Dynamic Metrics are Superior than Static Metrics in Maintainability Prediction : An Empirical Case Study

Hemlata Sharma[#], Anuradha Chug[*]

[#]*IT, Lal Bahadur Shastri Institute of Management, New Delhi, India*

[1]hemlatasharma@lbsim.ac.in

[*]*USICT, GGS IP University, New Delhi, India*

[2]anuradha@ipu.ac.in

Abstract— Software metrics help us to make meaningful estimates for software products and guide us in taking managerial and technical decisions like budget planning, cost estimation, quality assurance testing, software debugging, software performance optimization, and optimal personnel task assignments. Many design metrics have proposed in literature to measure various constructs of Object Oriented (OO) paradigm such as class, coupling, cohesion, inheritance, information hiding and polymorphism and use them further in determining the various aspects of software quality. However, the use of conventional static metrics have found to be inadequate for modern OO software due to the presence of run time polymorphism, templates class, template methods, dynamic binding and some code left unexecuted due to specific input conditions. This gap gave a cue to focus on the use of dynamic metrics instead of traditional static metrics to capture the software characteristics and further deploy them for maintainability predictions. As the dynamic metrics are more precise in capturing the execution behavior of the software system, in the current empirical investigation with the use of open source code, we validate and verify the superiority of dynamic metrics over static metrics. Four machine learning models are used for making the prediction model while training is performed simultaneously using static as well as dynamic metric suite. The results are analyzed using prevalent prediction accuracy measures which indicate that predictive capability of dynamic metrics is more concise than static metrics irrespective of any machine learning prediction model. Results of this would be helpful to practitioners as they can use the dynamic metrics in maintainability prediction in order to achieve precise planning of resource allocation.

Keywords— Static metrics, Dynamic metrics, Software maintainability prediction, Software quality, Machine learning.

1 INTRODUCTION

The objective of software engineering is to develop cost effective and quality oriented small, medium and large scale software products. A software metric is a quantitative measure of the degree to which a software system possesses properties like coupling, cohesion, inheritance, abstraction to name a few. Since quantitative measurements are essential in all sciences, computer science researchers and practitioners are putting all their efforts in measuring the quantitative information of the software components using number of metrics. The objective is to have reproducible and quantifiable measurements with numerous valuable applications such as schedule and budget planning, cost estimation, quality assurance testing, software debugging, software performance optimization, and optimal personnel task assignments.

Software maintenance is described as the process of modification of existing software product after delivery to correct faults and to improve performance ([1], [2], and [3]). Software maintainability is defined as the ease with which these modifications could be made into the existing software system. It has a significant impact on the total cost of software system as it is observed that maintenance activity consume almost 70% of the total project cost. Many researchers ([1], [2], [3], and [4]) worked on various constructs of metrics like coupling, cohesion, inheritance etc to identify the OO design quality. However, all these studies talks about the use of static metrics for capturing the design characteristics of the code. Due to the presence of dynamic behavior of the software, some part of the code might not get executed depending upon the inputs given by the user. These erroneous measurements of capturing the OO characteristics which are actually not executed, lead to the problem of inaccurate training from the existing data using machine learning algorithms which consequently results in imprecise predictions. In this research paper we have proposed the use of dynamic metrics instead of static metrics for capturing the OO characteristics and using them for maintainability predictions. In this regard, the following four research questions were addressed:

RQ1: Are static metrics same as dynamic metrics?
RQ2: Is the performance of dynamic metrics better than static metrics for maintainability predictions?
RQ3: Can dynamic metrics be used for maintenance predication?
RQ4: Which machine learning algorithm (linear regression, multilayer perceptron, gaussian process and SMOreg) is better for accurate prediction?

The rest of the paper is structured as follows: section 2 presents literature review. The relevance of dynamic metrics is discussed in section 3. Section 4 presents the research methodology adopted in the current study. The results and analysis are shown in section 5. Section 6 presents the threats to validity and finally conclusion and future directions are presented in sections 7.

