Biosc.Biotech.Res.Comm. Special Issue Vol 13 No 15 (2020) Pp-292-296

BBRC
Bioscience Biotechnology Research Communications

# Empirical Analysis of Object-Oriented Metrics for Prediction of Software Faults

Jyoti Goyal and Bal Kishan
*Department of Computer Science and Applications, MDU, Rohtak, India*

## ABSTRACT

Software fault prediction is the highly researched field due to its significance in the software industry. Object oriented metrics plays a significant role in software fault prediction. Various features of object orientation like inheritance, polymorphism, abstraction, data encapsulation is helpful in finding fault prone module that will significantly reduce the software maintenance cost. This paper presents an empirical analysis of the significance of object-oriented metrics for building an efficient fault prediction model. The experiment is performed on Camel 1.6 dataset collected from promise data repository. To exhibit the importance of object-oriented metrics for prediction of faults a correlation table is also prepared. The model was build using various machine learning techniques like SVC, Logistic regression, Adaboost, Bernoulli naïve bayes and the results proved that object-oriented metrics are good indicators of fault prone modules.

**KEY WORDS:** FAULTS, MACHINE LEARNING TECHNIQUES, OBJECT-ORIENTED METRICS, SOFTWARE FAULT PREDICTION.

## INTRODUCTION

The last two decade was completely dedicated to the object-oriented approach for software development. It is the root of modern-day software development. It helps to develop stable and maintainable software product. Object oriented metrics are the measurements of the various aspects of object-oriented software. Generally, each software metric is related to some functional properties of the software project such as coupling, cohesion, inheritance, code change etc., and is used to indicate an external quality attribute such as reliability, testability, or fault-proneness. The performance of the model is highly influenced by the type of metrics used. The popularity of object-oriented metrics lies in the researches where these were used twice as when compared with traditional metrics. This study investigates the significance object-oriented metrics with regard to defect prediction. A correlation analysis was also performed in order to investigate the relation between the software metrics and faulty modules. Figure 1 show various design measures related to fault proneness of a class.

ARTICLE INFORMATION

*Corresponding author email: jyoti.goyal24@gmail.com
Received 12th Oct 2020 Accepted after revision 29th Dec 2020
Print ISSN: 0974-6455 Online ISSN: 2321-4007 CODEN: BBRCBA

*Thomson Reuters ISI Web of Science Clarivate Analytics USA and Crossref Indexed Journal*

Clarivate Analytics

crossref

*NAAS Journal Score 2020 (4.31)*

*Online Contents Available at: http//www.bbrc.in/*
*Doi: http://dx.doi.org/10.21786/bbrc/13.15/51*

### II. Object Oriented Metrics Used For Software Fault Prediction

Object-oriented (OO) metrics are divided into two categories: structural metrics and dynamic metrics.

**A. Static metrics:** Static metrics measures different aspects of the source code by doing static analysis of the code. we have many structural OO metrics suites proposed by various researchers. Some of them are given below:

1. CK metrics suite proposed by chaidamber and kemere in 1994. It includes "Coupling between Object class CBO), Lack of Cohesion in Methods (LCOM), Depth of Inheritance Tree (DIT), Response for a Class (RFC), Weighted Method Count (WMC) and Number of Children (NOC)"
2. MOODS metrics suite is proposed by Harrison and

Counsel in 1998. It includes "Method Hiding Factor (MHF), Attribute Hiding Factor (AHF), Method Inheritance Factor (MIF), Attribute Inheritance Factor (AIF), Polymorphism Factor (PF), Coupling Factor (CF)"

3. Wei Li and Henry metrics suite is proposed by Li and Henry in 1996. It includes "Coupling Through Inheritance, Coupling Through Message passing (CTM), Coupling Through ADT (Abstract Data Type), Number of local Methods (NOM), SIZE1 and SIZE2"
4. Lorenz and Kidd's metrics suite are proposed by Lorenz and Kidd in 1994. It includes "PIM, NIM, NIV, NCM, NCV, NMO, NMI, NMA, SIX and APPM"
5. Bansiya metrics suite proposed by Bansiya and Davis in 2002. It includes "DAM, DCC, CIS, MOA, MFA, DSC, NOH, ANA, CAM, NOP and NOM"
6. Briand metrics suite is proposed by Briand et al. in 1997. It includes "IFCAIC, ACAIC, OCAIC, FCAEC, DCAEC, OCAEC, IFCMIC, ACMIC, OCMIC, FCMEC, DCMEC, OCMEC, IFMMIC, AMMIC, OMMIC, FMMEC, DMMEC, OMMEC"

