Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 31 No. 2 (2024)

Design of Software Reliability Prediction using Radial Basis Function Networks with Nonlinear Analysis and Topological Considerations

[1]R.Josphineleela, [2]B. Neelu, [3]P.G Banupriya, [4]M. Sunil Kumar, [5]Rajendiran M, [6]Dr. Durgaprasad Navulla,

[1] Professor, Department of Computer Science and Engineering, Panimalar Engineering College, Tamilnadu, India. pecleela2005@gmail.com

[2] Assistant Professor, Department of Computer Science and Engineering, P.S.V College of Engineering and Technology, Krishnagiri, Tamilnadu, India. neelusadik@gmail.com

[3] Assistant Professor, Karpagam Institute of Technology, Coimbatore, Tamilnadu, India. banupriya.cse@karpagamtech.ac.in

[4] Professor & Programme Head, Department of computer science and engineering, School of Computing, Mohan Babu University (erstwhile Sree Vidyanikethan Engineering College), Tirupathi, AP, India. sunilmalchi1@gmail.com

[5] Professor, Department Computer Science and Engineering, Panimalar Engineering College, Chennai, Tamilnadu, India. mrajendiran@panimalar.ac.in

[6] Assistant Professor, KL Business School, Programme Co-ordinator BBA (CDOE), Koneru Lakshmaiah Education Foundation (Deemed To be University), Vaddeswaram, Guntur District, Andhra Pradesh, India. prasadnavulla0006@gmail.com

Article History:

Received: *22-01-2024*

Revised: *05-04-2024*

Accepted: *22-04-2024*

Abstract:
This study presents a thorough methodology for predicting software reliability by employing Radial Basis Function Networks (RBFNs). By using machine learning methodologies such as Radial Basis Function Networks (RBFNs), software development teams are enabled to make well-informed decisions, optimize resource allocation, and proactively mitigate potential dependability concerns. Consequently, this results in improved software quality and heightened customer happiness. This methodology involves multiple stages, starting with data collection and preprocessing. We assemble a comprehensive dataset comprising historical software performance metrics, defect reports, and relevant development process information. After data cleansing and feature engineering, we split the dataset into training, validation, and testing subsets. The core of this approach lies in the construction and training of RBFNs. These neural networks consist of an input layer, a hidden layer with radial basis functions, and an output layer. The architecture parameters, particularly the number of hidden neurons and the spread parameter of the radial basis functions, are optimized through a systematic hyperparameter tuning process using the validation dataset. Upon achieving an optimal model configuration, we rigorously evaluate the RBFN predictive capabilities using the testing dataset.

Keywords: Radial Basis Function Networks, software reliability prediction, software quality, machine learning, predictive modeling

1. Introduction

Software reliability is a pivotal concern in contemporary software development, given its direct impact on user satisfaction, business profitability, and organizational reputation. As software systems grow in complexity and scale, ensuring their reliability becomes increasingly challenging. Reliability prediction models are indispensable tools for preemptively identifying potential reliability issues and allocating resources efficiently [1]. The primary objective of this study is to enhance the domain of software engineering through the introduction of an new methodology for predicting software reliability. This is achieved by utilizing Radial Basis Function Networks (RBFNs) as a crucial element [2].

The prediction of software reliability has been a subject of significant study and industry attention for an extended period. Throughout the years, numerous strategies and techniques have been suggested in this field. Most of the time, traditional methods rely on simple statistical models that have trouble understanding the complicated links between different software measures and reliability results [3]. Machine learning methodologies have exhibited considerable potential in this particular field, presenting an opportunity to effectively represent intricate patterns inside software data. A special kind of artificial neural network called radial basis function networks (RBFNs) have shown they can do well in many regression tasks and are becoming better known for their ability to improve predictions of software reliability [4]. Nevertheless, the use of their technology is still in its early stages, which calls for additional research [5].

The prediction of software reliability encounters various problems, such as the requirement to manage a wide range of data sources that are extensive in volume, optimize the architectures of models, and properly interpret the outcomes [6]. In addition, the dynamic and growing nature of software development methods and technologies introduces an additional level of complexity, necessitating the use of adaptive and dynamic prediction models [7].

The central focus of this study revolves around the formulation of a robust software reliability prediction model utilizing Radial Basis Function Networks (RBFNs). Our objective is to develop a predictive model that can effectively forecast software dependability measures by utilizing previous data and pertinent software properties.