978-1-4673-7231-2/15/$31.00 ©2015 IEEE

2. LITERATURE REVIEW

Many empirical studies have conducted to validate the strong relationship between design metrics and respective maintainability. These studies are useful as it helps in predicting the maintenance behavior of the software in the early phases of the software project development. Malhotra and Chug [4] have deployed static metrics and found that the performance of Group Method for Data Handling (GMDH) was concise in terms of the prediction accuracy and can be considered for maintainability prediction. Agarwal et al. [5] proposed an empirical study to identify the capability of each metric in capturing the distinctive characteristics of the software. The quality assessment of OO designs using dynamic metrics consisting of dynamic complexity and object coupling were measured on the basis of the execution scenarios by Yacoub et al. [6]. In their study, dynamic coupling metrics was based on the number of responses and requests sent and received from one object to another by incorporating the scenario profiles and identifying the transitions between composite states. Gupta [7] evaluated dynamic coupling metrics for OO software by using formal evaluation framework proposed by Briand et al. [8]. Result of their study shows that the dynamic coupling metrics satisfy all properties and parameters required for the evaluation framework. Gupta [9] proposed various methods which provide more desirable support for dynamic analysis of programs as they are based on dynamic modeling, profiles and aspect oriented approach. He also conclude that aspect oriented approach is efficient and easier to implement. Amjed and MacDonell [10] focused on research issues of dynamic software metrics and concluded that the mapping reviews aim to identify and characterize all research works that are related to a specific topic, using a defined and defendable search strategy. Several dimensions of dynamic coupling by tracing the flow of messages between objects at run-time was shown by Arisholm [11]. A convenient method was used to describe these dimensions through the concept of role-models proposed by Reenskaug et al. [12]. Kaur et al. [13] verified that complexity of a code is directly dependent on its understandability using dynamic set of metrics. Young [14] defined the degree to which a system's design is related to the level of difficulty for understanding and verifying. Ohata et al. [15] proposed a program slicing method using JAVA containing two components i.e. analysis libraries and graphical user interface (GUI). He also defined object oriented dependence cache (OODC) slicing which is an intermediate between static and dynamic slicing and observed that the size of OODC is 20 to 70% as large as that of static slice and inferred that OODC slice is more precise than static slice. An empirical validation of the metrics by computing the defined metrics for a representative set of well-known Java benchmarks, and by evaluating the metrics in terms of how well they corresponded to the commonly accepted qualitative appraisal of the benchmarks was done by Dufour et al. [16]. Allier et al. [17] concluded that the classical methods underestimate or overestimate the coupling for systems where dynamic features are used. Cook et al. [18], presented dynamic as well as static view points on software evaluation. Dynamic view point investigates how to model software evaluation trends and the static view point studies the characteristics of software artifacts to see what makes software more evolvable at the design and source code level. Although many authors have explored the utility of dynamic metrics, none of the study actually compare the performance of static metrics and dynamic metrics on same dataset with different machine learning model. Dynamic analysis is considered as "input-centric" analysis, whereas static analysis maybe termed as "program-centric" analysis. Dynamic analysis is more precise and accurate as compared to static metrics. In the current study using the same dataset for both set of metrics, exact comparison is drawn.

3. RELEVANCE OF DYNAMIC METRICS

The metrics can be classified into different categories like metrics for analysis model, metrics for design model, metrics for source code, metrics for testing, metrics for quality assurances etc. A metric should possess features like simple & computable, consistent, platform autonomous, objective, empirically and intuitively credible as stated by Pressman [19]. OO metrics helps a programmer / developer to understand and unravel the real life problems by utilizing the characteristics of OO programming. It is widely accepted that OO software development follows a different thought process than the traditional structured software development. The main advantages of OO design lies in its modularity and reuse of code. Thus the metrics, that were useful in evaluating structural development, may not be effective in evaluating OO software development. In this paper, dynamic analysis approach is adopted which deals with code/ design used at the time of execution. They are compared with static metrics as they addresses the source code irrespective of the code gets executed or not. These two metrics presents a complimentary evaluation mechanism in several ways. Dynamic analysis has the benefit of examining the concrete domain of program execution while static analysis must provide abstract over this domain in order to ensure termination of the analysis, thus losing information from the start.

