Empirical Validation of Variable Method Interaction Cohesion Metric (VMICM) for Enhancing Reusability of Object-Oriented (O-O) Software

Original Scientific Paper

Bharti Bisht
Research scholar, School Of Computer Applications, MRIIRS
Faridabad, India
onebharti@gmail.com

Parul Gandhi
Professor, School Of Computer Applications, MRIIRS
Faridabad, India
parul.fca@mriu.edu

Abstract *– Any object-oriented (O-O) module's primary goal is to build classes with a high level of coherent interaction between variables and methods. To increase the quality of O-O (Object-Oriented) software, various metrics emphasizing cohesiveness have been established so far. These metrics operate on both the design and the code levels. However, these metrics still fall short of fully measuring the cohesion of object-oriented (O-O) software. Based on several concepts of cohesive interlinkages between variables and procedures, the study proposed an enhanced cohesion metric. The four forms of cohesive linkages (VMRv, VMMv, VMRTv, and VMOv) between variables and procedures were the focus of this study. The axiomatic frame of reference was employed for theoretical validation, and univariate logistic regression was applied in the MATLAB environment for empirical validation. The approach of univariate logistic regression has been adopted because it provides incredibly accurate data and can even be applied to datasets that can be linearly separated. The proposed metric exhibits high cohesion, which is the ultimate perspective of a highly reusable Object-Oriented (O-O) module, as evidenced by the testing phase and even training the real dataset with reusability prediction in terms of high values of precision, recall, R2, and low value of RSME of VMICM metric. The study results demonstrated that the proposed metric can act as a measure for predicting the reusability of the Object-Oriented (O-O) system.*

Keywords*: cohesion, cohesive linkages, logistic regression, reusable, validation*

1. INTRODUCTION

The importance of software applications has increased in recent years, and the wide variety of advantages offered by them is drawing a large global audience [1, 2]. Any software development phase involves a variety of intricate [3] tasks that must be efficiently managed if high-quality software is to be produced. It is possible to reduce the work needed for similar procedures by using the enhanced software generated after careful monitoring of various activities at various stages, and this will also assist in increasing the overall quality of newly developed software [4]. Measurement supports an institution's efforts to enhance the overall software development process and even improve the product's overall quality. Additionally, this would assist programmers in addressing the complexity of a particular software component at a very early stage of the process.

Building high-quality software, and Object-Oriented- based techniques are still quite popular [5]. Good modular designs maximize cohesion. The attributes of high cohesion classes are not shared with other classes, and they can be reused.

Therefore, by reducing complexity, a high level of cohesion promotes reusability. To assess software modules concerning cohesion, many scholars have developed numerous Object-Oriented (O-O) metrics [6]. However, in several situations, such measures do not take into account cohesion properly and thus have only a limited view of measuring it. To accurately measure cohesiveness, there is therefore a need to develop cohesion metrics that vary given the idea of cohesive interactions. This study presents the following contributions:

- Proposing a new cohesion metric called VMICM that emphasizes variable-method interactions.
- Validating the proposed metric empirically using a univariate logistic regression approach and studying correlation with other cohesion-based metrics.
- Analyzing the relationship between metric and degree of inheritance.

The framework of this work is further organized as Section II provides a detailed literature analysis of various studies. Section III details about designing phase of the proposed metric i.e. VMICM metric and a case study. The validation (theoretically and empirically) of the proposed metric is discussed in Section IV. Section V finally outlines the conclusion drawn from the study.

2. LITERATURE ANALYSIS

In the Object-Oriented (O-O) model, cohesion measures the relatedness between variables as well as methods of a particular class [6, 7]. Relatedness depicts the similarity between variable-method interactions. Cohesion can be categorized into a range from highly desirable i.e. functional cohesion to very low desirable i.e. coincidental cohesion. Cohesion can be maximized by effective as well as efficient O-O (Object-Oriented) based systems. The complexity of particular software can increase due to the low value of cohesion which in turn increases bugs during the development phase. Any class with a low value of cohesion can be further segmented into many subclasses [7] that would help to increase the value of cohesion.

The requirements of common instance variable usage by method pairs serve as the foundation for cohesion. measurements of Tight Class Coherence (TCC) and Loose Class Cohesion (LCC) provided by Bieman et al. [8] The authors suggested that the two methods are considered related if they use or refer to the same instance variable either directly or indirectly. The fact that variable A occurs in the body of method M indicates that variable A is being used directly by M. While a direct reference to variable A by a method M' that is either directly or indirectly invoked by method M qualifies as an indirect use of variable A by method M. TCC is defined as the proportion of method pairs that are directly connected, whereas LCC is defined as the proportion of method pairs that are either directly connected or indirectly connected.

Kakkar et al. [9] developed a SCOM metric that emphasizes interlinkage intensity in addition to taking weight into account. The ratio of the number of shared variables between two techniques to the greatest number of variables that any method can access is used to determine how strongly two methods are interconnected. The weight component is determined by dividing the total number of variables in a certain class by the number of variables that are shared by the method pair. SCOM value is the sum of the products of the weight and interlinkage intensities of all potential method pairs. Kansal et al. [10] in their study do not consider criteria based on similarity but focus on interlinkage patterns between various methods to compute cohesion. A class's protected or private meth-ods never access any of the variables found in that class. If protected or private methods are invoked by 2 public methods then we can say that they are interconnected to each other. Khajenoori et al. [11] suggested the first-class-based cohesion metric, LCOM (Lack of Cohesion in methods). This metric computes several method pairs that do not share any variable. It also represents the count of methods having zero similarity minus the count of methods not having zero similarity. If there are several pairs of comparable method definitions, the class is more cohesive. If none of a class's method pairs employ instance attributes, then there is no similarity between them, and the LCOM metric's value will be zero. The LCOM metric value is a measurement of the distinctiveness of the various method pairings that are present in the class.

