Maintainability Prediction for Object-Oriented Software Systems Based on Intelligent Techniques: Literature Review

Anfal A. Fadhil ***Taghreed R. Al_Reffaee***
anfalaaf@uomosul.edu.iq *taghreed_reyad@uomosul.edu.iq*
Department of Software,
College of Computer Science and Mathematics,
University of Mosul, Mosul, Iraq

Received on: 25/08/2020 **Accepted on: 30/09/2020**

ABSTRACT

The maintainability of the software is one of the most substantial aspects when assessing software product quality. It is known as the easiness with which the current software can be changed. In the literature, a great number of models have been suggested to predict and measure maintainability during various stages of the Software Development Life Cycle, to conduct a comparative study of the existing suggested models of the prediction, only few attempts have been done. This study hints at the basics about the manner of how to measure maintainability in the object-oriented (OO) design knowing that the maintainability will be measured differently at every level. Also, we will concentrate on the artificial intelligence technologies of these studies.

Keywords: Maintainability, Object-Oriented, Metrics, Prediction, Measurement

توقع قابلية الصيانة لأنظمة البرمجيات كائنية التوجه المعتمدة على التقنيات الذكائية: دراسة مراجعة

انفال عبد المنعم فاضل **تغريد رياض الرفاعي**
قسم البرمجيات، كلية علوم الحاسوب والرياضيات
جامعة الموصل، الموصل، العراق

تاريخ استلام البحث: ٢٠٢٠/٠٨/٢٥ **تاريخ قبول البحث: ٢٠٢٠/٠٩/٣٠**

الملخص

تعد قابلية صيانة البرنامج أحد أهم الجوانب عند تقييم جودة منتج البرنامج. يتم تعريفه على أنه السهولة التي يمكن بها تغيير البرنامج الحالي. في الدراسات السابقة، تم اقتراح عدد كبير من النماذج للتنبؤ وقياس قابلية الصيانة خلال المراحل المختلفة من دورة حياة تطوير البرامج. ومع ذلك، تم إجراء عدد قليل من المحاولات لإجراء دراسة مقارنة لنماذج التنبؤ المقترحة الحالية. تشير هذه الدراسة إلى الأساسيات التي تتعلق بطريقة كيفية قياس قابلية الصيانة في التصميم الموجه للكائن مع العلم أن قابلية الصيانة سيتم قياسها بشكل مختلف على كل مستوى. بالإضافة إلى ذلك، سنركز على تقنيات الذكاء الاصطناعي لهذه الدراسات

الكلمات المفتاحية: قابلية الصيانة، المقاييس، الكائنات الموجهة، التنبؤ، القياس.

1. Introduction

One of the important stages in the process of Software Development Life Cycle (SDLC) is software maintenance. At this phase, defects introduced are the most serious when compared to those which could be introduced at other software development cycle'

stages [1]. The activities of the maintenance begin from the moment when a system comes into operation and stills for the rest of the product's life.

Thus, according to previous studies, researchers found that for several products this can last for an average of twenty years, unlike the development stage which can last from one to two years. Also, the time spent and effort needed to correct defects in this stage consumes about 40 to 70% of the entire life cycle' cost [2]. According to the standards of the Electrical and Electronics Engineers Institute (IEEE) [3], the maintainability of the software is the easiness with which a component or a system of the software can be changed to performance improvement, right faults, etc. There are different metrics suggested for the software calculation. It can be observed that maintainability of the software is the composite measure of different software features like readability, understandability, etc. From the past studies, it was clear that there is a robust relationship among the metric of the design and the predicting of software maintainability [4].

Modern trends display that most of the systems of the software are utilizing the technique of the Object Oriented to evolve the software products' quality. OO method improves the maintainability of the software system [5].

As the system of OO utilizes a great number of small approaches, a single problem of the maintenance is related to it. The relation among the metrics of Object-Oriented and effort of the software maintenance are nonlinear and complex. Metrics of Object-Oriented is a good platform to access the effort of the maintenance. Usually, the maintainability is calculated as the modifying in effort i.e. lines' number modified each class [6].

A good predictive model of software maintainability allows organizations to manage their resources of maintenance efficiently and also guide decision-making associated with software maintenance which can assist further to minimize the effort of the maintenance and hence they can reduce the total effort and cost of software project[7]. In the literature, several researchers have experimented with various models to predict the maintainability of the software, whether at the level of code or the more abstract level as levels of design and architecture.

Most of these models were suggested in the level of code while a few models were suggested at levels of design and architecture [8]. In this paper, we will be considering the most recent proposed techniques to predict Object-Oriented software maintainability utilizing artificial intelligent techniques.

The rest of this paper is systematic as following: Section 2: gives the most important Object-Oriented software maintainability metrics, Section 3: The goal of the study, Section 4: Literature Review, Section5: Result and Discussion. Finally, Section 6: Conclusion the work and states the possible future work.