4. RESEARCH METHODOLOGY

The methodology adopted in the current study is presented in Fig 1. In order to statistically test and evaluate the hypotheses, an open source software system Hodoku was used. For the same source code, static as well as dynamic metrics were collected and considered as two populations.

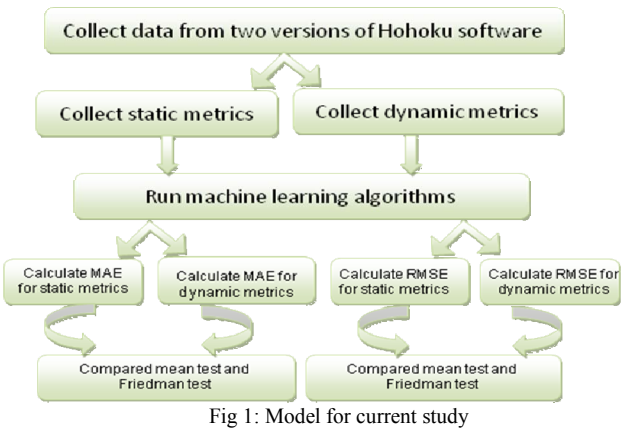

Fig 1: Model for current study

The goal and hypothesis formulation are discussed in section A. Section B presents the empirical data collection process. Section C presents the machine learning algorithms deployed in the current study for classification as well as prediction purpose. Section D discusses various prediction accuracy measures to adjudge the performance of static as well as dynamic metrics suite.

A. Hypotheses

The study presents two hypotheses for analysis. The first hypothesis states that there is a significant difference between the prediction performance of static metrics and dynamic metrics. The second states that the performance of dynamic metrics is better than static metrics. Our research hypotheses are labeled as H11and H12. Their respective null hypotheses are labeled H01 and H02.

H11: There is a significant difference between the predictions performance of static metrics and dynamic metrics.

H01: No significant difference exists between the predictions performance of static and dynamic metrics.

H12: The performance of dynamic metrics is better than static metric.

H02: The performance of dynamic metrics is not better than static metric.

For the first research hypothesis, regarding the significant difference between static and dynamic metrics, the comparison was done by mean paired t-test. We consider an empirical study component as a "success", represented as p1 or p2, for each population. Thus our first research and null hypotheses are:

H11: p2 > p1 H01: p2 = p1 Note the scenario p2 < p1 is included in H11. We selected a level of significance of 0.05 common for this type of studies. For the second research hypothesis, we would use the Chi- squared goodness of fit test was used, and ranking was done by Friedman test. Friedman test is a non-parametric test used to detect difference in multiple test attempts and involves ranking of rows followed by columns.

B. Empirical Data Collection

In this current study, we have used two versions of Java project named as Hodoku 1.1 and Hodoku 2.2. Subsequently with the help of tools, static and dynamic metrics were extracted. These metrics were utilized for training and testing purpose of data set on various machine learning algorithms. The list of static metrics and dynamic metrics along with their definition in brief are presented in Table 1 and Table 2 respectively.