Rasool et al. [12] discussed cohesive interlinkages between variables used by methods. They have proposed 3 types of cohesive interlinkages and then categorized them based on the ranking as well as weights. Both theoretical and empirical validation have been carried out. Pearson analysis and logistic regression are used to carry out experiments. Alsarraj et al. [13] introduced AECC which measures the degree of interconnectedness between distinct methods via variable-variable linkage as well as the invocation of methods in a specific class while taking the size of the cohesive fragment into account. even correlation experimentation of this metric with previously proposed class-based cohesion metrics was completed in this study. The comparison of related studies on cohesion-based metric approaches for reusability enhancement conducted by various scholars is shown in Table 1.

Table 1. Comparative Analysis of Various Frameworks related to reusability enhancement

Ref. No.	Motive	Metric used for Study	Conclusion
[4]	Developers of Object-Oriented software can identify flaws in class design and LSCC metric	LSCC metric	The findings imply that the cohesion metric which takes into consideration the level of interaction between each pair of methods, can more effectively explain the class quality
[14]	High Precision Cohesion Measure (HPCM), a new cohesion metric that aims to address the shortcomings of the preceding metrics	HPCM Metric	One of the desirable characteristics of a good object-oriented design is a class with high cohesiveness. A class that works well together is less likely to make mistakes and is simple to create and keep up with

[15]	Variable Frequency-Inverse Method Frequency (VF-IMF) metric is proposed to evaluate the degree of cohesion in modules and to group module methods to foster high cohesion	VF-IMF metric	To make it easier for a developer to create a module that ensures code reuse, the variables in this study are grouped using the VF-IMF metric to show what level of cohesion exists in a module among three low, high, and medium cohesions
[16]	Proposed a new method of measuring the cohesion of object-oriented software at the module level, where modules initially denote a single class and then a group of classes in later sections	UPBC metric	The suggested method aims to increase the object-oriented software's cohesion iteratively and repeatedly, calculating the cohesion score and using it to do clustering up until there is no longer a meaningful increase in the system's overall cohesion
[17]	The method devised in this study offers a technique to gauge cohesiveness measures at the class level	CSM metric, ACSM metric	Very cohesive classes must be designed with a strong coupling between their methods and a coherent internal description

3. MATERIAL AND METHODS

One of the most crucial aspects of an object-oriented (O-O) module is its high cohesiveness, which even aids in identifying the most reusable module. This section summarises the methodology of the proposed VMICM metric by anticipating high cohesiveness between variable-method interlinkages.

3.1. PROPOSED METRIC- VARIABLE METHOD INTERACTION COHESION METRIC (VMICM)

The majority of cohesion metrics that have been developed so far focus either on the use of methods within a class or their similarity due to the sharing of instance attributes. A new class cohesiveness metric, which takes into account the interlinking of variable methods from various contexts, has been proposed in this study. To better understand the variable methods interlinkages, this study has concentrated on four distinct coherent linkages between variables and methods. Considering variable $vi \in V$ and method $mn \in Md$ of class F, this study elucidated four types of cohesive interlinkages in the following manner:

i. Received (CRv) [17, 20]: ($V \times Md$) such that vi*CRv* mn, if $vi \in V$ is received as a parameter to a particular method $mn \in Md$.

ii. Manipulated (CMV) [17, 20]: ($V \times Md$) such that vi *CMv *mn, if $vi \in V$ is manipulated by method $mn \in Md$. This computation could be done either mathematically or it could be in the form of a function call [17, 20].

iii. Returned ($CRtv$) [20]: ($V \times Md$) such that vi*$CRtv$* mn, if the value of $vi \in V$ is returned by method mn $\in Md$.

iv. Override (COv): ($V{\times}Md$) such that vi*COv*mn, if the scope of vi is overridden by method $mn \in Md$.

3.1.1. Defining Proposed Metric- Variable Method Interaction Cohesion Metric (VMICM)

This proposed metric describes the MMI definition [20], which serves as the foundation for the widely acknowledged class-based cohesion features as indicated in [21], and this even has the potential to be utilized as a signal for the reconstruction of classes with weak cohesion. Non-inherited methods (both private as well as public) have been taken into consideration for the study. Eq.(1) represents the combination of several cohesive interlinkages in Class F as:

$$\sum CIv = \left(\begin{array}{l}\left(\mathrm{CRi1} \cup \mathrm{CMi1} \cup \mathrm{CRti1} \cup \mathrm{VMOi1}\right)\ldots \\ \cup \left(\mathrm{CRin} \cup \mathrm{CMin} \cup \mathrm{CRtin} \cup \mathrm{COin}\right)\end{array}\right) \quad (1)$$

where CIv represents the union of all proposed cohesive interlinkages between different variables and methods of Class F.

The two interconnected methodologies shown by VMICM are AVU (Average Variable Usage) and CSv (Cohesive Strength). AVU (Average Variable Usage) as defined in Eq. (2) denotes the average count of all variables used by all methods included in Class F:

$$\mathrm{AVU} = \frac{\sum \mathrm{CIv}}{\mathrm{Md}} \quad (2)$$

where Md represents the total methods present in a Class F.

