ISSN:2320-0790

An International Journal of Advanced Computer Technology

OBJECT ORIENTED PROGRAMMING ACCURACY PREDICTION USING UNBIASED ITERATION PARTICLE SWARM OPTIMIZATION (UIPSO)

Kirti Dubey, Vivek Sharma

TIT Advance, Bhopal, India

Abstract: In this paper an object oriented programing accuracy prediction system has been proposed based on unbiased iteration particle swarm optimization (UIPSO). First the object oriented modules have been uploaded in the framework then it is categorized in different k-modules. According to the user choice it is selected. The selected module is then tested by chi-square test and if qualifies then checked by software quality metrics like FM, OR and P. Finally the accuracy is calculated based on UIPSO. The obtained results show that the results are outperforms from the previous method in terms of accuracy and error rates.

Keywords: Chi-Square Test, FM, PO, OR, UIPSO

1.INTRODUCTION

To foresee about the great nature of the projects or the product, module is an important term [1]. Software quality estimation is most important research areas in the field software engineering that attracts the experts. A lot of methods have already been developed in the area of software quality estimation [2]. The main objective of software quality estimation is to discover the tasks within software which are prone to errors so as to minimize the maintenance cost of the software and these precautions of errors will improve the quality of software [2, 3]. Software maintenance is time and resource consuming activity. Tracking the defect as early as possible in a software life cycle will not only improve the effective cost but will also help to achieve the customers' satisfaction and reliability of the software developed. Subsequent to examining a few exploration woks around there, we go over with two principle issues. First is the commotion and second which is the principle issue is the parameter on which you can sort the projects for the improvement [1].

There are for the most part two sorts of clamor in the class of information quality and programming parameters. The first is worry with the mislabeled programming modules, brought on by programming engineers neglecting to distinguish, neglecting to report, or basically disregarding existing programming flaws [4]. Uprooting such boisterous occasions can altogether enhance the execution of adjusted programming quality-estimation models [5]. Another principle test is that, in certifiable programming ventures, we have to discover the parameters on that we can assess the quality [6-8]. So there is the expectation system on the premise we can learn and anticipate [9]. So there is the need of managed and unsupervised methodology, which we will examine in the ensuing subsections [10].

Software quality assessment is growing in this era due to the demand of object oriented modularity. Estimation of software quality is a major concern today. Different researchers have used different software parameter and metrics for quality classification or fault prediction models. The most used are f-measure, odd ratio and power. In this paper different aspects of software quality estimation evolving object-oriented programs have been analyzed and discussed. The parameters used and object oriented programming properties selection have been discussed along with the applicability and reusability parameters.

2.LITERATURE SURVEY

In 2012, Sodiya et al. [11] proposed a survivability model. It is for object-oriented software system. It has been proposed for the purpose of solving the problem of software degradation. The authors have been observed that the software programs can prevent code degradation.

In 2012, Dantas et al. [12] suggested that the code composition can be flexible by the expressive mechanism of object-oriented programming techniques. It provides a support to the programmers in factoring the complexity of a program. It also provides the evolution. The properties of composition code might introduce new flavours of complexity, and in turn cause side effects on program evolvability. They proposed a framework for supporting programs structured with different structure composition.

In 2013, Saraiva [13] suggested the problem of metric selection for the experiment in object-oriented software metrics (OOSM) is a big problem. So authors have been suggested different usages of OOSM metrics in academia and industry

In 2014, Sharma et al. [14] suggested that the quality attributes can be overwhelmed by the size of program, control structure and module interfaces. They have selected metrics based on the past research for the evaluation purpose. The evaluations have been done by open source data metrics, principal component analysis (PCA) and principal axis factoring (PAF) for the redundant information elimination. The results proved that the maximum metrics are comparable.

In 2014, Singh et al. [15] suggested that the unique features of object-oriented programming language have been used in designing the new tools but it also causes redundant complexity in case of improper use. They also suggested that it is hard to debug. They have provided an analysis based on the inheritance metrics. According to the authors new metrics can be capable in adding shared properties and methods which are useful in acting as the additional parameters.