TABLE 1. LIST OF STATIC METRICS

Metric	Description
WMC (Methods per Class)	The sum of McCabes's cyclomatic complexities of all local methods in a class.
DIT (Depth of Inheritance Tree)	The metric measures class level in the inheritance tree, root class is considered as zero.
NOC (Number of Children)	It counts number of immediate sub classes of a class in a hierarchy.
CBO (Coupling between Objects)	It represents the number of classes to which the given class is coupled.
RFC (Response For a Class)	The number of local methods plus the number of non local methods called by local methods.
LCOM (Lack of Cohesion of Methods)	The number of disjoint sets of local methods. Each method in a disjoint set shares at least one instance variable with at least one member of the same set.
LOC (Lines of code)	The number of lines of code excluding comments.
AMC (Average Method Complexity)	This metric measures the average method size for each class.
Change	The total number of lines Added, Deleted and Modified in a class.
Ca(Afferent couplings)	A class's afferent couplings is a measure of how many other classes use the specific class
Ce(Efferent couplings)	A class's efferent coupling is a measure of how many other classes is used by the specific class.
NPM (Number of Public Methods)	All methods which have public access specifiers are counted by NPM.
DAM (Data Access Metric)	The metric computes the ratio of attributes declared as private or protected to the total number of attributes declared in the class.
MOA (Measure of Aggregation)	It measures the HAS-A relationship between attributes at run time.
MFA (Measure of Functional Abstraction)	The metrics computes the count of the number of inherited methods of a class divide by the total number of methods which are accessible by member methods of the class.
CAM (Cohesion Among Methods of Class)	The relevance between the class methods based on the list of specifications of the methods is computed by this metric.
IC(Inheritance Coupling)	This metric produces the total number of super classes to which a given class is coupled.
CBM(Coupling Between Methods)	The metric measure the total count of new or redefined methods to which all the inherited methods are coupled.
AMC(Average Method complexity)	The average method size for each class is measured by AMC, where the size of a method is equivalent to the number of java binary codes in the method.

TABLE 2. LIST OF DYNAMIC MATRICS

Metric	Description
LCC (Loose Class Coupling)	This metric calculates the low dependency between object-structure at run-time.
LCOM (Lack Of Cohesion in Methods)	It is used to measure the lack of cohesion between methods at run time. This group of metrics aims to detect problem classes. A high LCOM value means low cohesion. This varies from 0 to 4.
NAK (Number of Assertions per KLOC)	Counts the number of declarations per KLOC at run time.
NOC (Number of Children)	The metric find out the total number of direct subclasses of a class at run time.
NOF (Number Of Fields)	This metric counts the number of data declarations used at run time.
NOM (Number Of Methods)	Total number of objects interacted at run tome
NOSF (Number of Static Fields)	It measures the total number of static attributes in the selected scope.
NOSM (Number of Static Methods)	This metric is the sum of methods that are static at run time.
NTM (Number of Test Methods)	Total sum of test methods that are encountered at run time is calculated by this metric.
TCC (Tight Class Coupling)	It measures the tight class coupling is the high dependency between object-structure at run time
WMC (Weighted Method Count)	It counts run time method complexity. A high WMC reduces the reuse probability for the class.
LOC (Lines Of Code)	This metric find out the total lines of code encountered at run time.
LOCm (Lines Of Comments)	Total number of lines of comments used at runtime.
Change	The total number of lines Added, Deleted and Modified in a class.

The descriptive statistics of static metrics and dynamic metrics are presented in Table 3 and Table 4 respectively followed by the interpretations. In the class-level metrics

data there are 18 static metrics and 17 dynamic metrics collected from all the classes individually and considered as independent variables. 'Change' is the dependent variable which is counted as the number of lines of source code added, deleted or modified in two consecutive versions of the same source code. Add is counted as one change, delete is counted as one change whereas for update is count as two changes.

TABLE 3. DESCRIPTIVE STATISTICS FOR STATIC METRICS

	Minimum	Maximum	Mean	Std.Deviation
WMC	2	165	27.43	38.052
DIT	0	6	3.23	2.133
NOC	0	1	.02	.151
CBO	1	93	15.95	16.575
RFC	4	548	88.25	99.323
LCOM	0	13032	910.68	2620.677
Ca	1	78	12.14	14.949
Ce	0	92	9.23	15.199
NPM	1	163	15.75	29.909
LCOM3	.00	2.0000	.810609	.3418614
LOC	13	6541	1271.66	1652.446
DAM	.00	1.0000	.743059	.4025629
MOA	0	104	4.77	15.798
MFA	.00	.9956	.608327	.4538965
CAM	.1000	.7500	.292427	.1531465
IC	0	2	.27	.585
CBM	0	15	.59	2.306
AMC	4.3333	207.1111	52.857680	51.8245686
Change	0	2716	256.07	545.435