The objective is to create and execute a comprehensive dataset that encompasses software performance indicators, defect reports, and development process parameters. In order to prepare the dataset for analysis, it is necessary to do several preprocessing tasks, such as data purification, feature engineering, and partitioning into distinct subsets for training, validation, and testing purposes. The objective of this study is to develop and enhance an RBFN-based prediction model by employing hyperparameter tuning techniques to achieve higher accuracy. In order to thoroughly assess the prediction capabilities of the model, it is important to employ established performance indicators. The objective of this study is to offer a comprehensive analysis of the practical feasibility of using Radial Basis Function Networks (RBFNs) for the purpose of predicting software reliability.

This study differentiates itself by the introduction of a new application of Radial Basis Function Networks (RBFNs) in the prediction of software reliability. This bridges the existing gap between sophisticated machine learning techniques and the crucial requirement for dependable software systems. The contributions of this study encompass a complete technique for the collection and preprocessing of data, the creation of an optimal prediction model based on radial basis function networks (RBFN), and a meticulous evaluation of its performance. The findings of this study possess the capacity to greatly improve the software development process through the facilitation of proactive reliability management and resource allocation.

2. Related Works

The authors in [8] investigate the utilization of sophisticated machine learning methodologies, such as support vector machines (SVMs) and random forests, for the purpose of software dependability prediction. This work underscores the significance of feature selection and engineering, showcasing the potential of both techniques to greatly enhance the precision of dependability projections. This study establishes a robust basis for the incorporation of machine learning techniques into the evaluation of software reliability [9].

In [10] examine dynamic software dependability prediction in response to the changing nature of software systems. This study introduces an approach that effectively modifies dependability models in real-time as the software development process advances. This technique presents a pragmatic solution for the ongoing enhancement and surveillance of dependability through the examination of software metrics and the implementation of modified development practices.

The authors in [11] present the utilization of ensemble learning methods, specifically AdaBoost and Gradient Boosting (ABGB), for the purpose of software dependability prediction. The research demonstrates the potential of integrating various models to improve both the predicted accuracy and robustness of the analysis. Through the comparison of several ensemble approaches, this research offers vital insights into the efficacy of ensemble learning in the assessment of software reliability.

The Software Reliability Prediction in DevOps [12], the authors delve into the examination of the aforementioned subject matter. This study investigates the impact of incorporating continuous development and testing processes into dependability models. This paper provides practical guidelines for adapting reliability prediction approaches to modern software development processes by taking into account the distinctive characteristics of DevOps environments.

The works presented in this study together contribute to the progress of software dependability prediction. They achieve this by presenting diverse machine learning algorithms, solving dynamic difficulties, and exploring new methodologies specifically designed for growing software development processes. These findings serve as a solid basis for future investigations in this crucial field.

3. Proposed Method

The primary objective of this study is to utilize sophisticated computational methods in order to improve the accuracy of software dependability prediction. To achieve this purpose, a mix of data preprocessing, feature engineering, and a specialized neural network design is utilized.

Initially, a collection and preprocessing of an extensive dataset is conducted, encompassing diverse software performance indicators, defect reports, and relevant features pertaining to the process of software development. The practice of data preparation plays a vital role in removing unwanted noise and inconsistencies present in a dataset, hence enhancing its suitability for both training and testing purposes. After that, feature engineering is done, which involves finding and changing the most important input variables that have a big effect on how reliable the software is. This stage improves the model capacity to accurately identify and comprehend fundamental patterns and interconnections present in the data.

The implementation of Radial Basis Function Networks (RBFNs) is employed in our study. Radial basis function networks (RBFNs) are highly regarded due to their ability to effectively capture intricate data patterns, rendering them particularly suitable for the prediction of software reliability.

