Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 132 (2018) 993–1001

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2018)

Taxonomy of machine learning algorithms in software fault prediction using object oriented metrics

Ajmer Singh[a*], Rajesh Bhatia[b], Anita Singhrova[a]

[a] *Computer Science & Engineering Department DCRUST , Murthal, Sonipat 131039, India*
[b] *Computer Science & Engineering Department, PEC University of Technology, Chandigarh, India*

Abstract

Prediction of Fault proneness of a software component is the compelling field of investigations in software testing arena. Software coupling plays a vital role in assessing the software quality through fault prediction and complexity measures. Various fault prediction models, have used the object oriented metrics for the predicting and localizing the faults. Many of these metrics have direct influence on the quality of software. More over prior knowledge of the fault proneness of a component may significantly reduce the testing effort and time. The measures of object oriented features like inheritance, polymorphism and encapsulation etc may be used to estimate fault proneness. Many researchers have investigated the usage of object oriented metrics in the software fault prediction. In this study we present taxonomy of usage these metrics in the fault prediction. We also present the analysis of machine learning techniques in fault prediction.

© 2018 The Authors. Published by Elsevier Ltd.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/3.0/)
Peer-review under responsibility of the scientific committee of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018).

Keywords: Software fault prediction; Object Oriented Testing; Object Oriented Coupling ; software faults prediction; machine learning

1877-0509 © 2018 The Authors. Published by Elsevier Ltd.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/3.0/)
Peer-review under responsibility of the scientific committee of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018).
10.1016/j.procs.2018.05.115

1. Introduction

The last two decades have witnessed the noticeable rise in the popularity of Object Oriented (OO) paradigm. Change is inevitable activity is software life cycle. The change may be required to resolve some fault or for perfective maintenance. The change in one component of OO systems may affect the other components due to inter-dependence relationship(s) among these components. More the interdependence of the software components more will be number of modules affected by a change in single module. Software coupling is the measure of such inter-dependence among various software components. There are specific features in OO systems like polymorphism, inheritance and encapsulation. These features proliferate this inter-dependence further and hence contribute more to the coupling. Various OO metrics have been evolved to estimate the coupling /complexity measure of the software. Also these software metrics have been rigorously used for software quality and reliability estimates. Researcher's fraternity has investigated the influence of OO complexity metrics on fault vulnerability of software components. Also many analysis tools have been witnessed in the literature. This work reports taxonomy of the research work related to prediction of software fault proneness on the basis of OO metrics. This work not only presents literature review of the work done but also various tools, techniques and data sets to perform experimentation. The work also highlights the application of machine learning algorithms in fault predictive analysis. The work is organized into following sections. The Section 2 depicts OO Metrics, 3 Analysis of Literature, 4 Comparative analysis, 5 Tools and Data Sets, 6 Conclusion and Future Scope.

2. Object Oriented (OO) Metrics

The researchers have proposed many metrics suites for OO software. These suites are used in multi-prospective contexts like quality indicators, complexity measure, fault proneness and reliability measure. The tables 1 given below describes the most frequently used OO metrics in the literature.

Table 1. Object oriented metrics

S.N	Chidamber & Kemerer metrics (CK) [1]	Tang & Chen metrics [2]	Henderson Metrics [3]	Li and Henry Metrics [4]	Li Metrics [5]	MOOD Metrics [6]	QMOOD Metrics [7]
1	Coupling Between Objects (CBO)	Number of Object/Memory Allocation (NOMA	Number of Attributes (NOA)	Data Abstracting Coupling (DAC)	Coupling Through Abstract Data Types (CTA)	Attribute Hiding Factor (AHF)	Number of Public Method`s (NPM)
2	Number of Children (NOC)	Average Method Complexity (AMC)	Number of Methods per Class (NOM)	Message Passing Coupling (MPC)	Coupling Through Message Passing (CTM)	Method Inheritance Factor (MIF)	Data Access Metric (DAM)
3	Depth of Inheritance tree (DIT)	_	_	Number of Methods NOM	_	Method Hiding Factor (MHF)	Measure of Aggregation (MOA)
4	Response for a Class (RFC)	_	_	SIZE1	_	Coupling Factor (COF)	Measure of Functional Abstraction (MFA)
5	Lack of Cohesion of Methods (LCOM)	_	_	SIZE2	_	Polymorphism Factor (POF)	Cohesion Among Methods of Class (CAM)
6	Weighted Methods Per Class (WMC)	_	_	_	_	Attribute Inheritance Factor (AIF)	_