TABLE 4. DESCRIPTIVE STATISTICS FOR DYNAMIC METRICS

	Minimum	Maximum	Mean	Std.Deviation
LCC	.0000	1.0000	.527456	.3198893
LCOM1	0	13262	684.59	2227.093
LCOM2	0	13158	613.89	2160.506
LCOM3	0	75	8.57	14.204
LCOM4	0	74	7.80	13.248
LCOM5	.0000	1.0947	.731398	.3198304
NAK	0	0	.00	.000
NOC	0	1	.02	.151
NOF	0	155	22.61	34.040
NOM	0	164	22.36	32.641
NOSF	0	103	6.59	19.434
NOSM	0	19	1.48	3.344
NTM	0	0	.00	.000
TCC	.0000	1.0000	.320577	.2635513
WMC	0	439	55.39	86.835
LOC	8	2235	415.73	510.934
LOCm	0	215	28.23	43.750
Change	0	2716	256.07	545.435

From these descriptive statistics the many observations were recorded. Some of the notables are mentioned below:

(i) For NOC, median and mean values obtained are found to be Minimum for both the system, so we draw conclusion that the use of number of children in both systems is limited.
(ii) The values for median and mean for change (dependent variable) in both systems will remain same, as it is a single data set.
(iii) NTM and NKM from dynamic metrics show that there is no existence of number of test methods and number of assertions in the dataset.
(iv) We have observed that the LOC of dynamic metrics is almost 3 times lesser then that of LOC of static metrics.
(v) WMC of dynamic metrics is almost two times more than that of WMC static metrics. This shows that cyclometic complexity of dynamic metrics is almost double as compared to static metrics.

C. USE OF MACHINE LEARNING TECHNIQUES

Table 5 presents the summary of the four machine learning techniques used in the current study. Some of the classification techniques are implemented using Weka 3.7[1] tool. We divided our data into 3:1 ratio, where three parts are used for training and rest is for testing the dataset.

TABLE 5. DESCRIPTION OF CLASSIFICATION ALGORITHMS

Classification Algorithm	Defination	Refer ence
Linear Regression	Linear Regression involves finding the 'best' line to fit two attributes, so that one attribute can be used to predict the other.	[20]
Multilayer Perception	MLP is a feed forward neural network.It iteratively learns a set of weights for prediction of the class label of tuples.	[20]
Gaussian Process	Gaussian process is a set of random variables any finite subset of which have a normal distribution.	[21]
SMOreg	SMOreg is an algorithm that implements sequential minimal optimization using extreme chunking by converting nominal attributes into binary ones and optimizing the target function for them.	[22]

D. Use of Prediction Accuracy Measures

Prediction accuracy measures are used to evaluate the performance of machine learning based system. Machine learning models have been used in the study for training by simultaneously deploying it with static as well as dynamic metric suite. Predicted value of 'change' is compared with actual value of 'change' to find out the error produced in each case. Mean absolute error (MAE) and root mean square error (RMSE) proposed by kitchenham [23] are used as a measure of prediction accuracy, lesser error value leads to higher accuracy. Formulas for these measures are as follows:

$$MAE = \sum_{i=1}^{n} |e_i| \Big/ N \qquad RMSE = \sqrt{\sum_{i=1}^{n} |e_i| \Big/ N}$$

Where N is the total number of iterations and e is the error i.e. difference between the actual and predicted values of dependent variable change.

5. RESULTS AND ANALYSIS

We have taken two different versions of OO software (Hodoku) and calculated change. Static metrics were calculated by using CKJM[2] tool. Dynamic metrics was collected by using NetBeans[3] . These two metrics were then analyzed on various machine learning algorithms. Table 6 shows the compiled results obtained for MAE and RMSE with static metrics as well as dynamic metrics for all classification machine learning algorithms. The results clearly indicate that the dynamic metrics produce more

[1] http://www.cs.waikato.ac.nz/ml/weka/downloading.html
[2] http://sourceforge.net/projects/ckjm/
[3] http://plugins.netbeans.org/plugin/42970/sourcecodemetrics

accurate results than static metrics in the maintainability prediction irrespective of the used machine learning algorithms. The Table 6 shows the MAE and RMSE for static metrics and dynamic metrics on various classification algorithms. Fig 2 and Fig 3 represents the graphical interpretation of the result. It is quite evident from the graph that with all the four classification ML algorithms, dynamic metrics was found to be more accurate.