In 2015 Padhy et al. [16] discussed the role of reusability. They have suggested the pros and cons from different researchers for analyzing the reusability. They have tried to focus on the need of reusability.

In 2015, Beniwal et al. [17] suggested that the testing is important in the development process. It ensures the reliability and the quality of the software. It is based on the effectiveness of the software and programming metrics. The authors have explained the roles of testing these metrics in object oriented programming languages. For this they have applied it on different projects and found the usefulness in g the software development process.

In 2015, Zhabelova et al. [18] suggested that the software metrics are useful in assessing the quality and modules identification. It also emphasized the cost of testing and maintenance. They have provided an analysis based on different metrics. According to the authors new metrics can be capable in analyzing power system protection software. They suggested that it can save the cost by the addition of additional parameters.

In 2015, De et al. [19] suggested that the object oriented programming provides efficient control in terms of complexity. So it can achieve faster development, better cost reduction, good quality, easier maintenance, increased scalability, better information structures, and increased adaptability. They have focused the limitations and the advantages.

In 2015, Suri et al. [20] suggested that the software metrics can be applied for the software development process and quality improvement. They have applied frequency and descriptive analysis for the metrics used. They have used cohesion, coupling and inheritance characteristics as the metrics for experimentation.

In 2016, Pawade et al. [21] differentiated different software complexity metrics in the basis of procedural and object oriented approach. They have found the problem of data size increase and complexity and maintability. They discussed various procedural and object oriented software metrics in this respect. For this they have calculated sample code complexity by using different procedural metrics. Their results indicate that the complexity for same code differs from metric to metric.

In 2016, Abilio et al. [22] suggested ffeature-oriented programming (FOP) to implement software product lines. It is on composition mechanisms that are also called refinements. They have selected metrics based on the observations from the previous works and finally it is compared for the performance parameters.

3.PROPOSED WORK

The object oriented programming accuracy prediction system has been presented in this paper. Object oriented modules are considered as the dataset. It is based on four object oriented properties as class module, object module, inheritance module and dynamic behaviour module. Six different clusters are considered for filtering the data based on the properties. If categorized the data according to the choices determined by the different groups of properties. For sending the data for quality testing it is first tested by chi-square test. It is checked by expected (E) and observed values (O).

$$\chi^2 = \sum \frac{(E - O)^2}{E}$$

Then object oriented metrics are applied for the quality metrics. F-measure (FM), odd ratio (OR) and power (P) have been used for quality estimation. In this section we discuss the above in details.

True positive (TP), false negative (FN) and false positive (FP) determine the FM ratio. It is also determined through precision (P) and recall (R). It is used as the evaluation metrics.

$FM = (2 * P * R) / (P + R)$

Where:

$P = TP / TP + FP$

$R = TP / TP + FN$

OR provides the accuracy of association of outcome and the input.

$OR = 2 * R (1-P) / (1-P * R)$

Power (P) is defined as:

$P = ((1-P)k-(1-R)k)$

Then final accuracy has been obtained by unbiased iteration particle swarm optimization (UIPSO). In this process in each iteration the additive factor is random so the results are unbiased irrespective of iteration.

UIPSO Algorithm

I_NS –Input set
FA– Final accuracy
R_T – Random trail
R_T P –Random trail previous
N–Total number of attributes

Input:

- $I_NS(I_NS_1, I_NS_2 \ldots. I_NS_n)$

Output:

- $FA_1, FA_2, \ldots\ldots FA_n$

Step 1: Input values from software metrics have been assigned.
Step 2: The values are initialized as the swarm values.
Step 3: Initial assignment not required any updations.
Step 4: Iteration process
i=1 to 5 do
$X_i = (I_NS_1 + I_NS_2 + I_NS_3 + \ldots\ldots\ldots. + I_NS_n)/N$
    If $(I_NS_{ti+1} > I_NS_{ti})$
    $S_{i+1} = S_i$
    while;
For 2 to 5
$X_{i+1} = X_i + (I_NS_1 \text{ X } R_{Ti} + I_NS_2 \text{ X } R_{Ti} + I_NS_3 \text{ X } R_{Ti} + \ldots\ldots\ldots. + I_NS_n \text{ X } RV_i)/n- R_T P_i$
    If $(I_NS_{ti+1} > I_NS_{ti})$
    $S_{i+1} = S_i$
    while;
Step 5: Accuracy as the results achieved.
Step 6: Finish

It is applied to calculate the overall and average object oriented programing accuracy prediction using unbiased iteration particle swarm optimization (UIPSO). The flowchart shown in figure 1 shows the overall phenomena. First the dataset is uploaded in the system, then k-selection is performed and then testing is done through chi-square test. Then software quality metrics have been applied and the classification accuracy has been created by UIPSO.

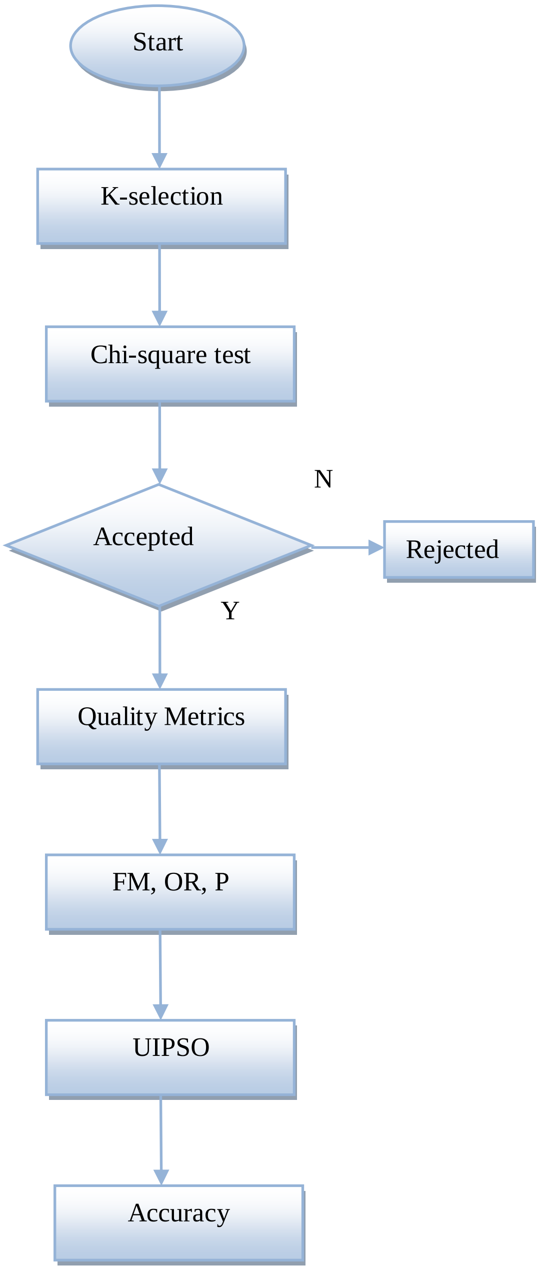

Figure 1: Flowchart

4.RESULT EVALUATION

In this paper the results are shown based on 11-20 group cluster. The result for FM based on class module, object module, inheritance module and DMA are shown in table 1. The result for OR based on class module, object module, inheritance module and DMA are shown in table 2. The result for P based on class module, object module, inheritance module and DMA are shown in table 3. Overall comparison is shown in table 4. The result accuracies are shown and comparison with previous method is shown in figure 1 to figure 4. Error comparison is shown in figure 5 to figure 6.