2. Maintainability Metrics

In the last years, out of many parameters of the quality i.e., efficiency, functionality, maintainability, portability, reliability, and usability, maintainability has a high priority in performing a large success for any system of the software, The metrics of OO is an effective manner to measure the maintainability of the software, the most famous and used metrics are:

- Chidamber and Kemerer (CK) [9].
- Li and Henry [10] metrics.

Metrics (Size) have been deemed for the maintainability prediction of the software.

2.1 Chidamber and Kemerer (CK) software metrics:

These metrics aim to assess the design of the OO system rather than implementation. This makes them extra appropriate to the OO model, the OO design places great focus on the stage of the software design, The suite of CK metric includes six metrics of the design complexity, these metrics can have utilized as predictors of maintainability, and the (LCOM) metric is disjoint with the maintainability. All (CK) metrics are briefly illustrated (exclude LCOM) as the following [9]:

1- **WMC (Weighted Methods per Class)**: This metric represents sum of weight for the whole method which is declared in the class. This metric measures the class' complication, to maintain and evolve the class, this metric predicts how much effort and time is needed. High value of this metric (WMC) refers to bigger complication and hence lower maintainability [5].
2- **DIT (Depth of Inheritance Tree):** This metric is measured by the ancestor' number of the classes. It is the longest path's length from a certain class to a class that represents the root in the hierarchy of inheritance and so it computes how far down a class is stated in the hierarchy of inheritance. The great value of the DIT metric refers to a bigger complexity of the design and extra fault-proneness. DIT is the maximum deepness in the hierarchy of the inheritance per class.
3- **NOC (Number of Children):** This metric represents the immediate child classes' number which is derived from a basic class. The high value of NOC refers to a bigger level of reuse, the extra effort needed for testing, fault proneness, and extra complexity[5]..
4- **CBO (Coupling Between Objects):** The interdependence of two objects is measured by coupling. For the class, this metric is measured via calculating another class's number to which it has coupled. If one class has methods and these methods utilize instance variables and/or methods of the other class then these two classes have coupled. The high CBO means complex design, complicates the testing of the class and decreases modularity.
5- **RFC (Response for a Class):** This metric represents the number of whole methods that can be implemented directly or indirectly in response to a message to an object of that class or by several methods in the class. (This contains methods available inside the hierarchy of the class). The high value of RFC indicates extra effort needed for testing, fault-proneness, and the bigger complexity of the design.
6- **Lack of Cohesion on Methods (LCOM):** this metric is utilized to measures the lack of cohesion of a class.

   Each value of the metrics above was proportional (inversely) to the maintainability of the system [9].

2.2 The B. Li and Henry (BH)

In addition to the previous metric mentioned above in subsection (2.1) the following metrics have been added to the CK (metric) [10]:

1- **Lack of Cohesion on Methods (LCOM):** this metric is same as that mentioned in the paragraph 2.1 point 6.
2- **Message Passing Coupling (MPC):** It is a metric of coupling measurement which utilized for computation the send statements' number that declared in the class
3- **Data Abstraction Coupling (DAC):** The abstract data types' number defined in a class is measured by DAC[10].

4- Number of Local Methods (NOM): It is a metric of an OO that counting the native methods' number of a class.

5- Size 1: It is a conventional Line of Code (LOC) metrics. This metric is utilized to count the semicolons' number which presents in a class.

6- Size2: number of methods plus number of attributes [10]

Once the data values of the maintainability for the software are specified, a try is made to institute a relation among the metrics and the desired maintainability of the software. The maintainability 'change' is measured as the "number of lines which changed per class". In this method, while developing the relationship, 'change' is deemed as a (dependent variable) and every one of the metric groups as (independent variables) set. Maintainability is thus supposed to be a function of the utilized metrics and may be represented as a function of various CK metrics as following[11]:

$$Maintainability = change = f(WMC, DIT, NOC, RFC, LCOM) \quad (1)$$

The grouping of both (CK and LH metrics) is considered to find the maintainability. LH metrics are (DAC, MPC, and NOM). The relationship can be represented as following [11]:

$$Maintainability = change = f(WMC, DIT, NOC, RFC, LCOM, MPC, DAC, NOM) \quad (2)$$

Also, SIZE metrics are utilized for maintainability prediction along with the CK and LH metrics' combination. The relationship is illustrated as following [11]:

$$Maintainability = change = f(WMC, DIT, NOC, RFC, LCOM, MPC, DAC, NOM, SIZE1, SIZE2) \quad (3)$$

To find the effectiveness of software maintainability prediction, only (SIZE) metric has used as represented in the following relationship [11]:

$$Maintainability = change = f(SIZE1, SIZE2) \quad (4)$$

There is also another metric of maintainability:

- Halstead's metrics.
- Data Access Metric (DAM).
- Mccabe metrics.
- Model Cyclomatic Complexity (CC) metrics.
- Function points.
- CDM and complexity of design metrics but their use in researches very little
- structural metrics

3. The goal of the study

Various studies have been conducted and proposed in the literature for predicting software Maintainability. The focus in this paper is given to supply an overview of development in the Prediction of Maintainability, The aim of this paper is to provide a Literature Review which contains the most essential and new research that participated in the field of Maintainability Prediction for object-oriented software systems based on intelligent techniques. In order to be available to researchers who will work in this field in the future, a state-of-the-art literature review has been carried out to answer the following research questions:

1-What is the intelligent technique's type that achieved better outcomes?

