Date of publication JUN-30, 2022, date of current version AUG-01, 2021.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.21.2.2587

Software Reusability Estimation based on Dynamic Metrics using Soft Computing Techniques

MANJU DUHAN, PRADEEP KUMAR BHATIA
Guru Jambheshwar University of Science &Technology, Hisar
(e-mail: duhan.manju@gmail.com, pkbhatia.gju@gmail.com)

Corresponding author: Manju Duhan (e-mail: duhan.manju@gmail.com).

ABSTRACT Dynamic metrics capture the run time features of object-oriented languages, i.e., runtime polymorphism, dynamic binding, etc., that are not covered by static metrics. Therefore, in this paper, we derived a new approach to measuring the software reusability of a design pattern based on dynamic metrics. To achieve this, the authors proposed a model based on five parameters, i.e., polymorphism, inheritance, number of children, coupling, and complexity, to measure the reusability factor by using various soft computing techniques, i.e., Fuzzy, Neural Network, and Neuro-Fuzzy. Further, we also compared the proposed model with four existing machine learning algorithms. Lastly, we found that the proposed model using the neuro-fuzzy technique is trained well and predicts well with MAE (Mean absolute error) 0.003 and RMSE (Root mean square error) 0.009 based on dynamic metrics. Hence, it is concluded that dynamic metrics are a better predictor of the reusability factor than static metrics.

KEYWORDS Neuro-Fuzzy system; Fuzzy system; Software Reusability; Neural Network Model; Dynamic metrics; Dynamic Polymorphism.

I. INTRODUCTION

SOFTWARE Reusability is an attribute of quality in which a new software system is implemented from the pre-existing software system. Reusability of code plays a vital role in improving software quality in terms of cost reduction, less development time and more reliability. To achieve this, object-oriented (OO) metrics play a crucial role in finding a reusable class code [1]. Many researchers served on various measures of metrics like coupling, cohesion, complexity, etc., to identify the OO design quality. However, most of the existing investigations were based on static metrics to measure the quality of software. But in modern OO software, there are features like run time polymorphism, dynamic binding, runtime complexity, etc., that are captured using dynamic metrics. So, the software industry's focus moves from static metrics to dynamic metrics to measure software reusability in advance. Therefore, in this paper, we have used dynamic metrics instead of static metrics to design a software reusability prediction model (SRPM) using soft computing techniques.

In this paper, we mainly focus on five factors i.e., Polymorphism, complexity, Inheritance, number of children, and, coupling to measure the class reusability based on the correlation of these factors with the output i.e., reusability. The first factor we decide to measure the reusability factor is a polymorphism that allows code-sharing and systematic reusing of code. Polymorphism means having the ability to take several forms [2]. There are mainly two types of polymorphism that exist in Object-Oriented languages. First is compile-time or static polymorphism, which achieves by function overloading or constructor overloading, where overloading means using the same name with a different signature. For measuring this factor, we took SP (Static Polymorphism) metric. Another one is runtime or dynamic polymorphism, which reaches by virtual functions or dynamic binding functions in OO programming languages [3]. In dynamic binding, the concept of overriding functions comes, which means a new definition given to the derived class function that exists with the same name in the base class. For measuring this factor, we took DP (Dynamic Polymorphism) metric [4]. Also, we took four static metrics from the famous Chidamber and Kemerer, CK metric suite [5] i.e WMC (Weighted Method Complexity) for complexity measure, DIT (Depth of Inheritance Tree) for inheritance measure, NOC (Number of Children) for the number of children measure and CBO (Coupling Between Objects) for coupling measure in OO environment. We have also used DCBO (Dynamic Coupling between Objects) from Mitchell and Power suite [6] and

DWMC (Dynamic Weighted Method Complexity) from Manju and Bhatia metrics suite [4]. Table 1 and Table 2 show the description of static and dynamic metrics used in the current study to measure the reusability factor.

Table 1. Static Metrics Description

Metric	Description
SP	"Ratio of number of overloading and overridden methods in class to the total number of methods in class"
WMC	"Sum of the complexity of the methods of a class"
DIT	"The maximum length from the node to the root of the tree"
NOC	"Number of immediate subclasses subordinated to a class in the class hierarchy"
CBO	"Count of the number of other classes to which it is coupled"

Table 2. Dynamic Metrics Description

Metric	Description
DP	"Ratio of number of overloading and overridden methods executed at runtime to the total number of times methods of class execute at runtime"
DWMC	"Number of times methods of class executed at runtime"
DIT	"The maximum length from the node to the root of the tree at run time"
NOC	"Number of immediate subclasses subordinated to a class in the class hierarchy at run time"
DCBO	"Count of the number of other classes to which it is coupled at runtime"

Further, we cannot predict the reusability of a class by using crisp logic, i.e., yes or no. Therefore, we need fuzzy logic (FL) to say the reusability of a class is low, medium, or high for flexibility. Hence, fuzzy logic allows us to predict the reusability of a class in the form of low, medium, and high rather than crisp logic. In addition, Neural Network (NN) approach provides adaptive learning capabilities to predict the reusability ranking of software, whereas fuzzy logic can generalize rules. Therefore, this study evaluates the reusability of object-oriented software systems using the fuzzy, neural network, and neuro-fuzzy (NF) approach [7]. Lastly, we have also used four famous existing machine learning algorithms [8, 9] that are described below.