Table 1: F-measure

Sno	File name	Class	Object	Inheritance	DMA
1	P10.java	0	0	0	0
2	P11.java	0.24	0	0.86	0
3	P12.java	0.31	0.03	0.86	0.03
4	P15.txt	0	1.1	0.86	1.1
5	P5.java	8.78	2.78	0	2.78
6	P9.java	0	0.1	0.86	0.1

Table 2: OR ratio

Sno	File name	Class	Object	Inheritance	DMA
1	P10.java	0	0	0	0
2	P11.java	0.16	0	0	0
3	P12.java	0.19	0.03	0	0.03
4	P15.txt	0	1.15	0	1.15
5	P5.java	0.91	0.79	0	0.79
6	P9.java	0	0.07	0	0.07

Table 3: P ratio

Sno	File name	Class	Object	Inheritance	DMA
1	P10.java	0	0	0	0
2	P11.java	0.16	0	0	0
3	P12.java	0.19	0.03	0	0.03
4	P15.txt	0	1.15	0	1.15
5	P5.java	0.91	0.79	0	0.79
6	P9.java	0	0.07	0	0.07

Table 4: Overall comparison

Sno	File name	FM	OR	P
1	P10.java	0.0	0.0	-0.0
2	P11.java	1.1	0.16	-0.16999999999999998
3	P12.java	1.23	0.25	-0.21999999999999997
4	P15.txt	3.06	2.3	0.1
5	P5.java	14.339999999999998	2.49	46.699999999999996
6	P9.java	1.06	0.14	-0.16