2- What are the Maintainability Prediction's metrics most utilized?

3-What is the data set of Maintainability Prediction most used?

In this Literature Review, a number of papers have been used, these papers were submitted in each stage of SDLC within a period of time and have been published in journals, conferences, and others which contain book chapters, symposiums. The

essential journals were chosen depending on the impact factor while conferences were identified according to the international repute that addresses issues in the software maintenance field. We limited our search to the period from 2010 to 2019 with add first research in this field. In this study, In the Maintainability estimation, most of the research that relies on intelligent techniques has been inclusion, and the research that using other technologies not intelligent has been exclusion.

4. Literature Review

A lot of methods were used for estimating software maintainability since at first provided in the year 1993. Such methods or approaches were based on artificial intelligence (AI), several types of research utilize one technique, while another research hybrid two techniques. Several approaches have provided in the following way:

4.1 Multiple linear Regression (MLR)

In 1993, Wei Li and Sallie Henry [10] studied the metrics of the software in the procedural paradigm as a quantitative means to assess the development process of the software as well as the software product quality. The paper focuses on some OO metrics of the software and these metrics validation with the effort of the maintenance in two commercial systems. Statistical analyses of a prediction model were performed incorporating 10 metrics. In addition, a more compact model was presented with fewer metrics. In 2010, S. Rizvi and R. Khan [12] developed a multivariate linear model, which is utilized to estimate the class diagrams' maintainability depending on their understandability and modifiability. Two extra multivariate models were suggested in order to quantify the modifiability and understandability of the class diagram. These models utilize the OO metrics of the design level. This early maintainability' quantification provides a chance to improve the class diagram' maintainability and as a result the final maintainability of the software. All three models have been validated during suitable statistical measures. In 2012, Alisara and Wanchai [13] suggested a multivariate linear regression for establishing the prediction model of the maintainability depending on extendibility and flexibility which are sub-features of maintainability as a standard for assessing the model of the class diagram' maintainability and evolve the Maintainability Estimation Tool (MET) to estimate a class diagram' maintainability. This tool assists the designer of the software to improve the class diagram's maintainability in the stage of the design and helps to minimize the increasingly high cost of software maintenance stage.

4.2 Vector Machine (SVM) and clustering technique

In 2010 and in the field of software maintainability's prediction, a study achieved by Cong and Jin-An [14] explored the support vector machine's applications and unsupervised learning utilizing metrics of Object-Oriented. The maintenance effort was the dependent variable while five Object-Oriented metrics were represented independent variables that decided the clustering system. This study approved that the technique of SVM and clustering were suitable for building a predictor of maintainability, and the Mean Absolute Relative Error (MARE) was used in the evaluation.

4.3 Neural Networks and fuzzy

In 2010, Arvinder, et al. [15] assessed and compared the application of various soft computing methods: Feed Forward Back propagation Neural Networks (FBNN), Radial Basis Function(RBF) Neural Network, Fuzzy Inference Systems (FIS) and Adaptive

Neuro-Fuzzy Inference Systems (ANFIS) to build Effort prediction models for Software Maintenance. Maintenance effort data for two commercial products of the software was utilized. In this work, the effort of the maintenance was the dependent variable while eight metrics of Object-Oriented represented the independent variables. It was observed that the techniques of soft computing can be utilized for building precise models for maintenance effort prediction of the software and the Adaptive Neuro-Fuzzy Inference System technique provides the most precise model based on the outcomes of MARE, MRE, p-value, and r metrics

In 2014, a study conducted by Momeni and Shiva [16], this study has shown that when compared Adaptive Neuro-Fuzzy Inference System (ANFIS) with other models like Fuzzy Logic, the ANFIS can give more precisely predict maintainability. For this, four metrics (CAE, CDA, CFA, and WOM) were selected. Four parameters were used for measuring the output: CAE is the first input parameter, which displays the aspects' number including advice produced through the operations' implementation in a specified module. It was utilized for measuring the operation's dependence on the advice; so modifying in the advice maybe affect the operation. For this metric, a greater value of a specified module refers to the last is coupled with extra aspects. CDA is the second input parameter that displays the modules' number which is affected by point cuts and introductions in a particular aspect. This measures the modules' number that is impacted by an aspect. The third input parameter is CFA which displays the modules' number that has fields that are called by a particular module. It measures the coupling among modules depends on on-field access. A greater value of CFA indicates tight coupling among the modules which implies complication, decrease in the testability, and also increase in the module being fault-prone. The last parameter is WOM which displays the operations' number (methods or advice) in a module. This is equal to the WMC (weighted operations per class) of the CK metrics. A class is considered to be more complex when it has greater operations numbers the operations' complexity is considered to be equal. Also, with a greater WOM value, additional effort is required for testing a class. A lower value of the WOM is desired for each module. These metrics were utilized for validation, testing, and training[16].