A. RANDOM FOREST (RF)

Random Forest is a supervised learning algorithm used in both classification and regression but is most commonly used in classification [8].

B. LINEAR REGRESSION (LR)

The Linear Regression model is based on the concept of best fit by finding the relationship between attributes. This algorithm also comes under the category of supervised learning classification algorithms [9].

C. SMOreg

The sequential Minimal Optimization Regression (SMOreg) algorithm performed best for solving quadratic programming problems to train support vector machines [10].

D. MULTILAYER PERCEPTRON (MLP)

MLP is a supervised learning classification algorithm that uses a back propagation algorithm for the training of attributes feed as input to an artificial neural network [10].

The rest of the paper organizes as follows: Section 2 summarizes the literature review, and section 3 presents the research methodology followed in this paper. Section 4 described the proposed formula and proposed models to measure reusability using soft computing techniques. Section 5 explains the experimental study done on 140 samples and the validation of models using the 2 test dataset. Lastly, Section 6 describes the conclusion and future directions referenced in this paper.

II. RESEARCH WORK

Many empirical studies exist in the literature to validate the strong relationship between design patterns and respective reusability. Zahara et al., 2013 [9] compared MLR, M5P, IBk and Additive Regression algorithms for reusability evaluation and found that IBk outperforms compared to other MLA with no distance weighting using the WEKA tool [12]. Pathy et al., 2015 [13] reviewed the static reusability metrics and concluded that reusability increases then reciprocally increase the DIT and NOC. Godara et al., 2016 [14] proposed an adaptive neuro-fuzzy inference system (ANFIS), SRPM based on static and traditional metrics and concluded that the hybrid neuro-fuzzy model trained well and gives satisfactory results for reusability measurement of software. Adekola et al., 2017 [15] proposed a new set of design principles for object-oriented and other reuse-oriented systems. They also explained how using internal attributes as cohesion could improve software maintainability and reusability. Papamichail et al., 2018 [8] proposed MLA to predict reusability and found that RF outperforms as compared to other MLA. Papamichail et al., 2019 [16] proposed a new dataset to measure the reuse rate of software components using the Maven repository with the help of SourceMeter and the AGORA tool. They concluded that the generated dataset effectively measures the reusability of software components based on static metrics. Mangayarkarasi et al., 2020 [17] proposed NN, and SRPM based on static and traditional metrics and found that the proposed mathematical model effectively measures the software cost with design reusability. Godara et al., 2021 [18] reviewed various SRPMs and concluded that most of the model's base is static metrics and more work is required to build SRPMs using various soft computing techniques. Hence, from the literature, it is clear that most of the studies conducted yet are based on static metrics i.e. mostly on CK metrics suite [19] [20]. No study is conducted till now based on dynamic metrics that have equal importance as static metrics to measure run-time features of object-oriented languages i.e. run-time polymorphism, dynamic binding etc. to find SR [21, 22]. Therefore, in this paper, we tried to find the reusability factor based on dynamic metrics to cover run time features of object-oriented languages using soft computing techniques i.e. FL, NN and NF.

III. RESEARCH METHODOLOGY

Fig. 1 presents the methodology adopted in the current study. 18 design patterns[1] were used in the study, having 92 classes to collect static and dynamic metrics data. Static metrics were collected from the CodeMR tool[2], and dynamic metrics were measured using the AspectJ tool[3], an implementation of aspect-oriented programming[4] on the eclipse platform[5]. After

[1] https://www.Geeksforgeeks.org/chain-responsibility-design-pattern/
[2] http://www.codemr.co.uk/
[3] http://www.eclipse.org/aspectj
[4]https://o7planning.org/en/10257/ java-aspect-oriented-programming-tutorial-

successfully collecting data, the proposed fuzzy model fed with static and dynamic metric data as input to measure design pattern reusability. Further, the proposed FL model was validated using the AHP technique [23]. After that, we took 140 samples from the Fuzzy model and fed them as input to the proposed neural network and neuro-fuzzy model. MATLAB tool was used to develop proposed models, i.e., FL, NN, and NF. Further, the authors compared the proposed model with the existing four machine learning algorithms, i.e., LR, MLP, RF, and SMOreg, based on predictive accuracy measures, i.e., MAE and RMSE [11]. We have used WEKA 3.8 tool[6] to measure MAE and RMSE values of four machine learning algorithms.

IV. PROPOSED WORK

In this section, we describe the proposed formula, working of proposed fuzzy logic (FL), neural network (NN), and neuro-fuzzy (NF) model with machine learning algorithms used in the current study in various subsections.