The table tendered the most popular metrics suites proposed by different researchers. It should be noted these metrics might not be distinct. And metrics of one author might be related with others. The meaning of almost every

metric is self explanatory. Besides the metrics presented in the table above, there also exist other metrics like McCabe cyclomatic complexity [36], Briand [53] and Halstead's Software Science [37]. These are also of significant importance but their usage in fault prediction is less as compared to other metrics

3. Analysis of Existing Literature

Object oriented software development process imbibes the concept of objects and classes to closely associate the programming methodology to real world entities and concepts. Though the object oriented features ease the process of software development but that might cause some particular issues. And some OO specific faults may exist in such systems. The research evidences pertaining to it are presented in following text.

A technical report by Basili, Briand, and Melo [9] investigated the object oriented design metrics CK metrics[1]. The results of the research established the evidence of usefulness of CK metrics in predicting proneness of a class in early phase of Software development life cycle. Briand et al. [10] experimentally studied the correspondence between OO metrics and fault proneness of software. Authors applied multivariate logistic regression models. El Emam, Melo, and Machado [11] 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 [12] applied Principal Component Analysis and Multivariate Logistic Regression model to estimate the proneness of different software components. A prediction model based on Artificial Neural Networks(ANN) was proposed by Thwin and Quah [13]. The proposed work estimated the quality of software on the basis of total faults in a class and the lines modified in that class.
The work of Gyimothy, Ferenc, and Siket[14], studied the effectiveness of individual CK metrics in OO software. The study collected the bugs data for Bugzilla, which is open source software. Authors applied logistic regression (LR) to study dependency of CK metrics with faults present in software. Kanmani et al.[15] compared the performance of two neural network models for fault prediction based on OO metrics. Authors compared performance of Probabilistic Neural Network (PNN) and Back Propagation Neural (BPN) in fault prediction analysis. The study reported that PNN outperforms in term of robustness. Many studies have considered all the faults similar but software faults may have different criticality and severity. So when predicting these faults equal weights may not be justified. The study of Zhou and Leung [16] considered different severities of faults. The study applied logistic regression technique to study the influence of CK metrics. The study also study performance analysis of random forest, naive Bayes network, and NNge in prediction. The study found strong statistical association of OO with fault vulnerability of classes. The study also found that CK metrics reveals low severity faults better. Singh, Kaur, and Malhotra [21] also carried out the similar investigations. In their study authors considered CK metrics and compared the performance of different prediction models. The models studied are Decision Tree (DT), Artificial Neural Network (ANN) and LR. Performance analysis was carried out by Receiver Operating Characteristic analysis. The results obtained indicated that high low and medium severity faults are predicted with better accuracy. The study also found that DT and ANN models perform better. A fault prediction model based on Artificial Immune Recognition System (AIRS) was proposed by Catal and Diri [17]. Authors worked upon both CK metrics and Halstead and McCabe metrics to determine their effect on fault proneness. Authors used class level bug data. As per their investigations the performance of AIRS is better than J48 based approach.
Olague et al. [18] 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 [27] 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 [19] 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. Authors also advocated the usage of neuro-fuzzy approach integrated with statistical techniques to reveal the relationships between these metrics and other dependent

variables. The study used ANFIS (Adaptive NeuroFuzzy Inference System) model. The results of their study were not consistent.

