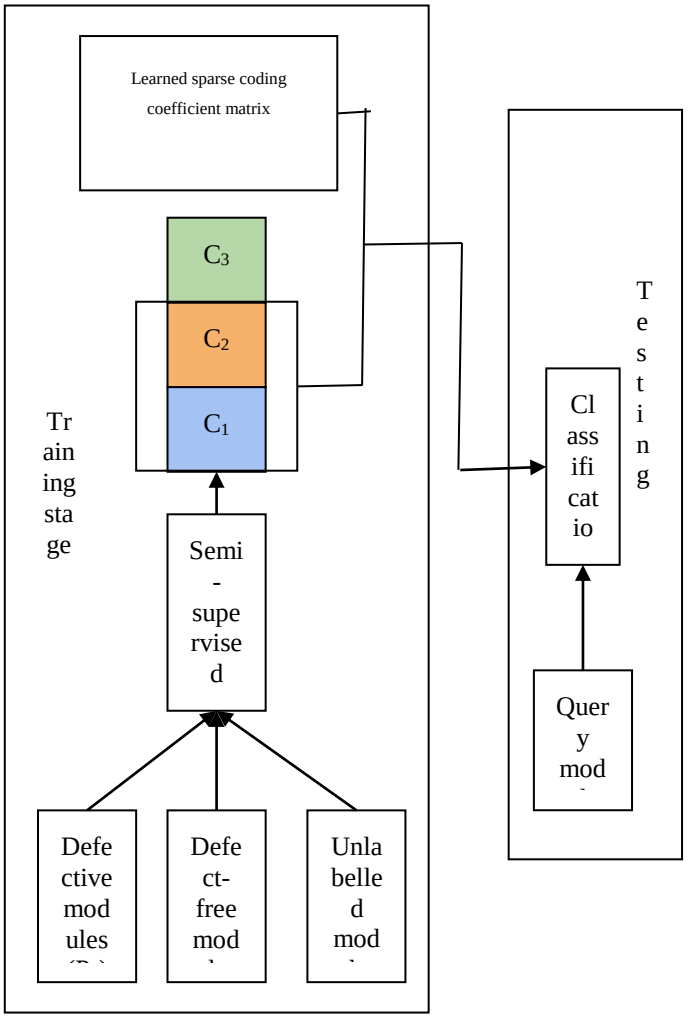

Figure 1. Basic SDL model

Kernel mapping will create modules to be distinct to some extent, and learning dictionaries obsessed on the kernel mapped modules are smart for predicting. For this initial, the initial information is mapped into a high-dimensional reproducing kernel Hilbert space (RKHS) then kernel principal element analysis (KPCA) is performed for reducing the spatial property of the modules that area unit kernel mapped. Let Pφ1, Pφ2, and Pφ3, singly denote the subsets of the defective, defect-free, and unlabeled modules once performing arts kernel mapping and KPCA, and Aφ =[Pφ1, Pφ2, Pφ3]. Then, the target perform of KSDL is given by

$$J_{(C,X)} = \underset{(C,X)}{arg\ min}\{r(P^{\varphi},C,X) + \lambda\|X\|_1\} \quad (5)$$

$$r(P^{\varphi}{}_i,C,X_i) = \begin{cases} \|P^{\varphi}{}_i - CX_i\|_F^2 + \|P^{\varphi}{}_i - C_iX_i\|_F^2 + \|C_iX_i\|_F^2 \Big|_{\substack{j=1,2,\\ j\neq i}} & i=1,2 \\ \|P^{\varphi}{}_i - CX_i\|_F^2 & i=3 \end{cases} \quad (6)$$

Figure 2. Flowchart of using CKSDL

Misclassifying the modules that area unit defective can end in a rise within the value and misclassification of defect-free modules will be associated with inflated value for developing. The techniques that area unit value sensitive will embrace the varied prices of misclassification into the prediction method. Taking inspiration from the cost-sensitive learning, penalty issue cost(i, j) is added into semi-supervised discriminative fidelity term (4) once a computer code module from category i is misclassified into category j. The prediction is formed inclined to categoryify the module into class i once cost (i, j) > cost (j, i) with associate aim to be told lexicon with least misclassification value. The semi-supervised discriminative fidelity term with penalty factors is given by the smallest amount misclassification value is obtained by more optimizing the price issue with the assistance of multiobjective improvement technique. The pseudocode for the multiobjective improvement technique is given below.