A. PROPOSED FORMULA

A formula is proposed based on metrics described in section 1. Our approach is to discover the formula to measure the reusability of a class diagram based on the following beliefs:

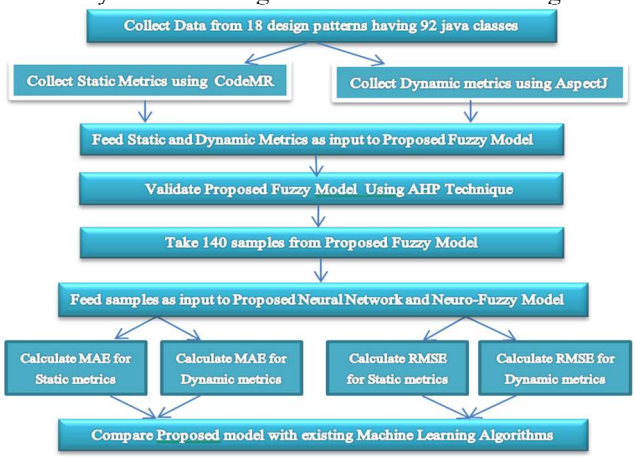

Figure 1. Research Methodology followed in Current Study

- More eminent the polymorphism factors in class, the greater the potential for reuse of code because polymorphism comes after inheritance. So, SP and DP have a decisive impact on reusability.
- The deeper a particular class is in the hierarchy, the greater the potential for reusing inherited methods. It declares that the reusability of a class increases with an increase in the DIT of a class. So, DIT has a definite impact on class reusability.
- A nominal value for NOC intimates the extent of reuse. Up to a particular threshold value, NOC has a convincing impact on class reusability.
- Excessive coupling indicates a weakness of class encapsulation and may inhibit reuse. It demonstrates that coupling (CBO and DCBO) harms the reusability of a class.
- **Excessive** complexity indicates the class is highly complex and hard to maintain. Therefore, WMC and DWMC harm class reusability.

Reusability of a class based on static metrics (RS) =

$$w_1*(SP)+w_2*(DIT)+w_3*(NOC)-w_4*(CBO)-w_5*(WMC). \quad (1)$$

Reusability of a class based on dynamic metrics (RD) =

$$w_1*(DP)+w_2*(DIT)+w_3*(NOC)-w_4*(DCBO)-w_5*(DWMC). \quad (2)$$

where w_1, w_2, w_3, w_4, and w_5 are the weights to be calculated using AHP (Analytic Hierarchy Process) technique in section 5.1.

Thus, the reusability of a class diagram is the class having maximum reusability at both compile and run time. Thus, Reusability of class diagram =

$$max(\sum_{i=1}^{n} reusability(class)_i), \quad (3)$$

where n is the total number of classes in the class diagram.

B. PROPOSED FUZZY LOGIC MODEL (FL)

The proposed FL model developed using the MATLAB tool takes five inputs, i.e., polymorphism, inheritance, number of children, coupling, complexity to the MFIS (Mamdani's Fuzzy Inference System), and one output, i.e., reusability, as shown in Fig. 2. We classify all inputs into fuzzy sets viz., Low, Medium, and High. Therefore, according to the formula, 3^5 rules were developed [4]. In addition, to fuzzify the inputs, the triangular membership function (trimf) has been chosen, namely Low, Medium and High. The output variable, i.e., software reusability, has five membership functions (trimf), namely Very Low, Low, Medium, High, and Very High as shown in Fig. 3. The proposed model considers all five inputs and provides a crisp value of software reusability using a rule base. The range of inputs is different in the case of static metrics and dynamic metrics. For static metrics ranges of input are SP[0-1], WMC[0-9], DIT[0-2], NOC[0-3], CBO[0-4] respectively. For dynamic metrics ranges of input are DP[0-1], DWMC[0-14], DIT[0-2], NOC[0-3], DCBO[0-4] respectively.

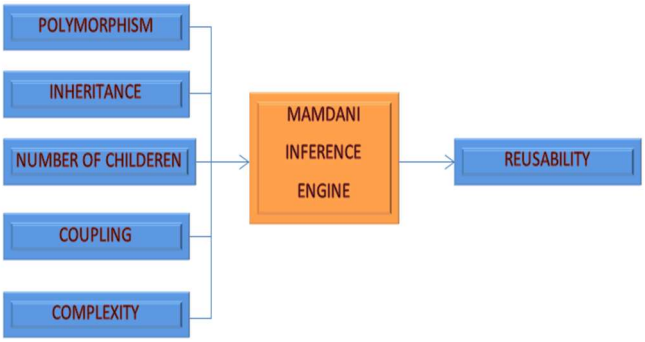

Figure 2. Proposed Fuzzy Model

with-aspectj

[5] https://www.eclipse.org/aspectj/doc/next/progguide/printable.html

[6]http://people.sabanciuniv.edu/berrin/cs512/lectures/WEKA/ WEKA% 20 Explorer % 20 Tutorial-REFERENCE.pdf