It is a well established principle of software engineering that good software should have low coupling and high value of cohesiveness. Some authors have worked upon cohesion metrics for the fault prediction as follows. Marcus, Poshyvanyk, and Ferenc [20] used object oriented conceptual cohesion for fault prediction. The authors proposed Conceptual Cohesion of Classes (C3), as a new measure of cohesion. C3 is determined on the basis of textual coherence of methods. An information retrieval approach supported with Latent Semantic Indexing was applied to textual coherence. The experiments were performed on three open source subject programs. The study advocates the combination of structural metrics and cohesion metrics for better prediction. Zhou, Xu, and Leung [23] empirically evaluated the capability of complexity metrics in predicting software faults. The metrics used are CK metrics and McCabe metrics. The data from three different versions of eclipse IDE were used by the authors. The authors compared the performance of LR, Navie Bayes, AdTree, K Star and Neural networks. Their results indicate many metrics have moderate ability to differentiate fault prone and fault non-prone class. As per their findings lines of codes and weighted method, McCabe complexity are good indicators of fault proneness.

It not only the metrics selected that are important for fault prediction but size of data sets and feature extraction techniques are equally important. Recent trends in the field of software fault prediction show that popularity of machine learning algorithm is significantly increasing. Some studies are depicted as follow. Catal and Diri [22] empirically studied impact of metric set, dataset size and features selection techniques in fault prediction models. The authors applied random forest (RF) and AIRS. The authors concluded that RF algorithms performed better for large data sets and Naive Bayes algorithms worked well for small data sets. Alan [25] applied RF machine learning algorithm for detecting outliers. For their study six metrics from CK suite were chosen for the experiment. The study found that threshold based outlier detection is promising one and should be applied before developing fault prediction models.

The relationship between OO metrics and change proneness of class was studied by Malhotra and Khanna [30]. The study comprises of analysis of three open source software. The authors found that RFC metric is good predictor of vulnerability to a change. The study also found that RF and Bagging methods are better than LR method in estimating the change proneness. Their work, Malhotra and Khanna [31] ,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 [32] put thresholds on the object oriented metrics. The metrics worked upon are CK metrics.

Not only the experimental studies but also there are some descriptive studies like survey, review etc. Some of these depicted here. A systematic literature review by Catal and Diri [24] reported literature various studies of fault prediction in software. The study is based on 74 research articles. The paper highlighted the datasets, methods and trends in fault prediction. The of Catal [26] also presented a systematic literature review of software fault prediction highlighting current trends in this arena. The study explored 90 research articles published between 1999 and 2009. The research work examined the both machine learning and statistical approaches. As per their findings the supervised learning based prediction methods are extensively used in literature. A systematic literature review by Hall et al.[28] was presented in 2012. The study is based on 208 research articles for analyzing the models used to predict faults in source code. The time frame of the survey is between January 2000 and December 2010. Another survey was conducted by Radjenović et al. [29]. The survey is based on 106 research articles published between 1999 and 2011. As per their findings the usage of OO metrics in fault prediction is almost twice as compared to other source code based metrics. The study concluded that CK suite is most frequently used metrics amongst all OO metrics.

It is evident from the literature reviews that object oriented metrics are extensively being exploited to examine fault proneness and change proneness of some software component. And it's still an area of investigation.

4. Comparative analysis of Machine learning based studies.

The application of machine learning techniques in the software fault prediction is one of the recent areas of investigation. Authors are using the machine learning techniques in multi prospective ways in this domain. Some key applications include feature selection, classification, outlier detection and model building. The techniques applied by various researchers in literature differ in many contexts like nature of data applied on, metrics considered, machine learning algorithm used and the tools used for experimentation etc. Data from some of recent studies are depicted in table 2. The table 2.given below highlights the qualitative analysis of machine learning algorithms used for fault prediction.

Table 2. Comparative analysis

S.N	Study	Algorithm(s) Used	Metrics evaluated	Programming Language of SUT	Tool(s) used for Machine learning
1	[9]	Logistic Regression	CK[1]	C++	NA
2	[10]	Logistic Regression	Briand [53] and CK[1]	JAVA	NA
3	[12]	Logistic Regression	CK[1] and Li[5]	C++	NA
4	[13]	Ward neural network and General Regression neural network	CK[1]	NA	NA
5	[14]	Decision tree and Neural network	CK[1]	C++	NA
6	[16]	Logistic Regression	Briand [53]	C++	SPSS
7	[17]	Naive Bayes network, Random forest, and NNge	CK[1]	C++	WEKA
8	[18]	J48 and AIRS	CK[1]	NA	WEKA
9	[19]	Logistic Regression	CK[1],MOOD[6], QMOOD[7]	JAVA	NA
10	[20]	Logistic Regression and PCA	Conceptual Cohesion of Classes	C++	NA
11	[22]	Random Forest, J48	McCabe[36], Halstead[37]	C++	WEKA
12	[25]	Random Forest	CK[1]	JAVA	WEKA
13	[31]	Random Forest, Bagging Multilayer perceptron and Adaptive Boosting	CK[1]	JAVA	WEKA