The total count of common variables between various methods pairs is represented by Eq. (3) as:

$$\mathrm{CVv} = |\,\mathrm{VM1} \cup \mathrm{VM2} \cup \mathrm{VM3} \ldots \cup \mathrm{VMn}\,| \quad (3)$$

Based on the count of shared variables between method pairs, Cohesive Strength (CSv) represents the degree of [21, 22] cohesive interlinkages between them. Cohesive Strength is defined by Eq. (3) as:

$$\mathrm{CS_v} = \begin{cases} \dfrac{\mathrm{CV_v}}{\mathrm{AVU}} & \text{if } \mathrm{CV_v} <= \mathrm{AVU} \\ 1 & \text{if } \mathrm{CV_v} > \mathrm{AVU} \end{cases} \quad (4)$$

Where:

CVv = Total count of common variables between various method pairs of Class F

AVU = Average count of all variables used by all methods included in Class F

$VMICM$ represents the average of cohesive strength (CSv) between variables and methods available in a particular class.

In the same way as an AVU analysis is carried out, it will take account of both privately and publicly inherited methods, and then their cohesive strength is added up. So, $VMICM$ proposed by this study is represented by Eq.(5) as follows:

$$\text{VMICM} = \frac{\text{CSv}}{\text{Md} * \frac{\text{Md} - 1}{2}} \tag{5}$$

3.2. RESEARCH METHODOLOGY

In this section, the process flow of the proposed metric $VMICM$ (Variable Method Interaction Cohesion Metric) is discussed. The research methodology used in this work was carried out in various phases as shown in Fig. 1. Different phases of the research flow are as follows:

- Initially, various metrics from the online $MAVEN$ repository were assimilated.
- To examine interlinkages between variable methods and their interaction with the reusability factor, cohesion metrics derived from the $MAVEN$ repository have been applied as input.
- The concept of the MMI was used as a foundation to create more coherent interlinkages.
- The next phase was the creation of a proposed metric $VMICM$ outline using 2 methodologies - AVU (Average Variable Usage) and CSv (Cohesive Strength).
- Theoretical validation has been performed using an axiomatic frame of reference. Alternative measurement hypotheses relating to the length, size, coupling, complexity, and unity of the software are analyzed using a base of reference for axioms in this model.
- Univariate logistic regression has been used in the MATLAB environment for empirical verification. This method was used to assess both the efficiency of the proposed $VMICM$ metric as well as those from the previous studies. It also demonstrates how the proposed metric contributes significantly to the reusability of Object-Oriented (O-O) software systems.
- Open-source JAVA-based classes were used as data sets in the testing and training phases.
- After repeated repetition of the training data set, the work was able to achieve high precision, recall, $R2$, and low $RSME$ values for the proposed metric.

Thus, this paper demonstrated that the proposed metric shows high cohesion and can act as a measure for predicting the reusability of Object-Oriented (O-O) classes.

4. RESULTS AND DISCUSSION

The theoretical and empirical validation of the proposed metric $VMICM$ using several techniques and approaches is discussed in this section.

4.1. THEORETICAL VALIDATION OF PROPOSED METRIC VMICM

This study incorporated the framework suggested by Briand for validating the proposed metric $VMICM$ theoretically. In this work, an axiomatic frame of reference for different software properties such as length, size, coupling, complexity, and cohesion are used to examine the various measurement parameters. To determine cohesion and verify that the $VMICM$ metric complies with the abovementioned benchmarks, there are several properties applied in this framework:

- Non-negativity[14-16]: $VMICM$ is calculated with the help of the modulus of the union of four types of cohesive interlinkages CRv, CMv, $CRtv$, and COv and the resultant value of $VMICM$ obtained was not less than
- Normalization [14]: When there are no cohesive interlinkages between the methods and the variable, $VMICM$ acquires 0 (i.e., the minimum value), and when there are all cohesive interlinkages, it reaches 1 (i.e., the maximum value). So, the value of the $VMICM$ metric falls between the range [0-Max].
- Cohesive modules [14, 15]: This property states that if two unrelated components are combined, the cohesiveness of the combined component will not increase. $VMICM$ metric satisfies this property.
- Monotonicity [14, 15]: Any cohesive interlinkages added to the proposed framework means that either $|CRv|$,$|CMv|$, $|CRtv|$, or $|COv|$ increases by 1. Consequently, the value of the $VMICM$ metric would be either increased or not affected at all.
- Null value [23]: The $VMICM$ metric shall be 0 if there is no cohesion among variables and methods of a particular class used in the analysis, such as CRv, CMv, $CRtv$, or Cov

All the properties suggested by Briand are met by the

proposed $VMICM$ metric. Thus $VMICM$ metric is proven to be a valid cohesion metric.

4.2. EMPIRICAL VALIDATION OF PROPOSED METRIC VMICM

We have empirically tested the proposed metric $VMICM$ in this section to assess reusability for an Object-Oriented (O-O) module. This study produces a highly cohesive metric that contributes towards the high reusability of Object-Oriented(O-O) software systems.

4.2.1. Experimental Set-up

The analysis in the above study was carried out using JAVA-based classes from an online $MAVEN$ Repository. They were relatively different from each other in terms of methods pairs used in particular classes, due to the differences in size between the classes chosen for this study.