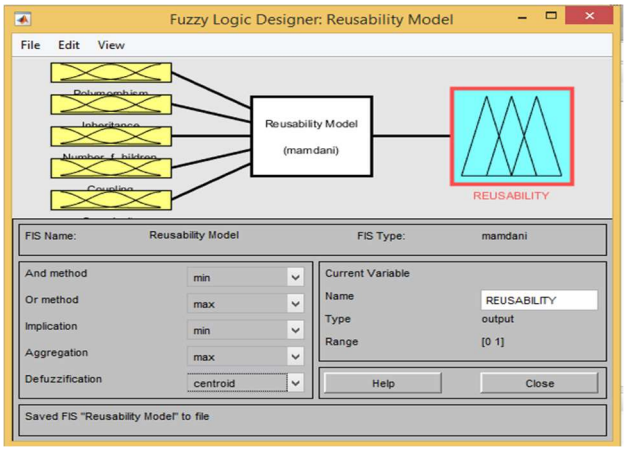

Figure 3. Proposed Mamdani's Fuzzy Inference System

C. NEURAL NETWORK MODEL (NN)

The proposed neural network, as described in Table 3, takes input generated from the proposed fuzzy model. The network has trained with five inputs by using trainlm as training functions and one output, i.e., software reusability. We set the number of neurons at the hidden layer to 20. The Transig function was used as a transfer function in the proposed ANN. The ANN was trained on these data sets (140 samples of different datasets generated by the fuzzy model) by the standard error back-propagation algorithm at a learning rate of 0.006, having the mean squared error as the training stopping criterion. The network divides the 140 samples into three parts, i.e., 98 samples (70%) for training, 21 samples (15%) for testing, and 21 samples (15%) for validating the neural network, as shown in Fig. 4.

Table 3. Proposed neural network Description

Input units	05
Output units	01
No. of neurons at the hidden layer	20
Algorithm	Back propagation
Training function	Trainlm
Network ratio	14:3:3

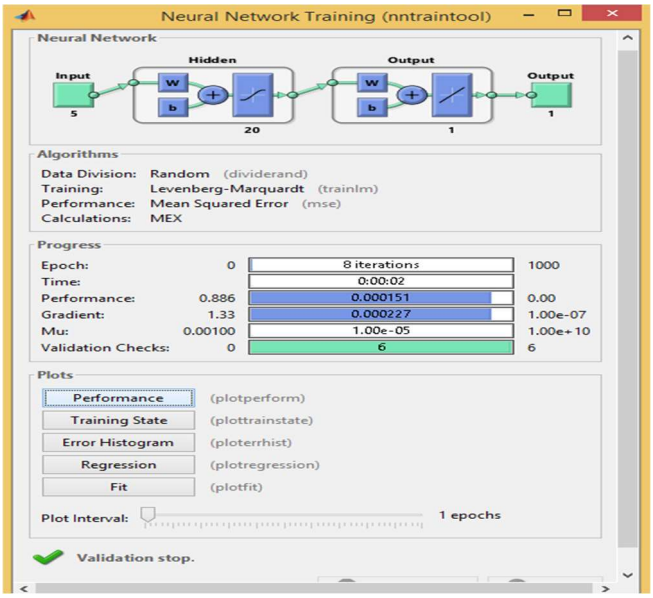

Figure 4. Proposed Neural network Model

B. PROPOSED NEURO-FUZZY MODEL (NF)

Neuro-fuzzy is a hybrid system [10], as described in Table 4, that benefits from a fuzzy logic approach that provides flexibility to a system rather than crisp logic and a neural network that can learn by itself. Therefore we can say that the neuro-fuzzy approach is a mixture of implicit and explicit knowledge. In this paper, the neuro-fuzzy model is applied with the help of the MATLAB tool using the ANFIS editor.

Table 4. Proposed Neuro-Fuzzy Model Description

Input units	05
Output units	01
No of the train data pairs	140
No. of Fuzzy rules generated	243
FIS Model	Sugeno
FIS training optimization method	Hybrid
FIS Input membership function	gaussmf(Low, Medium, High)
FIS Output membership function	Linear
FIS Generation method	Grid Partitioning
No. of Epochs	60

Firstly, raw data was loaded in the ANFIS editor using the load data option and set the type of data as training data in the load data part of the ANFIS editor, as shown in Fig. 5. After the successful loading of data, we can see the structure of the proposed neuro-fuzzy model. After that, the authors generated a Sugeno fuzzy inference system with three membership functions (gaussmf), i.e., low, medium, and high, using the grid partitioning method and then train the generated FIS using a hybrid optimization method by setting the number of epochs to 60. Further, trained ANFIS was tested on two validation datasets by selecting the load data type for testing, and again, the authors generated the FIS on testing data.