From the table above it is evident that logistic regression, random forest and neural network are very popular in fault prediction studies. Machine learning algorithms can be used for various statistical and prediction analysis of OO metrics. And WEKA provides a platform for execution and analysis of these algorithms. A brief of regression model and other techniques are given below.

4.1 Logistic Regression (LR)

Logistic regression is a statistical classification technique which is based on maximum likelihood estimation. It can be used in two ways: univariate regression and multivariate regression. Univariate LR can be performed for the analysis of one metric on the fault proneness in isolation. The Multivariate LR is usable where more than one metrics are to be analysed for their effect on fault proneness. LR is used where there one or more than one independent variable. The goal of LR is construct best fitting model to describe relationship between dependent and independent variable. The outcome of the LR is fitted logistic regression equation.

$$\log\left(\frac{\pi}{\pi-1}\right) = C_0 + C_1X_1 + C_2X_2 + \ldots + C_nX_n \quad (1)$$

Where π is the probability that a class hold a fault and X_i's are the variable OO metrics and C_i's are the coefficients whose values are estimated using some likelihood functions. There are methods like Hosmer-Lemeshow test, Classification table, ROC curve analysis which are used to evaluate goodness of LR equations. The studies [9, 10, 11, 16, 18, 23, 27, 30, and 31] have analysed effect of OO metrics on fault proneness based on LR techniques.

4.2 Other algorithms.

Learning can be either supervised or unsupervised. A supervised learning consists of dependent variable which can be predicted from the given set of independent variables. Using the set of variables a map function is generated that produces the desired outcome. It is observed that research studies have exploited many machine learning algorithms to predict effect of OO metrics on fault proneness of the software. The study of [21] used Decision tree and validation is done using receiver operating characteristic (ROC) curve. The main advantage of Decision tree is the it can implicitly identify the most affecting features from the data set. Also its performance is not affected by the type of relationship between attributes. Machine learning algorithms like random forest are suitable for multiclass data. And Bayes networks are based of rules of probability of prediction. In some studies [13, 16, 21, 22, 25] set of learning algorithms like Bayes network, random forest, and NNge (nearest neighbor with generalization) are applied for comparative analysis of prediction models. Artificial Immune Recognition System(AIRS) is machine learning algorithm inspired from vertebrate immune system. It can work with nominal and continuous data. The study of [17] applied AIRS and found that its performance is better than J48. Principal component analysis (PCA) is also a feature selection technique. PCA is used to emphasize variation and to produce the strong patterns in data set. Some studies [11, 14], have used PCA for fault prediction and to select the feature OO metrics. The neuro-fuzzy [19] and Latent Semantic Indexing [20] are other competitive algorithms exploited for the prediction purpose.

5. Data sets and Analysis Tools

Data sets ease the task of experimental evaluation of proposed methodologies by the researchers. For the evaluation of OO metrics related studies, it is found that some data sets are very commonly used. Out of these some data sets are private data sets and some are public domain data sets. The most commonly used public data sets are given as under

a) **NASA Data sets** [32]: One of the most used data set for predictive analysis is metrics data program (MDP) provided by NASA. The data set consists of module level data of 13 systems. The metrics given are static in nature. It can be publically accessed. To extract data from this data set software tool like Weka can be easily integrated with. Some pre-processing of these data sets may be required to as per the context of application.
b) **PROMISE repository** [33]: It is also a public repository. It hosts the data pertaining to software fault detection, software cost estimation and requirements tracing etc.
c) **Bug prediction dataset** [34]: A public data set. It comprises collection of models and metrics of software systems and their histories. Data hosted there may be useful in running a fault prediction technique and to compute the performance of techniques.
d) **Qualitas corpus [35]:** Qualitas Corpus is a repository that contains data java programs. Also the programs are open source. It contains data related to nearly 100 systems with multi-versions and contains source and binary of each system. It also supports metrics comparison of same software artifact.