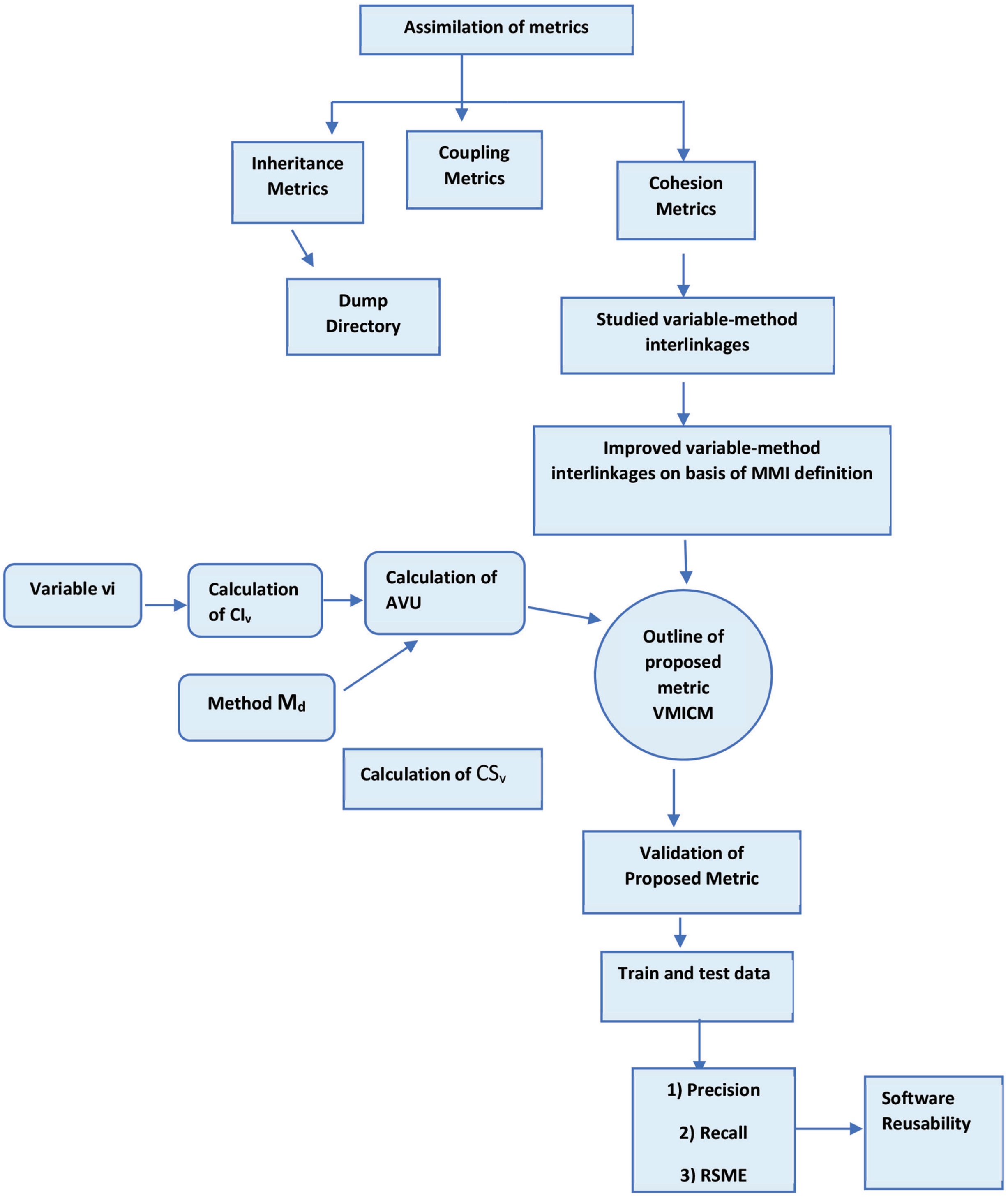


Fig. 1. Research Methodology Flowchart

Table 2 depicts the datasets used for this study.

Table 2. Datasets

Class Name	MD	CRV	CMV	CRtv	COv	Civ	AVU
Loan	3	3	3	3	1	10	3.3333
BMI	2	1	3	3	1	8	4
Employee	2	4	4	2	2	11	5.5
Course	3	2	1	1	1	5	1.6667
Box	2	4	4	2	1	12	6
Stack	3	1	1	1	1	4	1.3333

Complex	2	3	1	1	1	6	3
Shopping Cart	3	4	4	4	2	14	4.6667
Account	2	1	1	3	1	6	3
Lottery	3	4	4	4	1	13	4.3333
Person	3	2	2	2	2	8	2.6667
Triangle	3	3	2	3	2	10	3.3333
Stack of Integers	4	2	2	1	1	6	1.5
Car	3	3	3	3	3	12	4
Rectangle	2	2	1	2	1	6	3
Queue	5	3	1	1	3	8	1.6
Tax	2	2	2	2	2	8	4
Compute Change	6	4	4	5	2	12	2
Circle	3	1	1	4	1	4	1.3333
Calendar	2	2	2	6	2	8	4

After collecting the dataset from open-source projects, the proposed VMICM metric value is computed for the input classes. Earlier proposed cohesion metric values are also computed for the same input classes. Table 3 lists the calculated values of the VMICM metric and other cohesion-based metrics for each input class.

Table 3. Values of Proposed VMICM metric and other cohesion-based metrics of input classes used for the study

CNO	Class Name	CC	LAMCC	VMICM
CN1	Loan	0.67	0.34	3.3333
CN2	BMI	1.45	0.62	4
CN3	Employee	0.74	0.75	5.5
CN4	Course	0.48	0.45	1.6667
CN5	Box	0.457	0.356	6
CN6	Stack	1.22	0.833	1.3333
CN7	Complex	0.75	0.93	3
CN8	Shopping Cart	0.345	0.225	4.6667
CN9	Account	0.98	0.73	3
CN10	Lottery	0.35	0.26	4.3333
CN11	Person	0.350	0.456	2.6667
CN12	Triangle	0.451	0.382	3.3333
CN13	Stack of Integers	0.777	0.775	1.5
CN14	Car	0.25	0.38	4
CN15	Rectangle	0.65	0.52	3
CN16	Queue	0.90	0.82	1.6
CN17	Tax	0.83	0.73	4
CN18	Compute Change	0.35	0.28	2
CN19	Circle	0.534	0.486	1.3333
CN20	Calendar	0.76	0.50	4

To assess how the proposed VMICM metric can be adapted to the reusability of these classes, values for the proposed VMICM metric and all cohesion metrics are shown in the table above.