Figure 1: Object oriented design measures related to fault proneness of a class

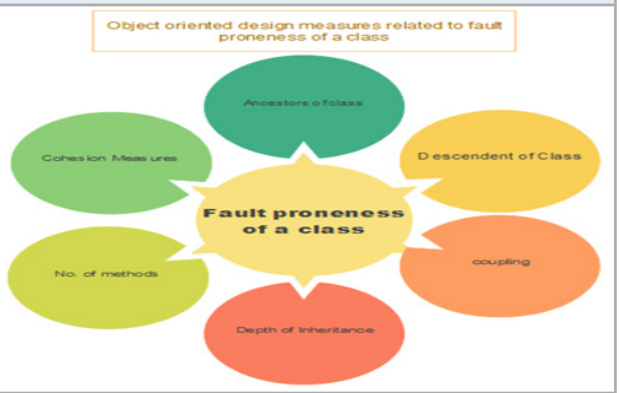
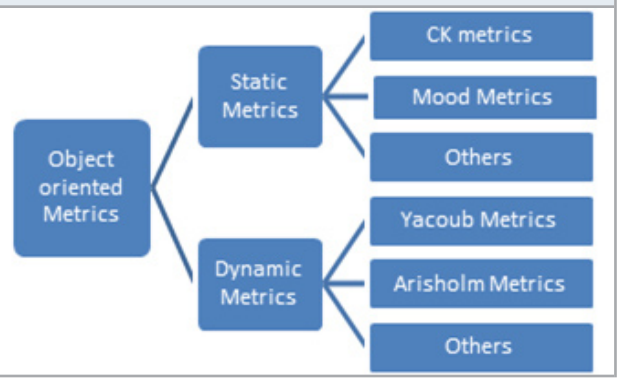



Figure 2: Classification of object-oriented metrics




**B. Dynamic metrics:** Dynamic metrics refer to the set of metrics which depends on the features gathered from a running program. These metrics reveal behaviour of the software components during execution, and are used to measure specific runtime properties of programs, components, and systems. There are many dynamic metrics suite that are given below:

1. Yacoub metrics suite proposed by Yacoub et al. in 1999. It includes "Export Object Coupling (EOC) and Import Object Coupling (IOC)"
2. Arisholm metrics suite proposed by Arisholm in 2004. It includes "IC_OD, IC_OM, IC_OC, IC_CD, IC_CM, IC_CC, EC_OD, EC_OM, EC_OC, EC_CD, EC_CM, EC_CC"
3. Mitchell metrics suite proposed by Mitchell and power in 2006. It includes "Dynamic CBO for a class, Degree of dynamic coupling between two classes at runtime, Degree of dynamic coupling within a given set of classes, RI, RE, RDI, RDE".

**III. Analysis Of Existing Literature:** Many previous studies have examined the capabilities of object-oriented metrics for software fault prediction and found that object-oriented metrics performed better than static code metrics in predicting software faults because metrics represent various structural characteristics of object-oriented software systems like coupling, cohesion, inheritance, encapsulation, complexity, and size metrics. The research evidences pertaining to it are presented in following text. Syed rashid aziz [2020] performed a survey on inheritance metrics in object-oriented metrics for finding the faults in the software modules.

Md. Fahimuzzman et al. [2019] proposed a framework to handle the class imbalance issue. Author used object-oriented metrics to find the correspondence between object-oriented attributes and faults. Emam et al. proposed a model to predict vulnerability of a class on the essence of CK metrics. The probability that class has a fault was estimated with logistic regression model. Fioravanti and Nesi applied Principal Component Analysis and Multivariate Logistic Regression model to estimate the proneness of different software components. The work of Gyimothy, Ferenc, and Siket studied the effectiveness of individual CK metrics in OO software. The study collected the bugs data for Bugzilla, which is open-source software. The study found that CK metrics reveals low severity faults better.

Singh, Kaur, and Malhotra also carried out the similar investigations. Olague et al. evaluated three OO metrics suites: CK, MOOD and QMOOD to predict fault proneness of the classes. Their study used 'Rhino' software as subject software. They concluded that QMOOD and CK metrics encompass similar constituents. Study also confirms the capabilities of these two suites in fault prediction. As per their study the components of MOOD suite are not much effective predictors of fault proneness. Elish, Al-Yafei, and Al-Mulhem experimentally compared effectiveness of MOOD, CK and Martin's suites for fault prediction in java packages. The eclipse IDE was used as subject program for the study. The study revealed that composite models based on MOODS and Martin's suites performed better than composite models based on CK and MOOD.