Software Analysis Tools: Table 3. Given under depicts the various code analysis tools that are frequently being used by the researchers. Most of the tools presented in the table belong to measuring of OO metrics. Also most of these tools support static analyses of the code i.e. code itself not required to be executed. Some tools are research tools developed by the authors like [39, 51, 52].

Table.3 Software Tools

S.N	Tool	Platform supported	Free/ Proprietary (P)?	Open Source?	Purpose	Metrics produced
1	JCMT [39]	JAVA	NA	NA	Static analysis	NA
2	JMT [40]	JAVA	Free	Y	Static analysis	DIT ,NOC WMC,WAC,CBO,PIM, NMO, RFC,LOC,NOP,LOC ,MIF ,AIF
3	CKJM [41]	JAVA	Free	Y	Static analysis	WMC,DIT,NOC,CBO,RFC,LCOM,Ca,NPM
4	Analyst4j [42]	java	free	Y	Static analysis	NA
5	CCCC[43]	C,C++, JAVA	Free	Y	Static analysis	WMC,DIT,NOC,CBO, RFC,LCOM,Ca,NPM
6	OOMeter [44]	JAVA, C#	NA	NA	Static analysis	Coupling and Cohesion metrics
7	Dependenc y Finder [45]	Java	free	Y	Static analysis	Basic Method, Basic Class, Basic Group, and Basic Project Measurements
8	JArchitect [46]	Java	P	N	Static Analysis, Model generation	different 82 code metrics
9	STAN [47]	Java	P	N	Static Analysis, Model generation	CK, McCabe's, Martin, LOC,ACD,FAT,Tangled
10	Source Meter[48]	C,C++, JAVA,C#	P	N	Static Analysis	60 different metrics
11	McCabe IQ tool[49]	Ada, C, C#, C++.NET, C++, JAVA, JSP, VB, VB.NET, COBOL, FORTRAN	P	N	Static and Dynamic Analysis	McCabe metrics
12	Understand for Java[50]	Ada,C,C++,CO BOL, JAVA,Pascal,D ephi,Web,	P	N	Model generation, Static analysis	Project Level, ClassLevel, Class OO Metrics,Program Unit Metrics, FileMetrics,File Average Metrics
13	JaBUTi [52]	JAVA	Free	Y	Static analysis	CK metrics
14	Author's tool[51]	JAVA	NA	NA	Static Analysis	Different 23 OO metrics
15	Columbus[54]	C++	NA	NA	NA	NA
16	SonarGraph[55]	Java, C# and C/C++	P	N	Static Analysis	Different 35 metrics

6. Conclusions and Future Research Avenues

This research article presents taxonomy of usage of OO metrics for fault proneness prediction. The work highlights various OO metrics used in the literature. These metrics are of vital significance in determining the quality of the software. These metrics may be helpful in reducing the efforts needed in the software maintenance phase of software life cycle. The work also propound that various machine learning algorithms have applied in fault prediction. But still there are sub-domains of these algorithms like Support Vector Machine, Dimensionality Reduction, Gradient Boosting and Deep Learning etc that can be worked upon. Also most of the OO metrics related research has been for fault prediction. Also these metrics may attribute to other activities of testing phase like test case selection, generation, prioritization and clone detection. The research pertaining to fault prediction may be extended to these fields as well. The research may also be initiated for development of tools for extracting OO metrics from the software. This paper also provides a set of data sets that have been extensively used by researchers to evaluate their

techniques/methodologies. Most of these are public domain and have very good collection of testing data. Also there exist software tools to ease the task of code analysis. In the work reported, we provided a set of such tools with their description. Most of these tools are free and support the most of the OO metrics. We considered both software tools and author designed tools. The prior knowledge of fault proneness and change proneness of a software component may ease the testing process. The processes such as fault localization, refactoring, debugging and test case minimization may be handled with more efficiency if testing is coupled with predictive measures. So the findings of this study may be extended to cost minimization of software maintenance. We are currently working on a research problem pertaining to object oriented software testing. The knowledge of fault prediction on the basis of OO metrics would be helpful to more accurate examination of problem. We are looking forward to investigate impact of OO metrics on software maintenance.