4.4 Fuzzy model

In 2012, Sanjay and Ajay [5] presented a model of fuzzy to quantify the OO software system's maintainability. This model accepts the projects of OO and estimates its maintainability. The result gained by the model of the fuzzy is validated by utilizing the technique of analytical hierarchy processing. The inputs for the suggested model on which maintainability depends are class, complexity, inheritance, coupling, and children's number. These inputs were specified depending on the study and utilizing extensive survey. The rule base was produced by the knowledge of the expert, with 243 rules for assessing an object-oriented software system. The suggested model assessed the maintainability of two object-oriented software systems. The outcomes are validated by the AHP technique. This model will assist the maintainability practitioners, software developers, and researchers to choose the better maintainable object-oriented software system when different alternatives are offered before them. [5]

4.5 Neural Network

In 2012, and in the OO software's maintainability field, a study conducted by Sanjay, et al.[17] suggested Multilayer Perceptron (MLP) neural network because of its adaptability and strength, by utilizing the model of the Multilayer Perceptron (MLP) and

matches the outcomes of this implementation with another model. The results have shown that the effectiveness of this model (MLP) was preferable than others (Ward and GRNN network) models.

In 2013, Vijay and Swati [6] selected the effort of maintenance as (dependent variable) and principal components of OO metrics as (dependent variables). In this study, number of lines per changed per class was a prediction. Two models of neural network are utilized: (ward, and Hopfield neural network). The Artificial neural network owns the features of predicting the quality of software precisely and recognizes faults by mechanisms of effective discovery.

Also in 2013, another study was conducted by Ruchika and Anuradha [18] proposed FFNN (Feed Forward Neural Network) modeling techniques for predicting the OO software maintainability. A set of metrics that Utilized recently by the inducement of two recent metrics that are extra significant and expressive in the applications of data-intensive (Code to Comments Ratio (CCR)), Number of Data Base Connections (NODBC), additional, These metrics taken from CK Metrics suite. For analyzing the suggested metric suite, the values of metrics were calculated on five applications (real-life), this makes utilizing for databases with a great amount. The outcomes displayed that suggested recent metrics set are so active pointer for maintainability in the climate which supplies remote communication to the server to access the huge files of the database. Depending on the outcomes, it can be said that the recent metrics set which suggested in the existing study should be capable to estimate the maintainability of the software extra accurately and precisely for those applications that make heavy utilization of databases through operations.

In the year 2018, a study conducted by Mallaiah, et al.[19] presented a backpropagation neural network (BPNN) to give the solution for OO designing of the software. The Cinema Booking System (CBS) was taken and given as the input document in the formal concept analysis which assisted to provide the relationship of the element in the lattice way. The BPNN analyzed the system and gave best solution for the design. The relationship among the settings was assessed and formulated the design and it was compared with the other design made by the expert manually. The system's performances were assessed and compared with the other methods such as Genetic algorithm and Imperialist Competitive Algorithm with Tabu Search (ICA-TS). The experimental outcome display that the performance of the suggested system is the better than the others. The suggested method attaines the cohesion values up to 0.5682 while current system cohesion value is 0.54072. This result gained by utilizing the BPNN due to it find the best value by analyzing every layer and hidden layer in the network. The relationship is set between the classes based on the solution of BPNN and it increases the cohesion value of the program.[19]

4.6 Group Method of Data Handling (GMDH), Genetic Algorithms (GA) and Probabilistic Neural Network (PNN)

In the year 2012, Ruchika and Anuradha [20] proposed utilizing a few algorithms of machine learning with a goal to predict the maintainability of the software and assess them. The suggested models are Genetic Algorithms (GA), Group Method of Data Handling (GMDH), and Probabilistic Neural Network (PNN) with the Gaussian activation function. The predicting model was built utilizing the techniques of machine learning above said. After performing the experimental study, these three algorithms' execution was compared with prevailing models that were taken from literature such as ANN (Artificial Neural Network) Model, GRNN (General Regression Neural Network)

Model, RT (Regression Tree) Model, Bayesian Model, Stepwise Selection Model, Backward Elimination Model, TreeNets Model, MARS (Multiple Adaptive Regression Splines) Model, ANFIS (Adaptive Neuro-Fuzzy inference System) Model, GN (Generalized Regression) Model, MLR (Multiple Linear Regressions) Model, and SVM (Support Vector Machine) Model. The experimental outcome displayed that GMDH can be utilized as a perfect alternative to the current techniques utilized for maintainability prediction. It helps in maintainability prediction extra precisely and accurately than prevailing models.