4.2.2. Data Selection

- Context Selection: This paper aims to demonstrate the fact that the proposed metric can act as a measure for the reusability of Object-Oriented (O-O) classes.
- Variable Selection: This work considers cohesion as an independent attribute and reusability measure as a dependent attribute.

4.2.3. Result Analysis

The univariate logistic regression technique [24, 25] is used in this study. The effectiveness of the proposed metric VMICM as well as currently utilized metrics from earlier studies have been independently examined using this method. It also demonstrates how the proposed metric contributes significantly to the reusability of Object-Oriented (O-O) software systems. Table 4 compares the proposed VMICM metric with existing cohesion-based metrics.

Table 4. Descriptive Analysis Results

Metric	Min.	Max.	Mean	Median	Standard Deviation
LCOM1	0.0	7875	89.45	20.0	336.7
LCOM2	0.0	7875	69.61	9.0	314.3
LCOM3	0.0	11.0	1.32	1.0	0.90
LCOM4	0.0	11.0	1.31	1.0	0.89
LCOM5	0.0	2.0	0.76	0.82	0.26
TCC	0.0	1.0	0.42	0.37	0.35
DCD	0.0	1.0	0.45	0.42	0.31
CAMC	0.0	1.0	0.39	0.39	0.16
NHD	0.0	1.0	0.64	0.67	0.16
LAMCC	0.0	0.9	0.54	0.51	0.22
VMICM	0.0	0.6	0.27	0.25	0.15

A graphical representation of a comparison of existing cohesion-based metrics and the proposed metric VMICM is shown in Fig. 2.

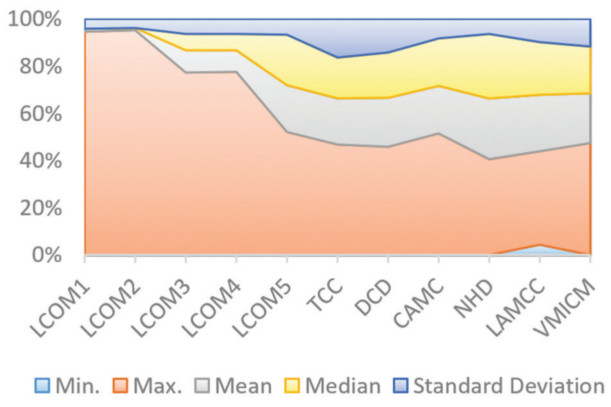


Fig. 2. Analysis of Proposed Metric VMICM and other metrics

In comparison to the existing metrics, according to the above analysis, the proposed VMICM metric has low mean and median values.

This lower value suggests that this study did not take into account an increased inheritance level. Increased level of inheritance leads to less reusable Object-Oriented (O-O) modules. Various statistics have been reported in this study for the evaluation of the proposed metric VMICM. These statistics are as follows:

- Precision [23-25]: This depicts the ratio between the counts of classes that are least reusable to the total count of reusable classes.
- Recall [23-25]: This depicts the ratio between the counts of classes that are reusable to the total count of actual classes [25] that are highly reusable.
- R2 [23-25]: This is one of the techniques of goodness-of-fit. It represents the percentage of the variance of the dependent attribute to the percentage of the variance of the independent attribute.
- RSME [23-25]: It represents the average magnitude of overall errors [24, 25]. This focus is on a relatively large count of errors.

The results of the above statistics evaluated for the study are shown below with the help of Table 5.

Metric	Precision	Recall	R2	RMSE
LCOM1	0.645	0.545	0.514	0.181
LCOM2	0.827	0.750	0.100	0.211
LCOM3	0.729	0.659	0.210	0.208
LCOM4	0.577	0.721	0.012	0.208
LCOM5	0.567	0.785	0.010	0.198
TCC	0.562	0.800	0.007	0.200
DCD	0.691	0.800	0.101	0.194
CAMC	0.553	0.752	0.221	0.208
NHD	0.589	0.660	0.013	0.219
LAMCC	0.823	0.875	0.414	0.210
VMICM	1.00	1.00	0.977	0.172

The observations that can be analyzed from the table are:

- High values of precision, recall, and R2 for the proposed metric VMICM demonstrate that it exhibits high cohesion, which is the ultimate perspective of a highly reusable Object-Oriented (O-O) module.
- The low RSME value of the proposed metric VMICM shows that it is the most suitable cohesion metric among those previously proposed and that it causes fewer errors, which results in a highly reusable module.

From the above analysis, we can conclude that our proposed metric VMICM is the best cohesion metric that contributes to a highly reusable Object-Oriented (O-O) system which is the main objective of this study. Fig. 3 shows a graphical comparison of the proposed metric VMICM and other cohesion metrics.

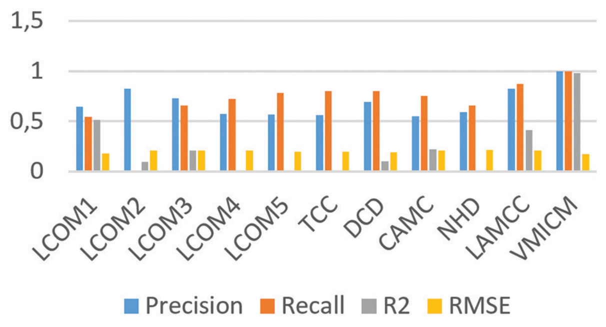


Fig. 3. Comparison of Univariate Statistics Results of Proposed Metric VMICM and other metrics

5. THREATS TO VALIDITY

Researchers suggest that there are various threats to the validity of any experimental analysis. We attempted to discuss the following threats in this section:

- Construct Validity [25]: This study deals with the measurement of variables used for this purpose. If these variables (dependent as well as independent) are accurately measured, we can say that they are valid constructively. This study even ensures the construct validity of dependent attributes.
- Internal Validity [25]: This deals with the relationship between cause and effect of both dependent attribute and independent attribute. If a particular study can establish effectively this relationship, we can conclude that it is valid internally.
- External Validity [25]: JAVA is used to implement the data set, i.e. all classes that are being considered. Applicability of the proposed cohesion metric should not be confined to one platform only, it should also apply to other Object-Oriented languages such as C++.