Input

Population Size: N, Maximum Iteration Number: T

Output:

Pareto Optimal Set

$i \leftarrow 0$

$P_i \leftarrow$ initPop(N)

while $i < T$ **do**

$C_i \leftarrow$ make New Pop(P_i)

$B_i \leftarrow P_i \cup C_i$

$F \leftarrow$ fast Nondominated Sort(B_i)

$P_{i+1} \leftarrow \emptyset$,

$j \leftarrow 1$

while $|P_{i+1}| + |F_j| \leq N$ **do**

crowingDistanceAssign(F_j)

$P_{i+1} \leftarrow P_{i+1} \cup F_j$

$j \leftarrow j + 1$

end while

sort(F_j) //according to crowding distance

$P_{i+1} \leftarrow P_{i+1} \cup F_j\,[1 : (N - |P_{i+1}|)]$

$i \leftarrow i + 1$

end while

return Pareto optimal solutions in P_i

The proposed CKSDL approach can be utilized for both CSDP and WSDP. In the WSDP case, the unlabeled module set P_3 incorporates the modules of within-project query which have to forecasted. Furthermore, in the CSDP situation, P_3 demonstrates the combined unlabeled models of within-project query and defect from the other projects. A common subspace can be acquired by learning the complete dictionary of both the source as well as the target projects. Along these lines, the issue of appropriation contrast across projects can be mitigated. The technique of utilizing the solution of CKSDL for CSDP and WSDP issues is illustrated in figure 2.

REFERENCES

1. Tong, H., Liu, B., & Wang, S. (2018). Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning. *Information and Software Technology*, *96*, 94-111.
2. Ma, Y., Qin, K., & Zhu, S. (2014). Discrimination analysis for predicting defect-prone software modules. *Journal of Applied Mathematics*, *2014*.
3. Kamei, Y., Shihab, E., Adams, B., Hassan, A. E., Mockus, A., Sinha, A., & Ubayashi, N. (2013). A large-scale empirical study of just-in-time quality assurance. *IEEE Transactions on Software Engineering*, *39*(6), 757-773.
4. Ricky, M. Y., Purnomo, F., & Yulianto, B. (2016, March). Mobile application software defect prediction. In *2016 IEEE Symposium on Service-Oriented System Engineering (SOSE)*(pp. 307-313). IEEE.
5. Kanmani, S., Uthariaraj, V. R., Sankaranarayanan, V., & Thambidurai, P. (2007). Object-oriented software fault prediction using neural networks. *Information and software technology*, *49*(5), 483-492.
6. Rathore, S. S., & Kumar, S. (2015). Predicting number of faults in software system using genetic programming. *Procedia Computer Science*, *62*, 303-311.
7. Czibula, G., Marian, Z., & Czibula, I. G. (2014). Software defect prediction using relational association rule mining. *Information Sciences*, *264*, 260-278.
8. You, G., & Ma, Y. (2016). A Ranking-Oriented Approach to Cross-Project Software Defect Prediction: An Empirical Study. In *SEKE* (pp. 159-164).
9. Ma, Y., Pan, W., Zhu, S., Yin, H., & Luo, J. (2014). An improved semi-supervised learning method for software defect prediction. *Journal of Intelligent & Fuzzy Systems*, *27*(5), 2473-2480.
10. Lu, H., Cukic, B., & Culp, M. (2014, July). A semi-supervised approach to software defect prediction. In *2014 IEEE 38th Annual Computer Software and Applications Conference* (pp. 416-425). IEEE.
11. He, Q., Shen, B., & Chen, Y. (2016, June). Software defect prediction using semi-supervised learning with change burst information. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)* (Vol. 1, pp. 113-122). IEEE.
12. 12.Chen, X., Shen, Y., Cui, Z., & Ju, X. (2017, July). Applying feature selection to software defect prediction using multi-objective optimization. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)* (Vol. 2, pp. 54-59). IEEE.
13. 13.Tantithamthavorn, C., McIntosh, S., Hassan, A. E., & Matsumoto, K. (2016, May). Automated parameter optimization of classification techniques for defect prediction models. In 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE) (pp. 321-332). IEEE.
14. 14.Laradji, I. H., Alshayeb, M., & Ghouti, L. (2015). Software defect prediction using ensemble learning on selected features. Information and Software Technology, 58, 388-402.