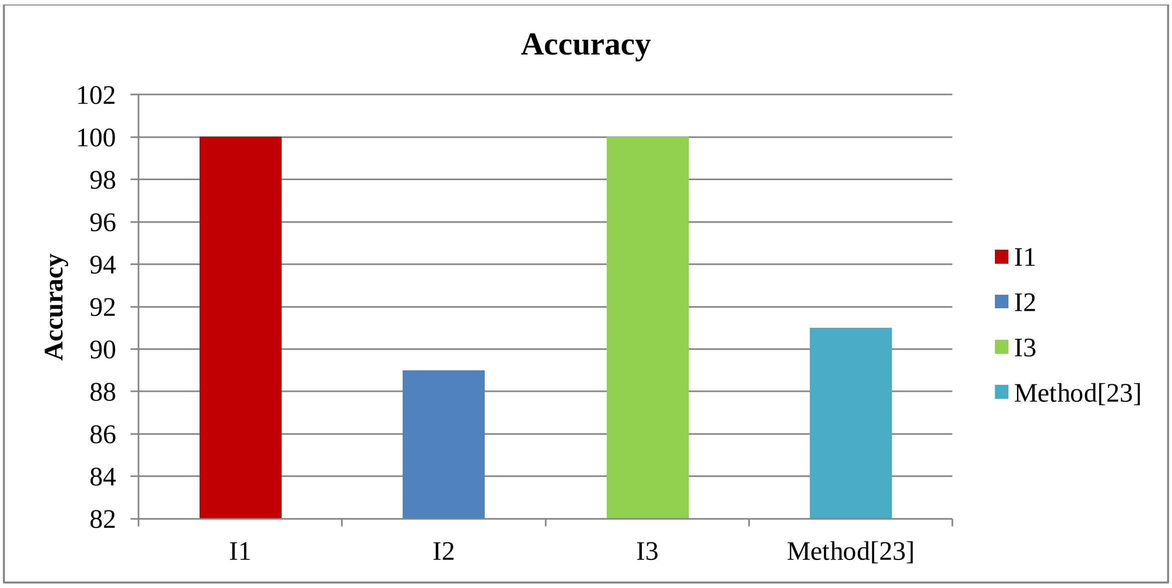

Figure 1: Accuracy with I1

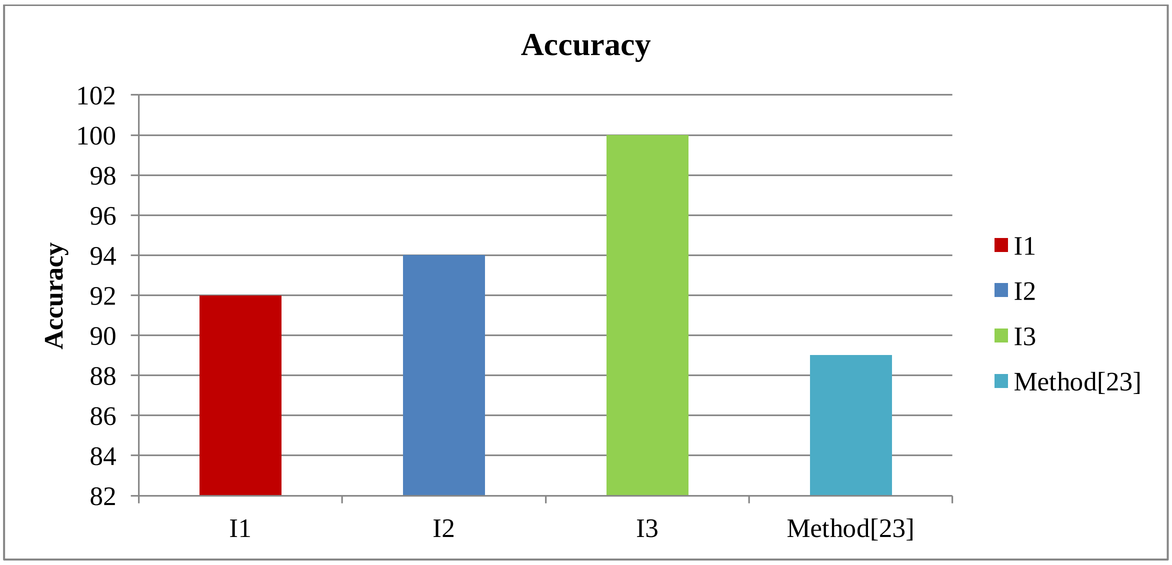

Figure 2: Accuracy with I2

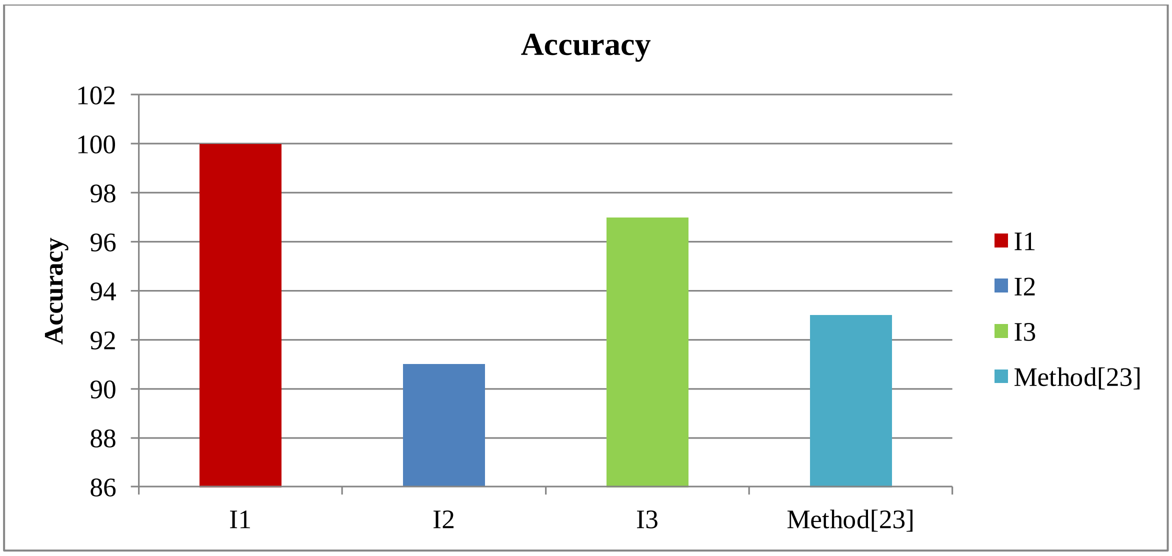

Figure 3: Accuracy with I3

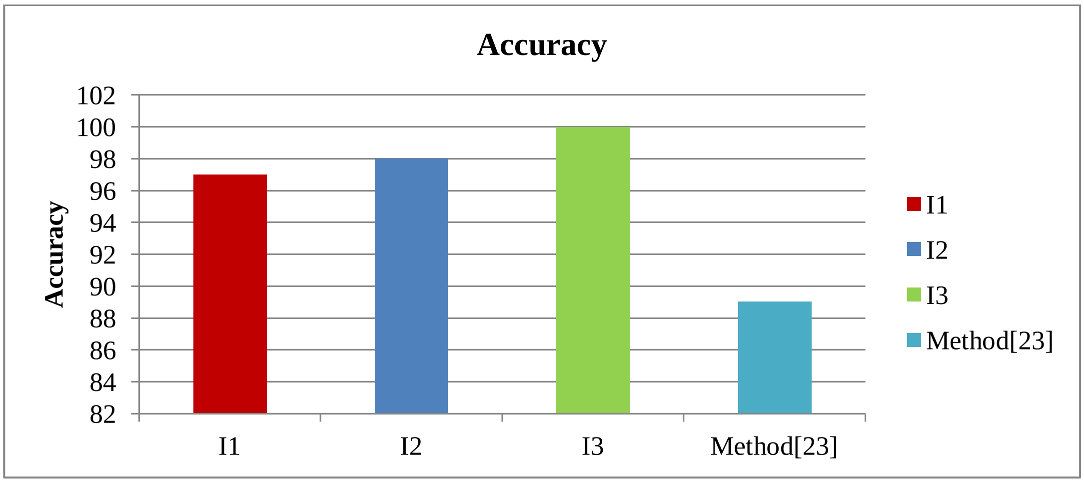

Figure 4: Accuracy with I4

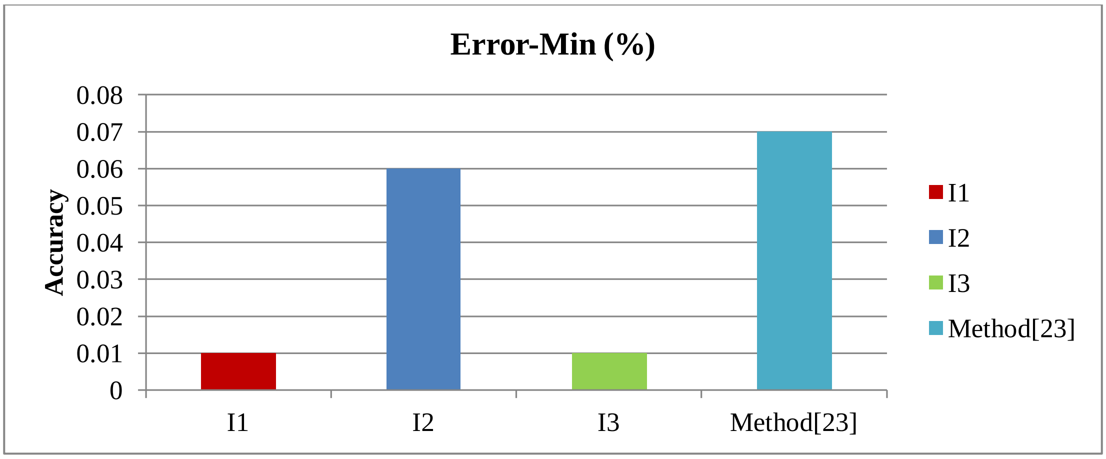

Figure 5: Error (%)-Min

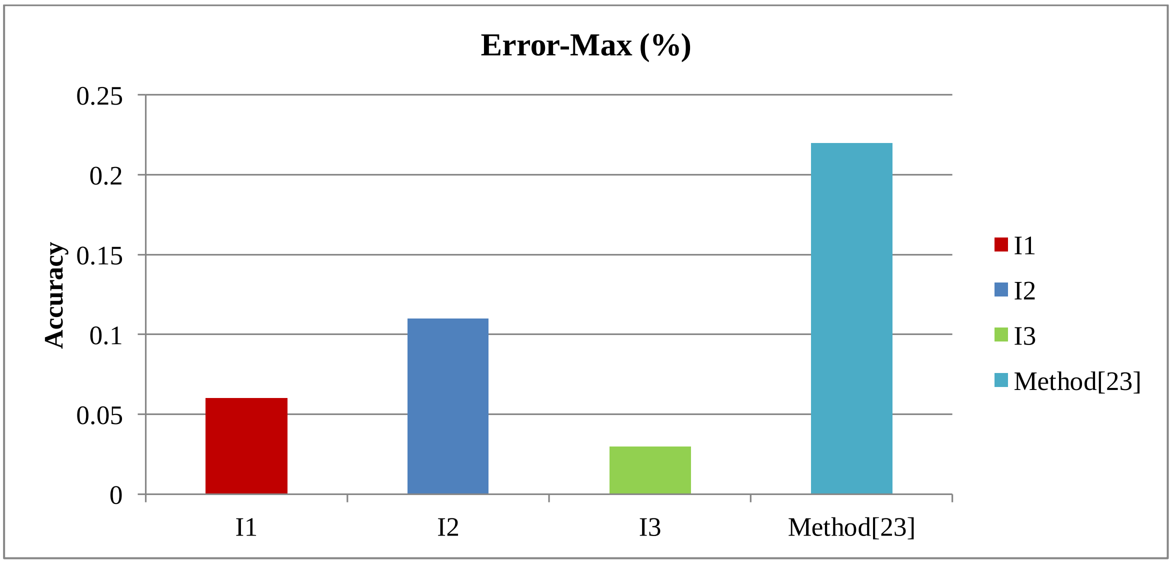

Figure 6: Error (%)-Max

5. CONCLUSION

This paper deals with the object oriented modules quality prediction. The comparison parameters are accuracy and error rates. The dataset is first tested by chi-square test and then the qualified data is then checked by FM, OR and P. The final values obtained by this process are then input to the UIPSO