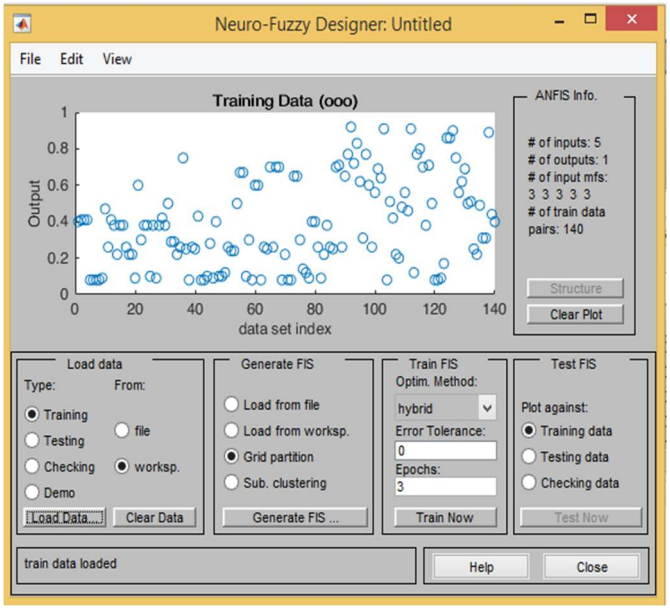

Figure 5. Loaded data in the Neuro-fuzzy model

E. MACHINE LEARNING ALGORITHMS USED IN THE CURRENT STUDY

The authors have used four MLA, i.e., LR, MLP, RF, and SMOreg, in the current study to find the reusability factor of software. The authors applied these algorithms by using WEKA 3.8 tool. After that, the classification of processed data has done by setting the cross-validation folds value to 30.

From the results, the authors observed that the time taken to build a model using WEKA 3.8 is significantly less than the fuzzy and neuro-fuzzy models.

V. EXPERIMENTAL STUDY

An experimental study has been conducted on 18 Design patterns coded in Java programming language. Statistics were applied to obtain values of metrics using the MATLAB tool. Simple functions, i.e., min, max, mean, median, std, are used in MATLAB to find the minimum, maximum, mean, median, and standard deviation of measured metrics values. Table 5 and Table 6 show the statistical results on metrics values of 92 classes taken from 18 design patterns.

Table 5. Statistical data of static Metrics

Metric Name	MIN.	MAX.	MEAN	MEDIAN	STD.DEV.
SP	0	1	0.43	0.5	0.40
DIT	0	2	0.63	1	0.56
NOC	0	3	0.58	0	0.92
CBO	0	4	0.78	0	1.08
WMC	1	9	0.23	2	1.65

Table 6. Statistical data of Dynamic Metrics

Metric Name	MIN.	MAX.	MEAN	MEDIAN	STD.DEV.
DP	0	1	0.36	0.23	0.39
DIT	0	2	0.60	1	0.53
NOC	0	3	0.55	0	0.90
DCBO	0	4	0.70	0	1.01
DWMC	0	14	3.08	2	3.33

To find the reusability of a system or design pattern, we first divide the static and dynamic metrics values by the number of components or classes in the system or design pattern, respectively. After that, processed data feed as input to the fuzzy model. In contrast, for reusability measurement of a class, no pre-processing of static and dynamic metrics values is required. Table 7 shows the reusability of 18 design patterns measured by proposed MFIS based on static and dynamic metrics where NC denotes the number of components in the design pattern.

Table 7. Reusability Measured by the Proposed MFIS

Class Name	SP	DP	DIT	NOC	CBO	WMC	DWMC	NC	Static MFIS	Dynamic MFIS
Abstract Factory	1.64	1.84	3	3	16	17	31	8	0.42	0.41
Adaptor	2.5	1.75	3	3	1	8	8	5	0.64	0.50
Bridge	3	1.5	4	4	3	9	16	5	0.64	0.40
Builder	2.6	2.6	3	3	5	28	19	6	0.55	0.55
Composite	1.33	1.25	3	3	0	8	20	4	0.55	0.50
Iterator	0.99	1.08	2	2	8	13	21	6	0.38	0.29
Factory Method	2	1	2	2	3	5	3	4	0.69	0.65
FlyWeight	1.32	1.80	2	2	3	12	32	4	0.50	0.40
Chain	3	2.29	3	3	4	10	16	5	0.55	0.44
Observer	1.33	0.94	3	3	2	17	22	5	0.47	0.33
ProtoType	1.5	1.12	5	2	1	7	20	4	0.75	0.70
State	2	2	2	2	5	6	11	4	0.53	0.53
Strategy	5.5	3.5	7	7	8	20	12	10	0.69	0.64
Template	2	2	2	2	0	13	18	3	0.64	0.59
Visitor	2.11	1.2	3	3	7	15	14	5	0.39	0.39
Mediator	1.83	1.83	3	3	6	15	11	5	0.33	0.33
Employee	1.83	1.5	4	3	0	7	5	4	0.64	0.67
Car Data	3	3	4	3	0	7	5	4	0.84	0.86

After measuring the static and dynamic metrics, we found that the value of the coupling measure, CBO and DCBO gives almost the same results. In the same way, there is very little variance in the values of DIT and NOC metrics values at compile and run time. Therefore, in table 7, we show only static values of DIT, NOC and CBO metrics. From table 7, we can say that there is a significant difference between the reusability factor measured by MFIS based on static, and dynamic metrics.