TABLE 6. RESULTS OBTAINED FOR MAE AND RMSE ON VARIOUS CLASSIFICATION ALGORITHMS

Classification Algorithms	MAE with Static metrics	MAE with Dynamic Metrics	RMSE with Static metrics	RMSE with Dynamic metrics
Linear Regression	84.3032	0	114.4666	0
MultilayerPerception	50.9605	19.6708	76.6499	23.2465
Gaussian Process	146.71	94.183	221.7978	171.6427
SMOreg	77.5841	2.7019	150.9276	2.8597

Further, results also indicate that linear regression is found to be the best suitable algorithm for maintainability predictions using dynamic metrics.

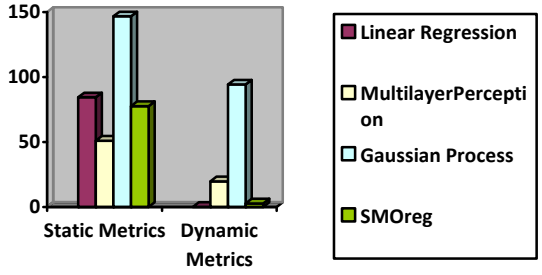

Fig.2. MAE of static metrics and dynamic metrics

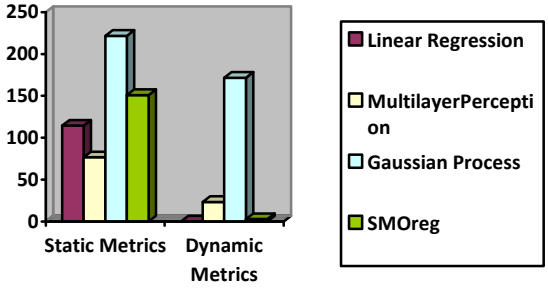

Fig. 3 RMSE values of static metrics and dynamic metrics

To check the hypothesis H11 and H01 being same or significantly different, the compare mean test was performed as compiled in Table 7. As the value of p is observed as less than 0.05, it indicates that there exists significant difference between the performance of static metrics and dynamic metrics. For performance rank test check, Friedman test was used to find the result on the second hypothesis H12 and H02 .The mean rank obtained by Friedman test with respect to MAE and RMSE was 1 for static metrics and 2 for dynamic metrics. The efficiency ratio for both the sets was calculated by dividing the value of prediction accuracy for static metrics to the dynamic metrics and compiled in Table 8. When static metrics were replaced by dynamic metric on the same dataset, we found that the efficiency ratio was increased from 1% to 8.43% for MAE and 1% to 11.44% for RMSE, which is highest in case of linear regression as evident from Table 8.

TABLE 7. RESULTS FOR COMPARE MEAN TEST

Pair of Static and Dynamic Metrics		Paired Differences 95% Confidence Interval Lower	Paired Differences 95% Confidence Interval Upper	t	DF	Sig. (2-tailed)
1.	St_MAE - Dy_MAE*	22.98	98.52	5.12	3	0.01
2.	St_RMSE- Dy_RMSE#	15.28	167.77	3.82	3	0.03

*St_MAE means the value of MAE on static metric and Dy_MAE means the value of MAE on dynamic metrics.
#St_RMSE means the value of RMSE on static metric and Dy_RMSE means the value of RMSE on] dynamic metrics.
DF: degree of freedom

TABLE 8. EFFICIENCY RATIO

Classification algorithms	Eff_Ratio_MAE	Eff_Ratio_RMSE
Linear Regression	84.3032	114.4666
MultilayerPerception	2.590667	3.297266
Gaussian Process	1.557712	1.292206
SMOreg	28.71465	52.77742