for the accuracy and error rates calculation. The results obtained are improved from the previous method. In future it can be extended with the methods presented in [24, 25].

REFERENCES

[1] 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.

[2] Kaur I, Kaur A. Empirical study of software quality estimation. In Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology 2012 Oct 26 (pp. 694-700). ACM.

[3] Alotaibi MB. Antecedents of software-as-a-service (SaaS) adoption: a structural equation model. International Journal of Advanced Computer Research. 2016 Jul 1;6(25):114.

[4] Reformat M, Pedrycz W, Pizzi NJ. Software quality analysis with the use of computational intelligence. Information and Software Technology. 2003 May 1;45(7):405-17.

[5] Baz A. Efficient data analysis approaches to enhance the quality of customer service in Saudi Government sector. International Journal of Advanced Computer Research. 2016 Jan 1;6(22):25.

[6] Rashid E, Patnaik S, Bhattacherjee V. Software quality estimation using machine learning: Case-Based reasoning technique. International Journal of Computer Applications. 2012 Jan 1; 58(14).

[7] Acharya, Anal, and Devadatta Sinha. "Assessing the Quality of M-Learning Systems using ISO/IEC 25010." International Journal of Advanced Computer Research 3.3 (2013): 67.

[8] Tashtoush Y, Al-Maolegi M, Arkok B. The correlation among software complexity metrics with case study. arXiv preprint arXiv:1408.4523. 2014 Aug 20.