A. VALIDATION OF PROPOSED MFIS USING AHP

The proposed fuzzy model is validated using the standard AHP (Analytic Hierarchy Process) technique given by Saaty [19]. Table 8 shows the calculated values of 5 factors: polymorphism, inheritance, number of children, coupling, and complexity. If the consistency index value is less than 0.1, then the decision value of 5 factors is accepted; otherwise, it's rejected. Hence, Eigen vector values are used as the final weight value to the corresponding factor, i.e., w_1 is 0.46 for polymorphism, w_2 is 0.24 for an inheritance, w_3 is 0.16 for the number of children, w_4 is 0.11 for coupling, and w_5 is 0.03 for complexity.

Consistency Index $(CI) = (\lambda max - n)/(n\text{-}1)$ where n=5.

From Table 8, we got the values of w_1, w_2, w_3, w_4, and w_5. Further, we put these weight values in equations (1) and (2) described in section 4.1 and calculated class reusability based on static and dynamic metrics.

Table 8. AHP Decision Values

Factors	Poly.	Inhe.	NOC	Coup.	Comp.	Eigen Vector (w)
Polymorphism	1	3	3	7	9	0.46
Inheritance	1/3	1	3	3	7	0.24
NOC	1/3	1/3	1	3	7	0.16
Coupling	1/7	1/3	1/3	1	7	0.11
Complexity	1/9	1/7	1/7	1/7	1	0.03
Total						1.000

After that, we calculated relative error (RE) and relative root square error (RRSE). For static metrics, the measured value of MAE is 0.036, and RMSE is 0.11. For dynamic metrics, the measured value of MAE is 0.026, and RMSE is 0.02. The resulted values indicate that dynamic metrics can better predict the reusability of design patterns.

B. ANALYSIS RESULTS

Data generated by the fuzzy model act as input to the neural network and neuro-fuzzy model. We took 140 samples from the MFIS model to feed as input to the NN and NF model. After successful training of the neural network and neuro-fuzzy model, the value of MAE and RMSE was obtained. In addition, a comparison of proposed models has been done with the existing four machine learning algorithms on the predictive accuracy measures, i.e., MAE and RMSE. Table 9 shows the comparison of various SRPMs.

Table 9. Comparison of various SRPMs

		Static Metrics		Dynamic Metrics	
SRPM	Model Name	MAE	RMSE	MAE	RMSE
Proposed Model	FL	0.08	0.10	0.05	0.07
	NN	0.02	0.04	0.009	0.03
	NF	0.01	0.02	0.003	0.009
MLA	SMOreg	0.06	0.08	0.07	0.09
	RF	0.04	0.07	0.05	0.08
	LR	0.06	0.07	0.07	0.09
	MLP	0.06	0.08	0.05	0.07

The highest value of MAE is 0.08 for static metrics and 0.05 for dynamic metrics. This indicates that the fuzzy model has the highest difference between predicted and actual values. The lowest value of MAE is 0.01 for static metrics and 0.003 for dynamic metrics using the neuro-fuzzy model which indicates that the proposed neuro-fuzzy model is trained well and has minimized the differences between predicted and actual values. Fig. 6 shows a comparison of MAE values for the static and dynamic metrics of SRPM.

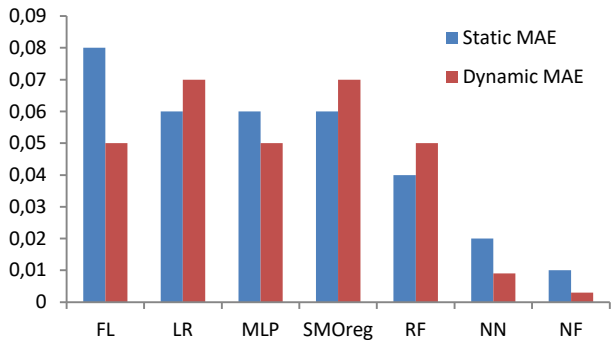

Figure 6. Comparison of MAE values of various SRPMs.

C. VALIDATION OF PROPOSED MODELS

The proposed model has been validated on two datasets. The result indicates that the neuro-fuzzy model trained well, which gave MAE 0.000 for dataset 1 based on dynamic metrics. Table 10 and Table 11 show validation results for static and dynamic metrics on two datasets, respectively.

Table 10. Validation results for Static Metrics

	Static metrics			
	Test Dataset1		Test Dataset 2	
Proposed Model	MAE	RMSE	MAE	RMSE
FL	0.076	0.082	0.081	0.098
NN	0.019	0.026	0.032	0.052
NF	0.007	0.013	0.004	0.009

Table 11. Validation results for Dynamic Metrics

	Dynamic metrics			
	Test Dataset1		Test Dataset 2	
Proposed Model	MAE	RMSE	MAE	RMSE
FL	0.055	0.064	0.048	0.062
NN	0.003	0.008	0.016	0.047
NF	0.000	0.002	0.001	0.006

Therefore, from Table 10 and Table 11, we can say that the neuro-fuzzy model performed well on both the test datasets irrespective of any model based on dynamic metrics. We compared the MAE values of dataset 1 and dataset 2, and we found that the value of error is minimized in the models based on dynamic metrics. The minimum value of MAE is 0.000, and RMSE is 0.001 for the neuro-fuzzy model based on dynamic metrics for test dataset 1, as shown in Fig. 7 and Fig. 8, respectively.