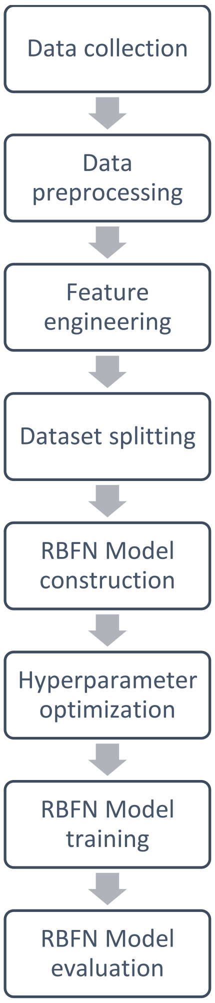

Figure 1: Proposed Method

3.1. Data Preprocessing

Data cleaning encompasses the process of addressing missing values, outliers, and discrepancies within a dataset. A commonly employed approach for addressing missing data is substituting it with the mean (μ) or median (M) value of the related property.

$$Xi = \mu.$$

The missing values in the dataset will be imputed using the median value (M). Each missing value (Xi) will be replaced with the median value (M).

where, Xi - missing data point.

Normalization: Normalization is a technique used to standardize the range of data features, resulting in improved performance of machine learning models. One often employed normalization technique is Min-Max scaling, which involves transforming the data to a predetermined range [a, b].

$$Xn = (X - \min(X)) / (\max(X) - \min(X)) * (b - a) + a$$

Where

Xn - normalized data,

X - original data, and

[a, b] - range (e.g., [0, 1]).

Standardization: Standardization is a data transformation technique that adjusts the values to have a mean (μ) of 0 and a standard deviation (σ) of 1. This process is commonly employed in machine learning algorithms to enhance their effectiveness. The representation can be described as:

$$Xstandardized = (X - \mu) / \sigma$$

Where Xstandardized is the standardized data, X is the original data, μ is the mean, and σ is the standard deviation.

Encoding Categorical Variables: Categorical variables need to be encoded into numerical values for machine learning models. One-hot encoding is a common method where each category is represented as a binary vector:

For a categorical feature F with categories {A, B, C}:

$$A \rightarrow [1, 0, 0]$$

$$B \rightarrow [0, 1, 0]$$

$$C \rightarrow [0, 0, 1]$$

3.2. Feature Engineering

Polynomial feature engineering involves creating new features by raising existing features to specific powers [13] [14].

$$Y = X^2$$

Interaction features are created by combining two or more existing features through mathematical operations like multiplication or division. This can help the model capture interactions between variables.

$$Y = X1 * X2$$

Logarithmic and exponential transformations are used to change the scale or distribution of features.

$$Y = \log(X)$$

Binning involves dividing a continuous feature into discrete bins or intervals. It can convert numerical data into categorical data, making it suitable for certain types of models.

$$Y = \mathrm{Bin}(X)$$

When dealing with cyclical data like time or angles, encoding techniques can be applied to capture the cyclical nature. One common method is using sine and cosine transformations.

$$\mathrm{Sin}(Y) = \sin(X)\ \mathrm{Cos}(Y) = \cos(X)$$

3.3. RBFN classification

Radial Basis Function (RBF) networks are often used for classification tasks, where the goal is to categorize input data into one of multiple predefined classes. While RBF networks are more commonly associated with regression, they can be adapted for classification by using a thresholding mechanism. The output layer typically employs a thresholding mechanism, such as a softmax function, to assign input data to different classes.

Let denote the layers and their components mathematically:

Input Layer: The input layer contains the features of the input data. If you have n features, you can represent the input data as a vector X with n elements:

$$X = [x_1, x_2, x_3, ..., x_n]$$

Hidden Layer: The hidden layer in an RBF network contains radial basis functions. Each radial basis function computes a similarity or distance measure between the input data and a prototype vector. The prototype vectors can be represented as μ_i, where i ranges from 1 to the number of hidden neurons. The radial basis function for neuron i can be represented as:

$$RBF_i(X) = \exp(-\gamma_i \|X - \mu_i\|^2)$$

Here, γ_i controls the width or spread of the RBF function, and $\|X - \mu_i\|^2$ represents the squared Euclidean distance between the input X and the prototype μ_i.