4.6.1 Group Method of Data Handling (GMDH)

In the year 2014, a study conducted by Ruchika and Anuradha [21] suggested an experimental study for assessing the efficiency of a novel method called Group Method of Data Handling (GMDH) to predict maintainability over another model. Two web-based software has been utilized for experimental study and developed utilizing C# Language. The code of the source of ancient and recent versions of the two applications was analyzed and collected against alterations made in each class. The modifies were calculated depending on the lines'. The number that added, omitted, or changed in the classes that belonged to the modern version with respect to the classes of the ancient version. Lastly, to create data points, metrics' values were combined with ''change''. In this paper, a trying has been done for examining and estimating the maintainability prediction models' efficiency of the software utilizing real-life projects based on the web. Three models were utilized: General Regression Neural Network (GRNN), GMDH, and Feed Forward 3-Layer Back Propagation Network (FF3LBPN). These models were evolved and the execution of the GMDH was contrasted with the other models (FF3LBPN and GRNN). Developers with the assistance of this experimental analysis can utilize this model to judge software maintainability during coding and designing. In the early stage of software development, the researchers and software practitioners can utilize the GMDH model to expect maintainability. They can, therefore, minimize the maintenance stage and thus save the time

4.7 A hybrid approach of neural network and genetic algorithm (Neuro–Genetic)

In 2015, a study conducted by Lov and Santanu [22] considered the software metrics (Chidamber and Kemerer) as a set to provide necessary data of the input for training the models of artificial intelligence (AI). Two methods of (AI) have been utilized to predict the maintainability, namely, neural network and Neuro-Genetic Algorithm (a hybrid method of NN and GA). The performance was assessed depending on the various performances utilizing Neuro-GA which achieved the best outcome for maintainability prediction when comparing with NN.

Also, in 2015, another study conducted by Lov, et al. [11] proposed the Neuro-Genetic algorithm (hybrid method of NN and GA) for predicting the maintainability on two various case studies: User Interface System (UIMS) and Quality Evaluation System (QUES). The network contains numbers of input neurons, hidden layer, and output layer. The sigmoidal or squashed-S function is utilized for the hidden layer. The Mean absolute error (MAE), Mean Absolute Relative Error (MARE), Root Mean Square Error (RMSE), and Standard Error of the Mean (SEM) were used to evaluate the parameters of the performance for this technique. The outcomes described that the specified subgroup metrics proved an enhanced estimation of the maintainability with higher precision.

Also, in 2015, Lov and Santanu [23] presented metrics of the software to predict the maintainability. The Neuro-GA method was coupled with parallel computing conception

for designing the model of the estimation. In this analysis, the concept involved the use of a changing number of computing nodes. Metrics of the software in combination with the techniques of feature reduction like rough set analysis (RSA) and principal component analysis (PCA) were utilized for analyzing the designed models' effectiveness to predict the maintainability of the software. These techniques have the capability for estimating the output depending on the accessible historical data. For training the network, metrics of the software were taken as input data, and predict the software product' maintainability. From this analysis, maintainability prediction with higher precision is more appropriate.

4.8 Gene Expression Programming (GEP)

In 2016, a study conducted by Sandhya and Anuradha [4] proposed Gene Expression Programming (GEP) for maintainability prediction. The performance of this algorithm was measured with different techniques of the machine learning like Support Vector Machine, Linear regression, Decision Tree Forest, Multilayer Perceptron and Radial basis function neural network. The experimental study was achieved with the assistance of four datasets that open source. Eleven bad smells were specified and were deemed as maintenance's effort. The suggested algorithm (GEP) has better outcomes than machine learning classifiers. So, this suggested algorithm can be utilized as a perfect alternate for maintainability prediction.

4.9 Behaviors Hidden Markov Model

In 2016, a study conducted by Morteza and Hassan [24] enhanced a new behaviors Hidden Markov Model to the OO software maintainability. The suggested model is founded depending on the modern versions of the software quality criterion and it is according to the several recent metrics' measurement. This model has been assessed on a well-known PHP framework. The outcomes displayed that the suggested model was active when compared with the earlier models.

4.10 Artificial neural network with genetic algorithm FGA, particle swarm optimization (PSO) and clonal selection algorithm (CSA)

In the year 2016, Lov and Santanu [25] utilized three techniques of artificial intelligence (AI), like a hybrid method of functional link artificial neural network (FLANN) with genetic algorithm (GA), particle swarm optimization (PSO) and clonal selection algorithm (CSA), i.e., FLANN-Genetic (FGA and AFGA), FLANN-PSO (FPSO and MFPSO), FLANN-CSA (FCSA) were have used for designing a model to predict the maintainability. Also, this study concentrated on the feature reduction techniques' effectiveness like principal component analysis (PCA)and rough set analysis (RSA) and when they were used to predict the maintainability.