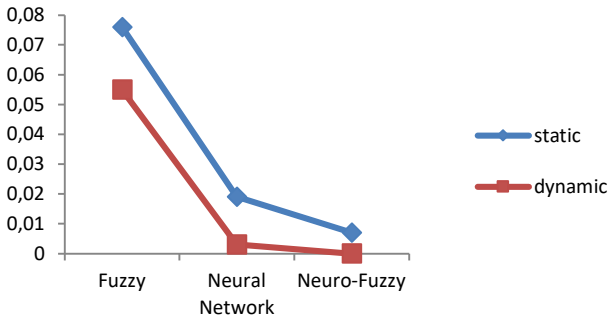

Figure 7. Comparison of MAE values of proposed models on dataset 1

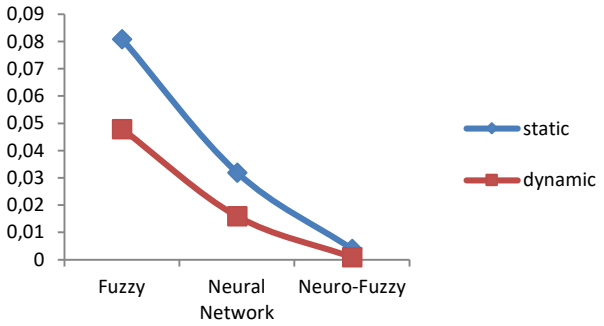

Figure 8. Comparison of MAE values of proposed models on dataset 2

VI. CONCLUSION AND FUTURE WORK

This paper proposed a model having five inputs, i.e., polymorphism, inheritance, number of children, coupling, complexity, and one output, i.e., reusability. We have also offered a formula based on these inputs to evaluate the reusability factor based on static and dynamic metrics. We have taken 140 samples from the web to find the reusability of a class. Further, the authors compared the proposed model with four machine learning algorithms, and we found that the neuro-fuzzy model is trained well and gives MAE 0.01 in the case of static metrics and 0.003 in the case of dynamic metrics. Also, the authors validated the proposed fuzzy, neural network, and neuro-fuzzy model on 2 test datasets. The result shows that the neuro-fuzzy model gives

satisfactory results with MAE 0.000 on dataset 1 based on dynamic metrics. Hence, we can say that dynamic metrics are a better predictor of the reusability factor than static metrics. Therefore, this model is beneficial for the software industry to predict the reusability of Object-Oriented software systems that would be very beneficial for developers about cost benefits. Static metrics-based SRPM can be used in the early phases of the software development life cycle. In contrast, dynamic metrics-based SRPM can be used after the coding phase. In future, the model will be more refined by considering other object-oriented metrics and large projects to calculate a project's reusability in advance quickly.

References

[1] R. S. Pressman, *Software Engineering – A Practitioner's Approach*, 7th ed., McGraw Hill, 2005.

[2] S. Benlarbi, W. L. Melo, "Polymorphism measures for early risk prediction," *Proceedings of the International Conference on Software Engineering*, 1999, pp. 334-344. https://doi.org/10.1145/302405.302652.

[3] K. H. T. Choi, E. Tempero, "Dynamic measurement of polymorphism," *Proceedings of the Thirtieth Australasian Computer Science Conference,* Victoria, Australia, 2007, vol. 62, pp. 211-220.

[4] Manju, P. K. Bhatia, "Empirical validation of dynamic metrics using knowledge based approach," *International Journal of Advanced Research in Engineering and Technology*, vol. 11, issue 12, pp. 3219-3230, 2020.

[5] S. R. Chidamber, C. F. Kemerer, "A metrics suite for object oriented design," *IEEE Transactions on Software Engineering*, vol. 20, issue 6, pp. 476-493, 1994. https://doi.org/10.1109/32.295895.

[6] A. Mitchell, J. F. Power, "A study of the influence of coverage on the relationship between static and dynamic coupling metrics," *Science of Computer Programming*, vol. 59, issue (1/2), pp. 4-25, 2006. https://doi.org/10.1016/j.scico.2005.07.002.

[7] H. Lounis, T. Gayed, M. Boukadoum, "Using efficient machine-learning models to assess two important quality factors: Maintainability and reusability," *Proceedings of the 2011 Joint Conference of the 21st International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement*, pp. 170-177, 2011, https://doi.org/10.1109/IWSM-MENSURA.2011.44.