References

[1] Chidamber SR, Kemerer CF. Towards a metrics suite for object oriented design. ACM; 1991 Nov 1.

[2] Tang MH, Kao MH, Chen MH. An empirical study on object-oriented metrics. InSoftware Metrics Symposium, 1999. Proceedings. Sixth International 1999 (pp. 242-249). IEEE.

[3] Henderson-Sellers B. Object-oriented metrics: measures of complexity. Prentice-Hall, Inc.; 1995 Dec 1.

[4] Li W, Henry S. Object-oriented metrics that predict maintainability. Journal of systems and software. 1993 Nov 1;23(2):111-22.

[5] Li W. Another metric suite for object-oriented programming. Journal of Systems and Software. 1998 Dec 1;44(2):155-62.

[6] Abreu FB, Carapuça R. Object-oriented software engineering: Measuring and controlling the development process. InProceedings of the 4th international conference on software quality 1994 Oct 3 (Vol. 186, pp. 1-8).

[7] Bansiya J, Davis CG. A hierarchical model for object-oriented design quality assessment. IEEE Transactions on software engineering. 2002 Jan;28(1):4-17.

[8] Martin RC. Agile software development: principles, patterns, and practices. Prentice Hall; 2002.

[9] Basili VR, Briand LC, Melo WL. A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering. 1996 Oct;22(10):751-61.

[10] Briand LC, Wüst J, Daly JW, Porter DV. Exploring the relationships between design measures and software quality in object-oriented systems. Journal of systems and software. 2000 May 1;51(3):245-73.

[11] El Emam K, Melo W, Machado JC. The prediction of faulty classes using object-oriented design metrics. Journal of Systems and Software. 2001 Feb 1;56(1):63-75.

[12] Fioravanti F, Nesi P. A study on fault-proneness detection of object-oriented systems. InSoftware Maintenance and Reengineering, 2001. Fifth European Conference on 2001 (pp. 121-130). IEEE.

[13] Thwin MM, Quah TS. Application of neural networks for software quality prediction using object-oriented metrics. Journal of systems and software. 2005 May 1;76(2):147-56.

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

[15] Kanmani S, Uthariaraj VR, Sankaranarayanan V, Thambidurai P. Object-oriented software fault prediction using neural networks. Information and software technology. 2007 May 1;49(5):483-92.

[16] Zhou Y, Leung H. Empirical analysis of object-oriented design metrics for predicting high and low severity faults. IEEE Transactions on software engineering. 2006 Oct;32(10):771-89.

[17] Catal C, Diri B. Software fault prediction with object-oriented metrics based artificial immune recognition system. In International Conference on Product Focused Software Process Improvement 2007 Jul 2 (pp. 300-314). Springer, Berlin, Heidelberg.

[18] Olague HM, Etzkorn LH, Gholston S, Quattlebaum S. 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. 2007 Jun;33(6):402-19.

[19] Xu J, Ho D, Capretz LF. An empirical validation of object-oriented design metrics for fault prediction.

[20] Marcus A, Poshyvanyk D, Ferenc R. Using the conceptual cohesion of classes for fault prediction in object-oriented systems. IEEE Transactions on Software Engineering. 2008 Mar;34(2):287-300.

[21] Singh Y, Kaur A, Malhotra R. Empirical validation of object-oriented metrics for predicting fault proneness models. Software quality journal. 2010 Mar 1;18(1):3.

[22] Catal C, Diri B. Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem. Information Sciences. 2009 Mar 29;179(8):1040-58.

[23] Zhou Y, Xu B, Leung H. On the ability of complexity metrics to predict fault-prone classes in object-oriented systems. Journal of Systems and Software. 2010 Apr 1;83(4):660-74.

[24] Catal C, Diri B. A systematic review of software fault prediction studies. Expert systems with applications. 2009 May 1;36(4):7346-54.