4.11 Neuro-Fuzzy with parallel computing

In 2017, Lov and Santanu [26] suggested a Neuro-Fuzzy method(hybrid NN and FL) to develop a model of maintainability utilizing various OO static source code metrics as input. This approach is used on data of the maintainability for two products of commercial software like UIMS and QUES. Rough set analysis (RSA) and principal component analysis (PCA) were utilized to choose an appropriate suite of metrics from the metrics used to get better the model's performance of maintainability prediction. From empirical outcomes, it is noted that the model (Neuro-Fuzzy) can efficiently estimate the OO software systems' maintainability. After executing the concept of parallel computing, it is noted that, when increasing the computing nodes' number, the training time gets

minimized to a significant quantity. It is noted that the chosen subgroup of metrics utilizing the techniques of feature selection (PCA, and RSA) was capable to estimate the maintainability with greater precision.

4.12 Neuro-Particle Swarm Optimization algorithm (NPSO).

In 2018, a study conducted by N. Baskar and C. Chandrasekar [27] proposed the Neuro-Particle Swarm Optimization algorithm (NPSO) to design the model of maintainability prediction. This technique has been used to predict maintainability on the dataset which collected from two various case studies like Quality Evaluation System (QUES) and User Interface System (UIMS). The Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), and Prediction was used to evaluate the parameters of the performance.

4.13 Exponential model with the help of regression technique

In the year 2019, a study conducted by N. Dasari [28] focused on the behavior of the non-linear of the maintainability and its factors, To discover, the public metrics of the OO software Maintainability operators i.e., Understandability, Modifiability, and Analyzability, the exponential model is utilized with the assistance of the regression method.

These public sub-factors were located in the form of the four multivariate models for measuring the OO software maintainability. The suggested models were statistically validated and used on several sample values of the maintainability factors to gain the best outcomes.

Table 1 describes a summary of all previous work mentioned in this review with List the approved metrics, the dataset used, and metrics for evaluating the model used.

TABLE (1). list researches that maintainability Prediction for object-oriented software systems based on intelligent techniques

No	Article	year	Prediction Model	Metric suite	dataset	Prediction Accuracy Measure
1	Li and Henry[10]	1993	Multiple linear regression (MLR)	CK	User Interface Management system (UIMS) and Quality Evaluation Systems(QUES)[10]	---
2	Rizvi and R. Khan [12]	2010	multivariate linear regression	Data Access Metric (DAM)	Matinee et al. [29] Marcela Genero et al. [30]	Mean square =2.768
3	Cong and Jin-An [14]	2010	Support Vector Machine (SVM)and clustering technique	CK and BH	Programs developed by the students in C++	MRE= 0.083, MARE= 0.218
4	Arvinder el al.[15]	2010	(FBNN) (RBF) , (FIS) and (ANFIS)	CK and BH	UIMS and QUES dataset	MARE(FBNN)= 0.265, MRE(FBNN)= 0.09, MARE(RBF)= 0.368 , MRE (RBF)= -0.03 , MARE(FIS)= 0.255, MRE (FIS)= 0.017 , MARE(ANFIS)= 0.308, MRE (ANFIS)= 0.093

5	Sanjay and Ajay [5]	2012	fuzzy model	CK	www.codeproject.com /KB/java/ last accessed on last accessed.	maintainability =25.65
6	Sanjay el al. [17]	2012	Multilayer Perceptron (MLP) Model	CK	UIMS and QUES	R Square= 0.8274 Mean absolute Error = 17.860 Min absolute Error =0.023 Max absolute Error = 67.104 r-Correlation Coefficient =0.946
7	Alisara and Wanchai [13]	2012	multivariate linear regression	CK and DAM	by parsing the XMI file	------
8	Ruchika and Anuradha [20]	2012	GMDH, GA, PNN	CK and BH	QUES	MMRE (GMDH)=0.210, MMRE (GA)= 0.220, MMRE (PNN)= 0.230
9	Vijay and Swati [6]	2013	Hopfield Neural Network	CK and BH	UIMS	R(Correlation Coefficient) = 0.846 P value = <0.001 R-square = 0.685 Mean absolute error= 17.96
10	Ruchika and Anuradha [18]	2013	FFNN	CCR, NODBC and CK	Proprietary systems namely FLM, EASY, SMS, IMS and ABP System	MARE(FLM)=0.3478 , MARE(EASY)=0.3676 , MARE(SMS)=0.4769 , MARE(IMS)=0.4966 , MARE(ABP)=0.3966 ,
11	Ruchika and Anuradha [21]	2014	GMDH	CK and BH	Proprietary systems namely FLM	MMRE= 0.3566
12	Hossein and Shiva [16]	2014	Adaptive Neuro Fuzzy Inference System (ANFIS)	CK	------	root-mean-square error(Mamdani Fuzzy approach)= 0.780 root-mean-square error(Sugeno fuzzy approach)= 0.210
13	Lov and Santanu [22]	2015	Neuro – Genetic	CK	QUES and UIMS	MMRE(QUES)= 0.4180 MMRE(UIMS)= 0.5332
14	Lov el al. [11]	2015	Neuro-Genetic algorithm (hybrid approach of neural network and genetic Algorithm)	CK and BH	QUES and UIMS	MMRE(QUES)= 0.3775 MMRE(UIMS)= 0.3155
15	Lov and Santanu [23]	2015	Neuro-GA with parallel computing	CK,,and BH	QUES and UIMS	MRE(QUES)= 0.3775, MMRE(UIMS)=0.3155
16	Sandhya and Anuradha [4]	2016	Gene Expression Programming	CK , BH and CC	jdeodorant (https://marketplace.eclipse.org/content/jdeodorant) and	RMSE Mean Rank = 1.10 MAE Mean Rank= 1.25