6. CONCLUSION AND FUTURE STUDY

This work has addressed the shortcomings of earlier proposed cohesiveness measurements. The study proposes a new class-based cohesion metric VMICM that accounts for four different types of cohesive interlinkages CRv, CMv, CRtv, and COv between variables and methods. Using theoretical and empirical analysis, the quality of the metric is also validated against several modules. The results showed that the proposed metric VMICM is the most appropriate cohesion-based metric contributing towards the highly reusable Object-Oriented (O-O) system. The theoretical validation of the proposed metric VMICM conforms with all the properties suggested by Briand, proving it to be a valid cohesion-based metric.

In the future, the proposed metric will be taken as an input dataset and then a hybrid data mining algorithm will be applied to uncover relationships between metrics values and different levels of reusability of a given class.

7. REFERENCES:

[1] K. Zozas, A. Ampatzoglou, S. Bibi, A. Chatzigeorgiou, P. Avgeriou, I. Stamelos, "REI: An integrated measure for software reusability", Journal of Software: Evolution and Process, Vol. 31, No. 8, 2019, pp. 1-18.

[2] Y. Zhou, Y. Mi, Y. Zhu, L. Chen, "Measurement and refactoring for package structure based on the complex network", Applied Network Science, Vol. 5, No. 50, 2020, pp. 1-20.

[3] Q. Sun, J. Wu, K. Liu, "Toward understanding students' learning performance in an object-oriented programming course: The perspective of program quality", IEEE Access, Vol. 8, No. 4, 2020, pp. 37505-37517.

[4] A. Gosain, G. Sharma, "Dynamic Metrics for Object-oriented Software Systems", IEEE Access, Vol. 7, No. 5, pp. 244-249.

[5] B. Mehboob, Y. Chun, "A metadata-driven process assessing for assessing stability and reusability based on the risk of change of software systems", Journal of Software: Practice and Experience, Vol. 53, No. 5, 2023, pp. 1218-1248.

[6] D. Singh, H. J. S. Sidhu, "Optimizing the software metrics for UML structural and behavioral diagrams using metrics tool", Asian Journal of Computer Science and Technology, Vol. 7, No. 2, 2018, pp. 11-17.

[7] A. M. Altaie, "Designing and implementing a tool for measuring cohesion and coupling of Object-Oriented Systems", Journal of Software: Practice and Experience, Vol. 13, No. 2, 2022, pp. 368-375.

[8] G. Rasool, Z. Arshad, "A review of code smell mining techniques", Journal of Software: Practice and Experience, Vol. 27, No. 11, 2015, pp. 867-895.

[9] J. Pantiuchina, M. Lanza, G. Bavota, "Improving Code: The (Mis) Perception of Quality Metrics", Proceedings of IEEE International Conference on Software Maintenance and Evolution, Madrid, Spain, 23-29 September 2018, pp. 80-91.

[10] T. M. Meyers, D. Binkley, "An empirical study of slice-based cohesion and coupling metrics", ACM Transactions on Software Engineering and Methodology, Vol. 17, No. 1, 2007, pp. 1-27.

[11] B. Mehboob, C. Y. Chong, S. P. Lee, J.M.Y. Lim, "Reusability affecting factors and software metrics for reusability: A systematic literature review", Journal of Software: Practice and Experience, Vol. 51, No. 6, 2021, pp. 1416-1458.

[12] S. Manhas, R. Vashisht, P. S. Sandhu, N. Neeru, "Reusability evaluation model for procedure-based software systems", International Journal of Computer and Electrical Engineering, Vol. 2, No. 6, 2010, pp. 1107-1111.

[13] S. Mal, K. Rajnish, "New class cohesion metric: An empirical view", International Journal of Multimedia and Ubiquitous Engineering. Vol. 9, No. 5, 2014, pp. 367-376.

[14] W. Schäfer, R. Prieto-Diaz, M. Matsumoto, "Software Reusability", 2nd Edition, Ellis Horwood, 1994.

[15] J. C. Esteva, R. G. Reynolds, "Identifying reusable software components by induction", International Journal of Software Engineering, Vol. 1, No. 4, 1991, pp. 271-292.

[16] G. Maheswari, K. Chitra, "Enhancing reusability and measuring performance merits of software component", International Journal of Innovative Technology and Exploring Engineering, Vol. 8, No.6, 2019, pp. 1577-1583.

[17] R. M. Andrianjaka, H. Razafimahatratra, T. Mahatody, M. Ilie, S. Ilie, R.N. Raft, "Automatic generation of software components of the Praxeme methodology from ReLEL", Proceedings of 24th International Conference on System Theory, Control, and Computing, Lunago, Spain, 24-28 November 2020, pp. 843-849.

[18] S. Khajenoori, D. G. Linton, C. A. Morris, “Enhancing software reusability through effective use of the essential modeling approach,” Information and Software Technology, Vol. 36, No. 11, 1994, pp. 495-501.

[19] D. Kansal, T. Aher, R. K. Joshi, “Sensitivity and Monotonicity in Class Cohesion Metrics”, Proceedings of the 12th Innovations on Software Engineering Conference, New York, USA, 10-15 February 2019, pp. 1-5.