RQ1: Are static metrics same as dynamic metrics?
Answer: No, there was difference found in the performance of static metrics and dynamic metrics as far as the prediction of maintainability is concerned. When the results were compared with respect to MAE and RMSE, it was found that dynamic metrics outperformed static metrics in terms of the prediction accuracy measures.
RQ2: Is the performance of dynamic metrics better than static metrics?
Answer: We also compared the performance of both the metric suite using paired mean t-test and compiled the results in Table 7. The assessment of p–value has less than 0.05 as shown in Table 7; hence we concluded that there exist significant difference between the static metrics and dynamic metrics. Additionally, Friedman test was also conducted by which the performance of dynamic metrics was tested. We found that it has assigned rank 1 for dynamic metrics and rank 2 for static metrics in respect of MAE and RMSE. Hence, we conclude that dynamic metrics performs significantly better than static metrics.
RQ3: Can dynamic metrics be used for maintenance predication?
Answer: Yes, as evident from Table 6, the rate of error is very low in the case of dynamic metrics whereas very high with static metrics. It was concluded that dynamic metrics could be comfortable deployed for maintenance prediction.
RQ4: Which Machine learning algorithm (linear regression, multilayer perceptron, gaussian process and SMOreg) is better suited for accurate calculation?
Answer: We found that linear regression could be considered as best algorithm among all four because this algorithm has shown minimum error rate. The efficiency ratio was also found be on higher in the case of linear regression (Table 8).

6 . THREATS TO VALIDITY

Like other empirical studies, limitations confronted during the current study are given as below:
(i) Datasets considered in the present study were written in JAVA language. Hence this study is likely to be valid for

the code written in other OO programming languages, for example C++, .NET. However, further research can only establish their usefulness in predicting the maintainability of other development paradigms.

(ii) The data set used is small in terms of number of instances. The impact of large dataset is yet to be established by further research.

(iii)Learning algorithms like linear regression, gaussian process, SMOreg and multilayer perceptron have been applied. However, few other algorithms which have gained importance in recent times (due to their effectiveness) like random forest, decision tree and naïve bays network are yet to be applied.

7. CONCLUSION AND FUTURE WORK

The objective of the current study was to analyze the effectiveness of dynamic metrics for maintainability prediction. In order to assess the capability of dynamic metrics in capturing the OO characteristics of the software, an open source code Hodoku was used. Four machine learning algorithms Linear Regression, Multilayer Perceptron, Gaussian Process and SMOreg were used in developing the prediction model. Independent variables were considered in two sets, one with dynamic metrics suite and another with static metrics suite for easy comparisons however number of changes in two consecutive versions is taken as dependent variable. We further compared the results of prediction using static metrics and dynamic metric deployed with all machine learning algorithms using prevalent prediction accuracy measures such as mean absolute error (MAE) and root mean square error (RMSE). The conclusions of the study are summarized as below:

1. The results show that dynamic metrics out performs static metrics suite irrespective of any machine learning algorithms used for predictions.
2. Among all the four machine learning models deployed in the current study linear regression was found to be most accurate in predictions.
3. As the values of MAE and RMSE are in the acceptable range, we also conclude that dynamic metrics can be used successfully for maintainability predictions.

In future, we are planning to conduct number of studies on medium and large datasets to analyze the effectiveness of dynamic metrics and obtaining more generalized results. This would help researchers and software practitioners to derive conclusive results over a wide array of software data sets. Moreover, we are also planning to use evolutionary algorithms in developing prediction models. Current study is empirically conducted on java; we are also planning to conduct same study on other OO paradigm such as C++, C#, VB. Net, Ada 95 etc.

REFERENCES

[1] R. Malhotra, A. Chug, "*Software Maintainability Prediction using Machine Learning Algorithms*", International Journal of S Software Engineering, vol 2, No.2, pp. 19-36, 2012.

[2] R. Malhotra, A. Chug, "*An Empirical Study to Redefine the Relationship between Software Design Metrics and Maintainability in High Data Intensive Applications"* , Lecture Notes in Engineering and Computer Science, Proceedings of The World Congress on Engineering and Computer Science, USA, pp. 61-66, 2013.

[3] R. Malhotra, A. Chug , " *An Empirical Validation of Group Method of Data Handling on Software Maintainability Prediction using Object Oriented Systems*", International Conference on Quality Reliability InfoCom Technology and Industrial Technology, India, pp. 49-57, 2012.