					robusta (https://marketplace.eclipse.org/content/robusta-eclipse-plugin)	
17	Morteza and Hassan [24]	2016	behaviors Hidden Markov Model	CK and CC	PHP frameworks.	Ratio(Codeigniter)= 1.222 ratio (cakephp)= 1.036 ratio (Yii frameworks)= 1.436
18	Lov and Santanu [25]	2016	artificial neural network with genetic algorithm FGA, particle swarm optimization (PSO) and clonal selection algorithm (CSA)	CK	QUES and UIMS	MMRE(QUES(FGA))= 0.3889 MMRE(UIMS(FGA))= 0.2881 , MMRE(QUES(PSO))= 0.3650 MMRE(UIMS(PSO))= 0.3238 , MMRE(QUES(CSA))= 0.4469 MMRE(UIMS(CSA))= 0.2843
19	Lov and Santanu [26]	2017	Neuro-Fuzzy with parallel computing	CK and BH,	QUES and UIMS	MMRE(QUES)= 0.3377 MMRE(UIMS)= 0.3366
20	N. Baskar and C. Chandrasekar [27]	2018	Neuro-Particle Swarm Optimization algorithm (NPSO).	CK and BH	QUES and UIMS	MMRE(QUES)= 0.2931 MMRE(UIMS)= 0.2936
21	Mallaiah el al. [19]	2018	back propagation neural network (BPNN)	CK	The Cinema Booking System (CBS)	cohesion value=0.5522
22	N. Dasari [28]	2019	exponential model with the help of regression technique.	CK and structural metrics	Marcela Genero et al.[30]	R=0.967 Std. Error= 0.134

5. Result and Discussion

After the intelligent techniques have presented that were used to Maintainability Prediction for object-oriented software systems, what metrics each technology used to predict the Maintainability and on which data set were applied, they were found the intelligent techniques GMDH[20], GA[20], PNN[20], Neuro – Genetic[22], a hybrid approach of neural network and Genetic Algorithm[11], Neuro-GA with parallel computing[23], FGA[25], PSO[25], CSA[25] and Neuro-Fuzzy with parallel computing[26] and NPSO[27] have used CK & HB metrics on QUES and UIMS dataset, The results were comparable, depending on the MMRE metric, Where it was ranged between (0.4469-0.210) for a QUES dataset and GMDH the best the intelligent techniques that achieve result Which equal (0.210). but regarding the UIMS dataset, the

result was It ranged between (0.5332-0.2843) and CSA the best intelligent techniques that achieve result Which equal (0.2843), and CK & HB metrics and the QUES and UIMS dataset Most used of Maintainability Prediction as shown in table 1. The rest of the intelligent techniques mentioned in the table relied on different measures and different data sets, so the best technique was not determined.

6. Conclusion

In this review, the most essential techniques have used in the literature and applied for prediction the maintainability of the OO software and for building prediction models were have presented. Several of these studies utilized one technique, while others hybridized the techniques to get an effective prediction model. Based on the results, this Literature Review showed that the most used metrics are Chidamber and Kemerer (CK) and B. Li and Henry (BH). And most datasets were are QUES (Quality Evaluation Systems) and UIMS (User Interface Management system).and the best the intelligent techniques on the QUES dataset was GMDH that achieve MMRE equal (0.210) and the best the intelligent techniques on the UIMS dataset was CSA, which achieve MMRE equal (0.2843).

For future work, the researchers recommend explore new metrics for prediction the maintainability, and explore modern techniques and models for software maintainability prediction.

Acknowledgement: First of all, we would like to thank Allah, the Lord of the creation for His bestowed and determination upon me to accomplish this work. Secondly, the authors are very grateful to the University of Mosul/ College of Computer Sciences and Mathematics for their provided facilities, which helped to improve the quality of this work.

REFERENCES

[1] Mens, Tom, et al. "Evolving Software Systems", Springer (2014).

[2] Kumar, Rajendra, Dhanda, Namrara, "Maintainability Measurement Model for Object-Oriented Design." In International Journal of Advanced Research in Computer and Communication Engineering, (2015):68-71.

[3] IEEE Std. "Standard Glossary of Software Engineering Terminology." IEEE Computer Society Press, Los Alamitos, CA, 1993

[4] Tarwani, Sandhya and Chugv, Anuradha, "Predicting Maintainability of Open Source Software using Gene Expression Programming and Bad Smells." 5th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO), India, 2016

[5] Dubey, Sanjay and Rana, Ajay. "A Fuzzy Approach for Evaluation of Maintainability of Object Oriented Software System", International Journal of Computer Applications 49, no. 21 (2012).