Xu, Ho, and Capretz found that SLOC, RFC, CBO and WMC metrics of CK suite are reliable ones in predicting the fault. Authors incorporated Spearman's rank correlation coefficient method and LR technique to

study the inter-dependence of metrics and software fault-proneness. Malhotra and Khanna further enacted machine learning and search based techniques (SBT) to determine the relationship of OO metrics and change prediction. In their work they investigated effectiveness of six SBT, four machine learning techniques and the Logistic Regression (LR). This study advocates the use of methods based on SBT in classification of classes for their change proneness. In Malhotra and Bansal put thresholds on the object-oriented metrics. The metrics worked upon are CK metrics.

Table 1. Description of tool

| Tool | Description | IDE |
|---|---|---|
| Anaconda Distribution | Anaconda is a birthplace of python data science. Anaconda is a package manager, an environment manager, and Python distribution that contains a collection of many open-source packages | Jupyter Notebook |

Table 2. Description of Dataset

| Dataset | No. of instances | No. of attributes | No. of buggy classes | No. of non-buggy classes |
|---|---|---|---|---|
| Camel 1.6 | 965 | 24 | 189 | 776 |

Figure 3: Detailed description of dataset

| | wmc | dit | noc | cbo | rfc | lcom | ca | ce | npm | lcom3 | ... | dam | moa | mfa | cam | ic | cbm | amc | max_cc | avg_cc | bug |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5 | 3 | 0 | 7 | 10 | 0 | 1 | 7 | 4 | 0.250 | ... | 1.000 | 1 | 0.921 | 0.360 | 1 | 2 | 7.400 | 1 | 0.600 | 0 |
| 1 | 4 | 1 | 0 | 3 | 5 | 4 | 1 | 2 | 3 | 0.667 | ... | 1.000 | 1 | 0.000 | 0.500 | 0 | 0 | 3.000 | 1 | 0.500 | 0 |
| 2 | 20 | 4 | 0 | 26 | 95 | 144 | 2 | 26 | 13 | 0.842 | ... | 1.000 | 0 | 0.727 | 0.197 | 4 | 5 | 20.300 | 3 | 1.000 | 0 |
| 3 | 3 | 2 | 0 | 8 | 22 | 3 | 2 | 6 | 2 | 2.000 | ... | 0.000 | 0 | 0.750 | 0.667 | 1 | 3 | 54.000 | 15 | 5.333 | 1 |
| 4 | 8 | 1 | 0 | 25 | 20 | 22 | 22 | 3 | 6 | 0.571 | ... | 1.000 | 0 | 0.000 | 0.250 | 0 | 0 | 20.875 | 1 | 0.750 | 1 |
| 5 | 3 | 1 | 0 | 6 | 3 | 3 | 5 | 2 | 3 | 2.000 | ... | 0.000 | 0 | 0.000 | 1.000 | 0 | 0 | 0.000 | 1 | 1.000 | 0 |
| 6 | 1 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 2.000 | ... | 0.000 | 0 | 0.000 | 1.000 | 0 | 0 | 3.000 | 0 | 0.000 | 0 |
| 7 | 6 | 1 | 0 | 9 | 17 | 15 | 2 | 7 | 6 | 2.000 | ... | 0.000 | 0 | 0.000 | 0.333 | 0 | 0 | 6.667 | 4 | 1.333 | 0 |
| 8 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2.000 | ... | 0.000 | 0 | 0.000 | 0.000 | 0 | 0 | 0.000 | 0 | 0.000 | 0 |
| 9 | 4 | 1 | 0 | 7 | 8 | 6 | 0 | 7 | 3 | 2.000 | ... | 0.000 | 0 | 0.000 | 0.375 | 0 | 0 | 4.500 | 1 | 0.750 | 0 |

Rosli et al proposed a fault prediction model by using the values of object-oriented metrics from the web application as input values to the GA to predict the faulty systems. The main goal of the proposed model is to find the most likely software modules that might be the most problematic module in the future. The authors selected eight internal metrics to clarify the main design attributes of object-oriented applications, and these metrics are lines of code (LOC), coupling between object classes (CBO), response for a class (RFC), depth inheritance tree (DIT), number of children (NOC), weighted methods per class (WMC), lack of cohesion (LCOM), and number of public methods (NPM) and they used a single constrained fitness function to specify the optimal metrics conjunction in order to maximize the percentage of faults, which means fewer faults. It is evident from the literature reviews that object-oriented metrics are extensively being exploited to examine fault proneness of a software component. And it's still an area of investigation.