[4] R. Malhotra,A. Chug,"*Application of Group Method of Data Handlingmodel for software maintainability prediction using object oriented systems*", International Journal of System Assurance Engineering and Management, ISSN 0975-6809, 5(2),pp.165-173, 2014.

[5] K.K.Aggarwal,Y. Singh,A.Kaur.,R. Malhotra, "*Empirical study of OO matrics*", Journal of Object-Technology, Vol.5, No.8, pp 149-173,2006.

[6] S. M. Yacoub, H.H. Ammar, and T.Robinson, "*Dynamic Metrics for OO Designs*", 5th International Software Metrics Symposium, Boca Raton, FL, USA, pp.50-61,1999.

[7] V.Gupta,"*Validation of Dynamic Coupling Metrics for object oriented Software*", ACM SIGSOFT Software Engineering Notes, vol 36, pp.1-5, September,2011.

[8] L.C. Briand, P. Devanbu, W.L. Melo, "*An investigation into coupling measures for C++*", in Proc. Of International Conference on Software engineering (ICSE'97), Boston, MA, pp. 412-421, 1997.

[9] V.Gupta, "*Measurement of Dynamic Metrics Using Dynamic Analysis of Programs*", Applied Computing Conference, Istanbul, Turkey, pp. 27-37, May,2008.

[10] A. Tahir, S.G.MacDonell, "*A systematic mapping study on Dynamic Metrics and Software Quality*", 28th International Conference of Softawre Maintenace, IEEE, pp. 326-325, 2012.

[11] E.Arisholm, "*Dynamic coupling measures for Object-Oriented Software*", 8th symposim on Software Metrics, IEEE,vol.30,no.8, pp. 491-506,August,2004.

[12] T.Reenskaug, P. Wold, O.A. Lehne, "*The OOram Software Engineering Method*". Manning/Prentice-Hall,1995.

[13] K.Kaur, K.Minhas, N.Mehan, and N.Kakkar, "*Static and Dynamic Complexity, Analysis Of Software Metrics*", World academy of Science , Engineering and Technology,vol 3, pp.08-23,2009.

[14] M. Young," *The Technical Writer's Handbook*", Mill Valley, CA: University Science, 1998.

[15] F.Ohata, K. Hirose, M.Fujii, "*A Slicing Methsod for Object–Oriented Programs Using Lightweight Dynamic Information*", proceedings of APSEC,pp. 273-280, Deceber,2001.

[16] B. Dufour, K. Driesen, L. Hendren, and C.Verbrugge," *Dynamic metrics for Java*", Proceedings of the 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, pp. 149–168, 2003.

[17] S .Allier, S. Vaucher, B. Dufour and H. Sahroui, "*Deriving Coupling Metrics from Call Graphs*",Working Conference on Source Code analysis and Manipulation , pp. 43-52,2010.

[18] S.Cook, H.Ji and R. Harrison, "*Dynamic and Static views of Software evaluation*", International conference on Software Maintenance,Italy,IEEE,pp. 592-601,2001.

[19] R S. Pressman, "*Software Engineering: A Practitioner's Approach*", 7/e, McGraw-Hill, 1982.

[20] J.Han and M. Kamber, "*Data Mining Concepts and Techniques*", 2/e, Elsevier, 2006.

[21] N.Ramakrishnan, C. Bailey-Kellogg, S. Tadepalli, and V. N. Pandey, "*Gaussian Processes for Active Data Mining of Spatial Aggregates*", SIAM Data Mining, 2005.

[22] J.Ulrich,"*Supervised Machine Learning for Email Thread Summarization*", Austin, (Master's thesis), University Of British Columbia, Vancouver, 2006.

[23] B.A. Kitchenham, L.M. Pickard, S.G. MacDonell, M. J. Shepperd, "*What accuracy statistics really measure*", Proceedings-Software, IEEE, vol. 148, no. 3, pp. 81-85,2001.

[24] http://www.mathworks.com.