[6] Dhaka, Vijay and Agrawal, Swati. "Optimization of Object-Oriented Metrics Using Hopfield Neural Network." International Journal of Soft Computing and Engineering 3(2013).

[7] Liang, Ping. " A Quantitative Approach to Software Maintainability Prediction." International Forum on Information Technology and Applications (2010).

[8] Zighed, Narimane and Nora Bounour. "Comparative Analysis of Object-Oriented Software Maintainability Prediction Models." foundations of computing and decision sciences 43, no. 4(2018) No

[9] Chidamber, S. R. and Kemerer, C.F., "A Metrics Suite for Object Oriented Design." IEEE Transactions on Software Engineering 20, no. 6(1994):476-493.

[10] Li, Wei and S. Henry. "Object-oriented metrics which predict maintainability." Journal of Systems and Software 23 no.2(1993):111-122.

[11] Kumar, Lov, et al. "Validating the Effectiveness of Object-Oriented Metrics for Predicting Maintainability." Procedia Computer Science 57 (2015): 798 – 806.

[12] S. W. A. Rizvi and R. A. Khan. "Maintainability Estimation Model for Object-Oriented Software in Design Phase (MEMOOD)." Journal of computing 2(2010).

[13] Hincheeranan, Alisara and Wanchai Rivepiboon. "A Maintainability Estimation Model and Tool." International Journal of Computer and Communication Engineering 1,no. 2(2012).

[14] Jin, Cong and Jin-An Liu. "Applications of Support Vector Mathine and Unsupervised Learning for Predicting Maintainability Using Object-Oriented Metrics." Second International Conference on Multimedia and Information Technology (2010).

[15] Kaur, Arvinder, et al. "Soft computing approaches for prediction of software maintenance effort." International Journal of Computer Applications 1, no.16(2010).

[16] Hossein, M. and Shiva, Z. "Aspect-Oriented Software Maintainability Assessment Using Adaptive Neuro Fuzzy Inference System (ANFIS)." Journal of mathematics and computer Science12, (2014): 243 – 252

[17] Sanjay, K.D., et al. "Maintainability Prediction of Object-Oriented Software System by Multilayer Perceptron Model." ACM SIGSOFT Software Engineering Notes 37, no. 5(2012)

[18] Ruchika, M., and Anuradha Ch. "An empirical study to redefine the relationship between software design metrics and maintainability in high data intensive applications." Proceedings of the World Congress on Engineering and Computer Science 1(2013): 61-66

[19] Mallaiah, V., et al. "Automization of Object Oriented Design Using Back Propagation Neural Network for Better Maintainability of Software." International Journal of Intelligent Engineering and Systems 11, no.4(2018)

[20] Ruchika, M. and Anuradha, Ch. "Software Maintainability Prediction using Machine Learning Algorithms." Software Engineering: An International Journal 2, no.2(2012):9-36

[21] Ruchika, M. and Anuradha, Ch. "Application of Group Method of Data Handling model for software maintainability prediction using object oriented systems." Int J Syst Assur Eng Manag 5(2014):165–173

[22] Lov, K. and Santanu, Ku. "Neuro – Genetic Approach for Predicting Maintainability Using Chidamber and Kemerer Software Metrics Suite." Advances in Intelligent Systems and Computing 2015

[23] Lov, K. and Santanu, Ku. "Predicting Object-Oriented Software Maintainability using Hybrid Neural Network with Parallel Computing Concept.", Proceedings of the 8th India Software Engineering Conference(2015)

[24] Morteza, A. and Hassan, R. "A Model for Object-Oriented Software Maintainability Measurement." International Journal of Intelligent Systems Technologies and Applications 8, no.1(2016):60-66

[25] Lov, K. and Santanu, Ku. "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software." The Journal of Systems and Software 121(2016):1–21

[26] Lov, K. and Santanu, Ku. "Software maintainability prediction using hybrid neural network and fuzzy logic approach with parallel computing concept." International Journal of System Assurance Engineering and Management 8(2017):1487–1502

[27] N. Baskar and C. Chandrasekar, "An Evolving Neuro-PSO-based Software Maintainability Prediction." International Journal of Computer Applications 179,no.18(2018)

[28] N. Dasari, "Common Metrics For Object – Oriented Software Maintainability Model." International Journal of scientific and technology research 8(2019)

[29] Matinee, K., et al. "A Methodology for Constructing Maintainability Model of Object-Oriented Design", Proc. 4th International Conference on Quality Software, IEEE Computer Society(2004):206-213

[30] Marcela, Genero, et al. "A Controlled Experiment for Corroborating the Usefulness of Class Diagram Metrics at the Early Phases of Object-Oriented Developments." Proc. of the ADIS 2001, Workshop on Decision Support in Software Engineering 84(2001).