[25] Alan O, Catal PD. An outlier detection algorithm based on object-oriented metrics thresholds. InComputer and Information Sciences, 2009. ISCIS 2009. 24th International Symposium on 2009 Sep 14 (pp. 567-570). IEEE.

[26] Catal C. Software fault prediction: A literature review and current trends. Expert systems with applications. 2011 Apr 1;38(4):4626-36.

[27] Elish MO, Al-Yafei AH, Al-Mulhem M. Empirical comparison of three metrics suites for fault prediction in packages of object-oriented systems: A case study of Eclipse. Advances in Engineering Software. 2011 Oct 1;42(10):852-9.

[28] Hall T, Beecham S, Bowes D, Gray D, Counsell S. A systematic literature review on fault prediction performance in software engineering. IEEE Transactions on Software Engineering. 2012 Nov;38(6):1276-304.

[29] Radjenović D, Heričko M, Torkar R, Živkovič A. Software fault prediction metrics: A systematic literature review. Information and Software Technology. 2013 Aug 1;55(8):1397-418.

[30] Malhotra R, Khanna M. Investigation of relationship between object-oriented metrics and change proneness. International Journal of Machine Learning and Cybernetics. 2013 Aug 1;4(4):273-86.

[31] Malhotra R, Khanna M. Mining the impact of object oriented metrics for change prediction using Machine Learning and Search-based techniques. InAdvances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on 2015 Aug 10 (pp. 228-234). IEEE.

[32] NASA MDP: http://mdp.ivv.nasa.gov

[33] PROMISE Resipository: http://promise.site.uottawa.ca/SERepository/datasets-page.html

[34] Bug prediction dataset: http://bug.inf.usi.ch/index.php

[35] Tempero E, Anslow C, Dietrich J, Han T, Li J, Lumpe M, Melton H, Noble J. The Qualitas Corpus: A curated collection of Java code for empirical studies. InSoftware Engineering Conference (APSEC), 2010 17th Asia Pacific 2010 Nov 30 (pp. 336-345). IEEE.

[36] McCabe TJ. A complexity measure. IEEE Transactions on software Engineering. 1976 Dec(4):308-20.

[37] Halstead MH. Elements of software science.

[38] Henry S, Kafura D. Software structure metrics based on information flow. IEEE transactions on Software Engineering. 1981 Sep(5):510-8.

[39] Bidve VS, Sarasu P. Tool for measuring coupling in object-oriented java software. International Journal of Engineering and Technology. 2016;8(2):812-20.

[40] https://www2.informatik.huberlin.de/swt/intkoop/jcse/tools/jmt.html

[41] https://www.spinellis.gr/sw/ckjm/doc/indexw.html

[42] https://www.javalobby.org//java/forums/t93556.html

[43]https://sourceforge.net/projects/cccc/

[44] Alghamdi JS, Rufai RA, Khan SM. OOMeter: A software quality assurance tool. InSoftware Maintenance and Reengineering, 2005. CSMR 2005. Ninth European Conference on 2005 Mar 21 (pp. 190-191). IEEE.

[45] http://depfind.sourceforge.net/

[46] https://www.jarchitect.com

[47] http://stan4j.com/

[48] https://www.sourcemeter.com/

[49] http://www.mccabe.com/iq.htm

[50] https://scitools.com/features

[51] AlGhamdi, Jarallah, Mahmoud Elish, and Moataz Ahmed. "A tool for measuring inheritance coupling in object-oriented systems." *information SCiences* 140.3-4 (2002): 217-227.

[52] Vincenzi, A. M. R., et al. "JaBUTi: A coverage analysis tool for Java programs." *XVII SBES–Simpósio Brasileiro de Engenharia de Software* (2003): 79-84.

[53] Briand, Lionel, Prem Devanbu, and Walcelio Melo. "An investigation into coupling measures for C++." *Proceedings of the 19th international conference on Software engineering*. ACM, 1997.

[54] Ferenc R, Beszédes Á. Data exchange with the Columbus schema for C++. In Software Maintenance and Reengineering, 2002. Proceedings. Sixth European Conference on 2002 (pp. 59-66). IEEE

[55] https://www.hello2morrow.com/products/sonargraph/explorer