[20] P. Kakkar, M. Sharma, P. Sandhu, “Modeling of Reusability of Procedure based Software Components using Naive Bayes Classifier Approach”, International Journal of Computer Applications, Vol. 55, No. 5, 2012, pp. 12-17.

[21] P. Gandhi, P. K. Bhatia, “Estimation of generic reusability for object-oriented software an empirical approach”, ACM SIGSOFT Software Engineering Notes, Vol. 36, No. 3, 2011, pp. 1-4.

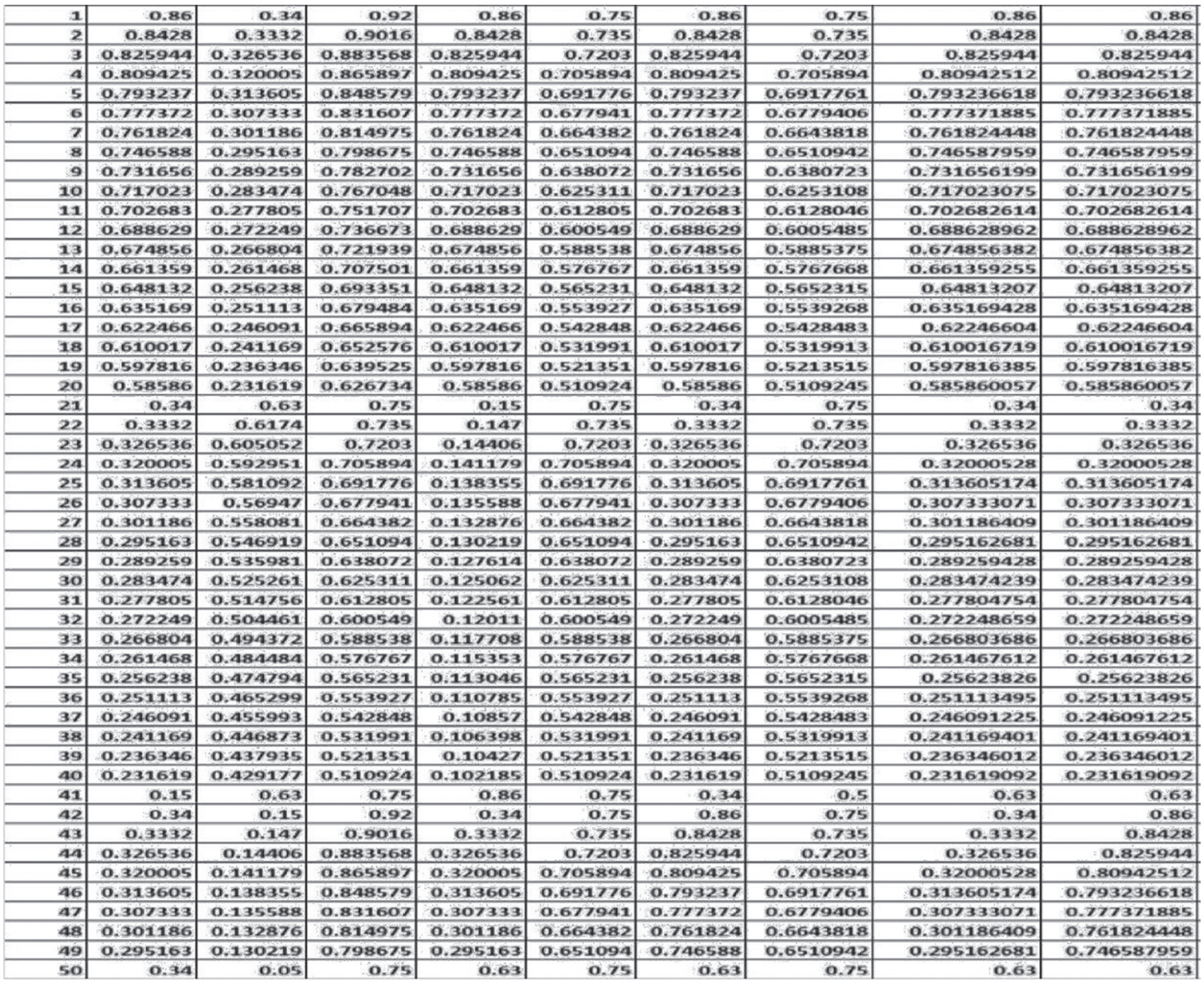
[22] A. Sharma, E. Al, “Maintainability Evaluation for Object-Oriented Software Metrics Using Tool Cohesion Inheritance (COIN)”, International Journal of Software Engineering, Vol. 12, No. 4, 2021, pp. 233-238.

[23] M. Alzahrani, S. Alqithami, A. Melton, “Using Client-Based Class Cohesion Metrics to Predict Class Maintainability”, Proceedings of IEEE 43rd Annual Computer Software and Applications Conference, New York, NY, USA, 15-20 June 2019, pp. 72-80.

[24] B. Bisht, P. Gandhi, “Software Reusability of Object-Oriented Systems using Data Mining Techniques”, International Journal of Recent Technology and Engineering, Vol. 7, No. 4, 2020, pp. 48-53.

[25] R. G. Alsarraj, A. M. Altaie, A. A. Fadhil, “Designing and implementing a tool to transform source code to UML diagrams”, Periodicals of Engineering and Natural Sciences, Vol. 9, No. 4, 2021, pp. 430-440.