**Dataset and Tool Used in the Study:** To perform the experiment, we have used the dataset camel-1.6 class level dataset available in promise data repository. The tool we used is anaconda python distribution. The description of dataset and tool is mentioned in the following table 1 and Table 2. The detailed description of dataset is shown in Fig. 3:

Figure 4: Data balancing ratio of dataset

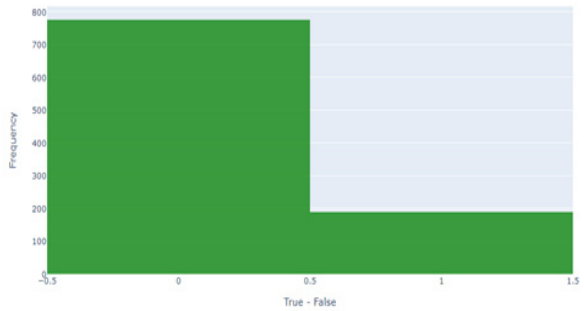



Figure 5: Identification of outliers using boxplot

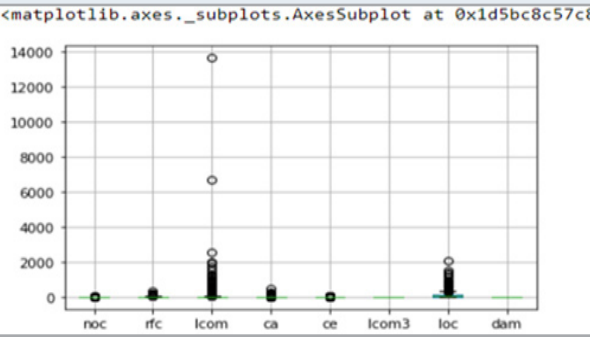


Figure 7: Performance score of SVM

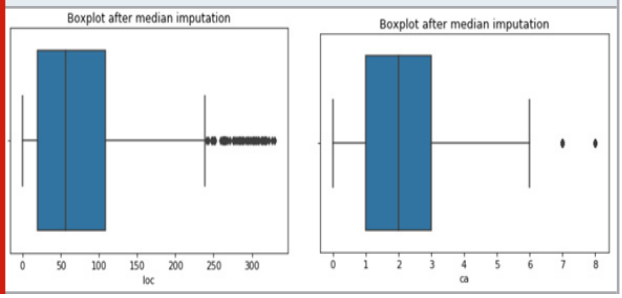


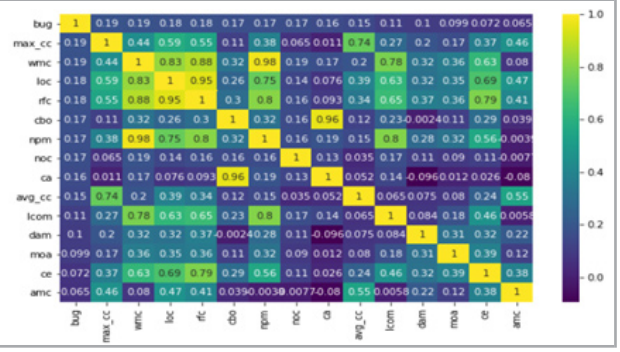

Figure 7: Performance score of SVM

```
[[222   0]
 [ 68   0]]
              precision    recall  f1-score   support

           0       0.77      1.00      0.87       222
           1       0.00      0.00      0.00        68

    accuracy                           0.77       290
   macro avg       0.38      0.50      0.43       290
weighted avg       0.59      0.77      0.66       290
```

Table 3. Results of Classifiers

| Name of Classifier | Accuracy score |
|---|---|
| Logistic Regression | 0.75 |
| AdaBoost | 0.86 |
| Bernaulli Naïve Bayes | 0.76 |

Figure 8: Corelation of object-oriented metrics with faults




**Dataset and Tool Used in the Study:** To perform the experiment, we have used the dataset camel-1.6 class level dataset available in promise data repository. The tool we used is anaconda python distribution. The description of dataset and tool is mentioned in the following table 1 and Table 2. The detailed description of dataset is shown in Fig. 3:

## CONCLUSION AND FUTURE SCOPE

In this paper we have empirically analysed the importance of object-oriented metrics in finding the fault prone modules. The results proved that object-oriented metrics are good indicator of faulty modules. Also, we find the correlation of various software metrics with the faults. The value close to 1 is highly correlated with the bug. So, on that basis we also recommend the set of metrics that are good indicators of fault prone modules. This study greatly helps the practitioners in finding the right set of object-oriented metrics for prediction of faults in their future projects. In the future scope they can implement the same model with different dataset or with different classifiers.