[8] M. Papamichail, T. Diamantopoulos, I. Chrysovergis, P. Samlidis, A. Symeonidis, "User perceived reusability estimation based on analysis of software repositories," *Proceedings of the IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE)*, 2018, pp. 49–54. https://doi.org/10.1109/MALTESQUE.2018.8368459.

[9] R. Feldt, F. G. Neto, R. Torkar, "Ways of applying artificial intelligence in software engineering," *Proceedings of the 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering*, 2018, pp. 35–41.

[10] D. Stefano, T. Menzies, "Machine learning for software engineering: case studies in software reuse," *Proceedings of 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2002)*, pp. 246–251, 2002.

[11] S. I. Zahara, M. Ilyas, T. Zia, "A study of comparative analysis of regression algorithms for reusability evaluation of object-oriented based software components," *Proceedings of 2013 International Conference on Open-Source Systems and Technologies*, 2013, pp. 75–80. https://doi.org/10.1109/ICOSST.2013.6720609.

[12] S. S. Aksenova, *Machine Learning with WEKA*, WEKA Explorer Tutorial, 2004.

[13] N. Padhy, R. Panigrahi, S. Baboo, "A systematic literature review of an object oriented metric: reusability," *Proceedings of the International Conference on Computational Intelligence and Networks*, Bhubaneswar, 2015, pp. 190–191. https://doi.org/10.1109/CINE.2015.44.

[14] D. Godara, O. P. Sangwan, "Neuro-fuzzy based approach to software reusability estimation," *Proceedings of the International Conference on Sustainable Computing Techniques in Engineering Science and Management (IJCTA)*, pp. 3811-3891, 2016.

[15] O. D. Adekola, S. A. Idowu, S. O. Okolie, J. V. Joshua, A. O Akinsanya, M. O. Eze, E. Seun, "Software maintainability and reusability using cohesion metrics," international journal of computer trends and technology (ijctt), vol. 54, pp. 63–73, 2017. https://doi.org/10.14445/22312803/IJCTT-V54P111.

[16] M. Papamichail, T. Diamantopoulos, A. Symeonidis, "Software reusability dataset based on static analysis metrics and reuse rate information," *Journal of System and Software*, vol. 27, 104687, 2019, https://doi.org/10.1016/j.dib.2019.104687.

[17] P. Mangayarkarasi, R. Selvarani, "Dynamic reusability prediction model for SMEs based on realtime constraints," *International Journal of Engineering Trends and Technology – Special Issues*, pp. 63-75, 2020.

[18] D. Godara, O. P. Sangwan, "Software reusability estimation using machine learning techniques – A systematic literature review," *Proceedings of the Evolving Technologies for Computing, Communication and Smart World, Lecture Notes in Electrical Engineering*, Springer, Singapore, vol. 694, pp. 53-68, 2021. https://doi.org/10.1007/978-981-15-7804-5_5.

[19] J. Sanz-Rodriguez, J. M. Dodero, S. Sanchez-Alonso, "Metrics-based evaluation of learning object reusability," *Software Qual Journal*, vol. 19, issue 1, pp. 121-140, 2011. https://doi.org/10.1007/s11219-010-9108-5.

[20] A. K. M. Fazal-e Amin, A. Oxley, "Reusability assessment of open source components for software product lines," *International Journal on New Computer Architectures and Their Applications (IJNCAA),* vol. 1, issue 3, pp. 519–533, 2011.

[21] M. Mijač, Z. Stapic, "Reusability metrics of software components: Survey," *Proceedings of the Central European Conference on Information and Intelligent Systems*, pp. 221-231, 2015. DOI: 10.13140/RG.2.1.3611.4642.

[22] A. L. Imoize, D. Idowu, T. Bolaji, "A brief overview of software reuse and metrics in software engineering," *World Science News*, vol. 122, pp. 56–70, 2019.

[23] T. L. Saaty, "How to make a decision: The analytic hierarchy process," *European Journal of Operational Research*, vol. 48, issue 1, pp. 9-26, 1990. https://doi.org/10.1016/0377-2217(90)90057-I.

[24] F. Taibi, "Empirical analysis of the reusability of object-oriented program code in open-source software," *International Journal of Computer, Information, System and Control Engineering*, vol. 8, issue 1, pp. 114–120, 2014.

[25] V. Dimaridou, A. C. Kyprianidis, M. Papamichail, T. Diamantopoulos, A. Symeonidis, "Assessing the user-perceived quality of source code components using static analysis metrics," *Communications in Computer and Information Science (CCIS)*, vol. 868, pp. 3–27, 2018. https://doi.org/10.1007/978-3-319-93641-3_1.

***MANJU DUHAN** has completed PhD degree at Guru Jambheshwar University of Science and Technology, Hisar, India. Her research interests are Neural networks, Machine learning, Neuro-fuzzy, and Software Quality Metrics in Object-Oriented environments.*

***PRADEEP KUMAR BHATIA,** Doctor of Sciences, a Professor, Department of Computer Science and Engineering, Guru Jambheshwar University of Science and Technology, Hisar, India. His research areas are Software Engineering, Computer graphics and Artificial Intelligence.*