8. APPENDIX



1	0.86	0.34	0.92	0.86	0.75	0.86	0.75	0.86	0.86
2	0.8428	0.3332	0.9016	0.8428	0.735	0.8428	0.735	0.8428	0.8428
3	0.825944	0.326536	0.883568	0.825944	0.7203	0.825944	0.7203	0.825944	0.825944
4	0.809425	0.320005	0.865897	0.809425	0.705894	0.809425	0.705894	0.80942512	0.80942512
5	0.793237	0.313605	0.848579	0.793237	0.691776	0.793237	0.6917761	0.793236618	0.793236618
6	0.777372	0.307333	0.831607	0.777372	0.677941	0.777372	0.6779406	0.777371885	0.777371885
7	0.761824	0.301186	0.814975	0.761824	0.664382	0.761824	0.6643818	0.761824448	0.761824448
8	0.746588	0.295163	0.798675	0.746588	0.651094	0.746588	0.6510942	0.746587959	0.746587959
9	0.731656	0.289259	0.782702	0.731656	0.638072	0.731656	0.6380723	0.731656199	0.731656199
10	0.717023	0.283474	0.767048	0.717023	0.625311	0.717023	0.6253108	0.717023075	0.717023075
11	0.702683	0.277805	0.751707	0.702683	0.612805	0.702683	0.6128046	0.702682614	0.702682614
12	0.688629	0.272249	0.736673	0.688629	0.600549	0.688629	0.6005485	0.688628962	0.688628962
13	0.674856	0.266804	0.721939	0.674856	0.588538	0.674856	0.5885375	0.674856382	0.674856382
14	0.661359	0.261468	0.707501	0.661359	0.576767	0.661359	0.5767668	0.661359255	0.661359255
15	0.648132	0.256238	0.693351	0.648132	0.565231	0.648132	0.5652315	0.64813207	0.64813207
16	0.635169	0.251113	0.679484	0.635169	0.553927	0.635169	0.5539268	0.635169428	0.635169428
17	0.622466	0.246091	0.665894	0.622466	0.542848	0.622466	0.5428483	0.62246604	0.62246604
18	0.610017	0.241169	0.652576	0.610017	0.531991	0.610017	0.5319913	0.610016719	0.610016719
19	0.597816	0.236346	0.639525	0.597816	0.521351	0.597816	0.5213515	0.597816385	0.597816385
20	0.58586	0.231619	0.626734	0.58586	0.510924	0.58586	0.5109245	0.585860057	0.585860057
21	0.34	0.63	0.75	0.15	0.75	0.34	0.75	0.34	0.34
22	0.3332	0.6174	0.735	0.147	0.735	0.3332	0.735	0.3332	0.3332
23	0.326536	0.605052	0.7203	0.14406	0.7203	0.326536	0.7203	0.326536	0.326536
24	0.320005	0.592951	0.705894	0.141179	0.705894	0.320005	0.705894	0.32000528	0.32000528
25	0.313605	0.581092	0.691776	0.138355	0.691776	0.313605	0.6917761	0.313605174	0.313605174
26	0.307333	0.56947	0.677941	0.135588	0.677941	0.307333	0.6779406	0.307333071	0.307333071
27	0.301186	0.558081	0.664382	0.132876	0.664382	0.301186	0.6643818	0.301186409	0.301186409
28	0.295163	0.546919	0.651094	0.130219	0.651094	0.295163	0.6510942	0.295162681	0.295162681
29	0.289259	0.535981	0.638072	0.127614	0.638072	0.289259	0.6380723	0.289259428	0.289259428
30	0.283474	0.525261	0.625311	0.125062	0.625311	0.283474	0.6253108	0.283474239	0.283474239
31	0.277805	0.514756	0.612805	0.122561	0.612805	0.277805	0.6128046	0.277804754	0.277804754
32	0.272249	0.504461	0.600549	0.12011	0.600549	0.272249	0.6005485	0.272248659	0.272248659
33	0.266804	0.494372	0.588538	0.117708	0.588538	0.266804	0.5885375	0.266803686	0.266803686
34	0.261468	0.484484	0.576767	0.115353	0.576767	0.261468	0.5767668	0.261467612	0.261467612
35	0.256238	0.474794	0.565231	0.113046	0.565231	0.256238	0.5652315	0.25623826	0.25623826
36	0.251113	0.465299	0.553927	0.110785	0.553927	0.251113	0.5539268	0.251113495	0.251113495
37	0.246091	0.455993	0.542848	0.10857	0.542848	0.246091	0.5428483	0.246091225	0.246091225
38	0.241169	0.446873	0.531991	0.106398	0.531991	0.241169	0.5319913	0.241169401	0.241169401
39	0.236346	0.437935	0.521351	0.10427	0.521351	0.236346	0.5213515	0.236346012	0.236346012
40	0.231619	0.429177	0.510924	0.102185	0.510924	0.231619	0.5109245	0.231619092	0.231619092
41	0.15	0.63	0.75	0.86	0.75	0.34	0.5	0.63	0.63
42	0.34	0.15	0.92	0.34	0.75	0.86	0.75	0.34	0.86
43	0.3332	0.147	0.9016	0.3332	0.735	0.8428	0.735	0.3332	0.8428
44	0.326536	0.14406	0.883568	0.326536	0.7203	0.825944	0.7203	0.326536	0.825944
45	0.320005	0.141179	0.865897	0.320005	0.705894	0.809425	0.705894	0.32000528	0.80942512
46	0.313605	0.138355	0.848579	0.313605	0.691776	0.793237	0.6917761	0.313605174	0.793236618
47	0.307333	0.135588	0.831607	0.307333	0.677941	0.777372	0.6779406	0.307333071	0.777371885
48	0.301186	0.132876	0.814975	0.301186	0.664382	0.761824	0.6643818	0.301186409	0.761824448
49	0.295163	0.130219	0.798675	0.295163	0.651094	0.746588	0.6510942	0.295162681	0.746587959
50	0.34	0.05	0.75	0.63	0.75	0.63	0.75	0.63	0.63