[9] Dileep MR. An Effective Approach for Indexed Data Access based on Linear Search Technique Using Reference Key Algorithm. International Journal of Advanced Technology and Engineering Exploration. 2015 Apr 1; 2(5):49.

[10] Kanmani S, Uthariaraj VR, Sankaranarayanan V, Thambidurai P. Object-oriented software fault prediction using neural networks. Information and software technology. 2007 May 31; 49(5):483-92.

[11] Sodiya AS, Aborisade DO, Ikuomola AJ. A survivability model for object-oriented software systems. In Computational Aspects of Social Networks (CASoN), 2012 Fourth International Conference on 2012 Nov 21 (pp. 283-290). IEEE.

[12] Dantas F, Garcia A, Whittle J. On the role of composition code properties on evolving programs. In Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement 2012 Sep 19 (pp. 291-300). ACM.

[13] Saraiva J. A roadmap for software maintainability measurement. In Proceedings of the 2013 International Conference on Software Engineering 2013 May 18 (pp. 1453-1455). IEEE Press.

[14] Sharma R, Sabharwal S, Nagpal S. Empirical analysis of object oriented metrics using dimensionality reduction techniques. In Recent Advances and Innovations in Engineering (ICRAIE), 2014 2014 May 9 (pp. 1-5). IEEE.

[15] Singh NB, Alsadoon A, Prasad PW, Singh AK, Shrestha AK. Impact of shared attributes and methods in calculation of object-oriented inheritance metrics. In Emerging Technology Trends in Electronics, Communication and Networking (ET2ECN), 2014 2nd International Conference on 2014 Dec 26 (pp. 1-7). IEEE.

[16] Padhy N, Panigrahi R, Baboo S. A Systematic Literature Review of an Object Oriented Metric: Reusability. In Computational Intelligence and Networks (CINE), 2015 International Conference on 2015 Jan 12 (pp. 190-191). IEEE.

[17] Beniwal R. Analysis of Testing Metrics for Object Oriented Applications. In Computational Intelligence & Communication Technology (CICT), 2015 IEEE International Conference on 2015 Feb 13 (pp. 41-46). IEEE.

[18] Zhabelova G, Vyatkin V. Towards software metrics for evaluating quality of IEC 61499 automation software. In Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on 2015 Sep 8 (pp. 1-8). IEEE.

[19] De Silva DI, Kodagoda N, Kodituwakku SR, Pinidiyaarachchi AJ. Limitations of an object-oriented metric: Weighted complexity measure. In Software Engineering and Service Science (ICSESS), 2015 6th IEEE International Conference on 2015 Sep 23 (pp. 698-701). IEEE.

[20] Suri B, Singhal S. Investigating the OO characteristics of software using CKJM metrics. In Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions), 2015 4th International Conference on 2015 Sep 2 (pp. 1-6). IEEE.

[21] Pawade D, Dave DJ, Kamath A. Exploring software complexity metric from procedure oriented to object oriented. In Cloud System and Big Data Engineering (Confluence), 2016 6th International Conference 2016 Jan 14 (pp. 630-634). IEEE.

[22] Abilio R, Vale G, Figueiredo E, Costa H. Metrics for feature-oriented programming. In Emerging Trends in Software Metrics (WETSoM), 2016 IEEE/ACM 7th

International Workshop on 2016 May 15 (pp. 36-42). IEEE.

[23] Mekruksavanich S. An adaptive approach for automatic design defects detection in object-oriented systems. In Digital Arts, Media and Technology (ICDAMT), International Conference on 2017 Mar 1 (pp. 342-346). IEEE.

[24] Li K, Gao Y. Fuzzy clustering with the generalized entropy of feature weights. International Journal of Advanced Computer Research. 2016; 6(27):195.

[25] Mareli M, Twala B. Global optimisation using Pareto cuckoo search algorithm. International Journal of Advanced Computer Research. 2017; 7(32):164-75.