## REFERENCES

Aziz, S. R., Khan, T. A., & Nadeem, A. (2020). Efficacy of Inheritance Aspect in Software Fault Prediction—A Survey Paper. IEEE Access, 8, 170548-170567.

Bansal, M., & Agrawal, C. P. (2014, February). Critical analysis of object oriented metrics in software development. In 2014 Fourth International Conference on Advanced Computing & Communication Technologies (pp. 197-201). IEEE.

Basili, V. R., Briand, L. C., & Melo, W. L. (1996). A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering, 22(10), 751-761.

Capretz, L. F., & Xu, J. (2008). An empirical validation of object-oriented design metrics for fault prediction. Journal of Computer Science, 4(7), 571.

E. Arisholm, L.C Briand, A. Foyen, "Dynamic Metric Coupling Measures for Object Oriented Software", IEEE Transactions of Software Engineering, 30(8):491-506, 2004.

Elish, M. O., Al-Yafei, A. H., & Al-Mulhem, M. (2011). Empirical comparison of three metrics suites for fault prediction in packages of object-oriented systems: A case study of eclipse. Advances in Engineering Software, 42(10), 852-859.

Emam K E, Benlarbi S, Goel N and Rai S N 2001 The confounding effect of class size on the validity of objectoriented metrics. IEEE Trans. Softw. Eng. 27(7)

F.B. Abreu, "The MOOD Metrics Set", In Proc. ECOOP'95, Workshop on Metrics, 1995.

Fioravanti, F., & Nesi, P. (2001, March). A study on fault-proneness detection of object-oriented systems. In Proceedings Fifth European Conference on Software Maintenance and Reengineering (pp. 121-130). IEEE.

Gupta, D. L., & Saxena, K. (2017). Software bug prediction using object-oriented metrics. Sādhanā, 42(5), 655-669.

Gyimóthy, 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.

J Bansiya and C.G. Davis, "A Hierarchical Model for Object Oriented Design Quality Assessment", IEEE Transactions on Software Engineering, Vol. 28, No. 1, 2002.

Kanungo, P., & Sharma, M. K. Approach for Improve Software Fault Prediction based on Mandel bugs.

Li, W. (1998). Another metric suite for object-oriented programming. Journal of Systems and Software, 44(2), 155-162.

Lorenz, Mark and Kidd, Jeff, "Object Oriented Software Metrics", Prentice Hall Publishing, 1994.

Malhotra, R., & Bansal, A. J. (2015). Fault prediction considering threshold effects of object oriented metrics. Expert Systems, 32(2), 203-219.

Malhotra, R., & Khanna, M. (2013). Investigation of relationship between object-oriented metrics and change proneness. International Journal of Machine Learning and Cybernetics, 4(4), 273-286.

Olague, H. M., Etzkorn, L. H., Gholston, S., & Quattlebaum, S. (2007). Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes. IEEE Transactions on software Engineering, 33(6), 402-419.

Pham, V., Lokan, C., & Kasmarik, K. (2020). A Better Set of Object-Oriented Design Metrics for Within-Project Defect Prediction. In Proceedings of the Evaluation and Assessment in Software Engineering (pp. 230-239).

Ponnala, R., & Reddy, C. R. K. (2019). Object Oriented Dynamic Metrics in Software Development: A Literature. International Journal of Applied Engineering Research, 14(22), 4161-4172.

S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object Oriented Design", IEEE Transactions on Software Engineering, Vol. 20, No.6, pp. 476-493, 1994.

Singh, Y., Kaur, A., & Malhotra, R. (2009). Application of support vector machine to predict fault prone classes. ACM SIGSOFT Software Engineering Notes, 34(1), 1-6.

Sohan, M. F., Kabir, M. A., Jabiullah, M. I., & Rahman, S. S. M. M. (2019, February). Revisiting the class imbalance issue in software defect prediction. In 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE) (pp. 1-6). IEEE

Y. Hassoun, R. Johnson, S. Counsell, Dynamic coupling metric: proof of concept, in IEE Proceedings -Software, 152: 273-279, 2005

Y. Hassoun, R. Johnson, S. Counsell, "A Dynamic Runtime Coupling Metric for Meta Level Architectures", in Proceedings of Eighth Euromicro Working Conference on Software Maintenance and Reengineering (CSMR '04), pp. 339, 2004.