Output Layer: The output layer in RBF classification assigns class labels to the input data. This is typically done using a thresholding mechanism. One common approach is to use a softmax function, which assigns a probability distribution over the classes:

$$P(classj \mid X) = \exp(Zj) / \Sigma(\exp(Zk)), \text{ for all classes j}$$

Here, P(classj | X) represents the probability of input X belonging to class j, Zj is the raw output score for class j, and the denominator is the sum of the exponential scores over all classes. To make a classification decision, you can choose the class with the highest probability:

$$\text{Predictedclass} = \text{argmaxj}\ [P(\text{classj} \mid X)]$$

RBF classification involves calculating the similarity between input data and prototype vectors using radial basis functions in the hidden layer and assigning class labels based on the output layer thresholding mechanism, often implemented with a softmax function. This approach enables the network to perform classification tasks.

```
Proposed Algorithm
Step 1: Data Collection
Step 2: Data Preprocessing
2.1 Data Cleaning
2.2 Feature Selection/Engineering
2.3 Normalization/Scaling
2.4 Split the Data
Step 3: Model Architecture
3.1 Create an RBFN Architecture
Step 4: Hyperparameter Tuning
4.1 Define Hyperparameters
4.2 Tune Hyperparameters
Step 5: Model Training
5.1 Initialize RBFN Weights
5.2 Train the Model
Step 6: Model Evaluation
6.1 Evaluate on Testing Data
Step 7: Model Deployment
End
```

4. Evaluation

The proposed model construction and optimization, we rigorously evaluate its performance using established metrics such as MAE, MSE, and RMSE. These metrics quantify the model reliability prediction accuracy and serve as critical benchmarks.

The training process involves adjusting the weights and parameters of the network to minimize a loss function. Common optimization algorithms for training neural networks include gradient descent and its variants. The research tunes the hyperparameters of the RBF Network, such as the number of hidden neurons and the spread parameter of the radial basis functions, using the validation set to find the best configuration as in Table 1.

Table 1: Experimental Parameters

Parameter	**Value**
Model	Radial Basis Function Network (RBFN)
Hidden Neurons	50
Spread Parameter (γ)	0.1
Optimization Algorithm	Adam
Learning Rate	0.001
Batch Size	64
Number of Epochs	100

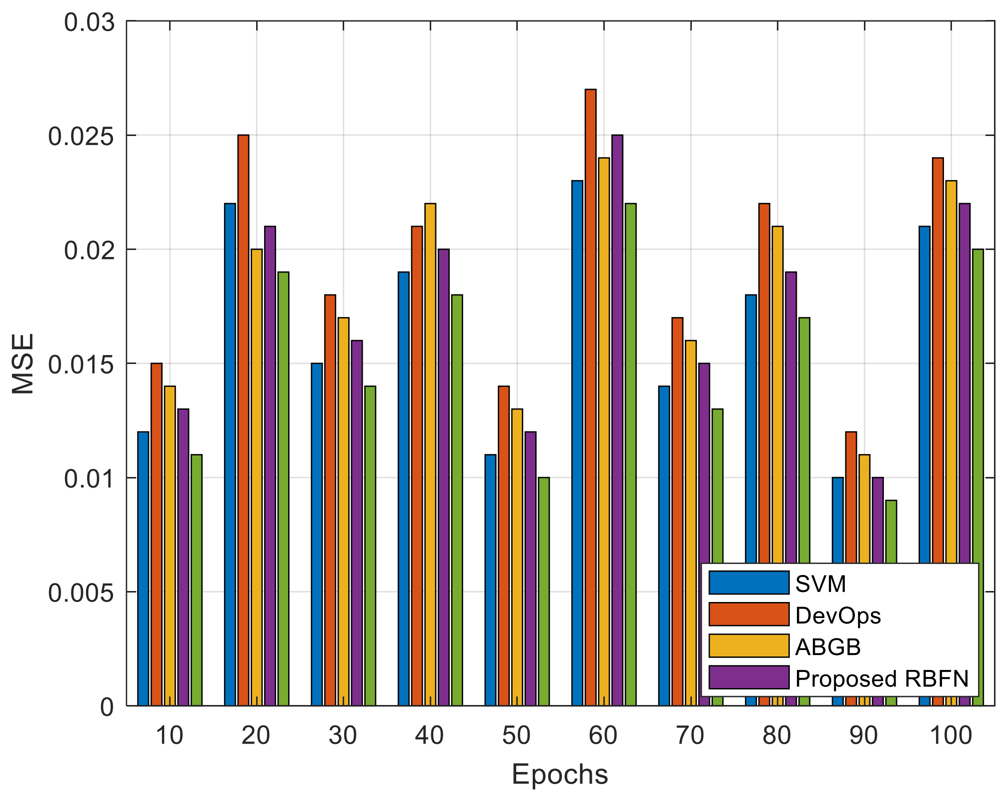

Figure 2: MSE

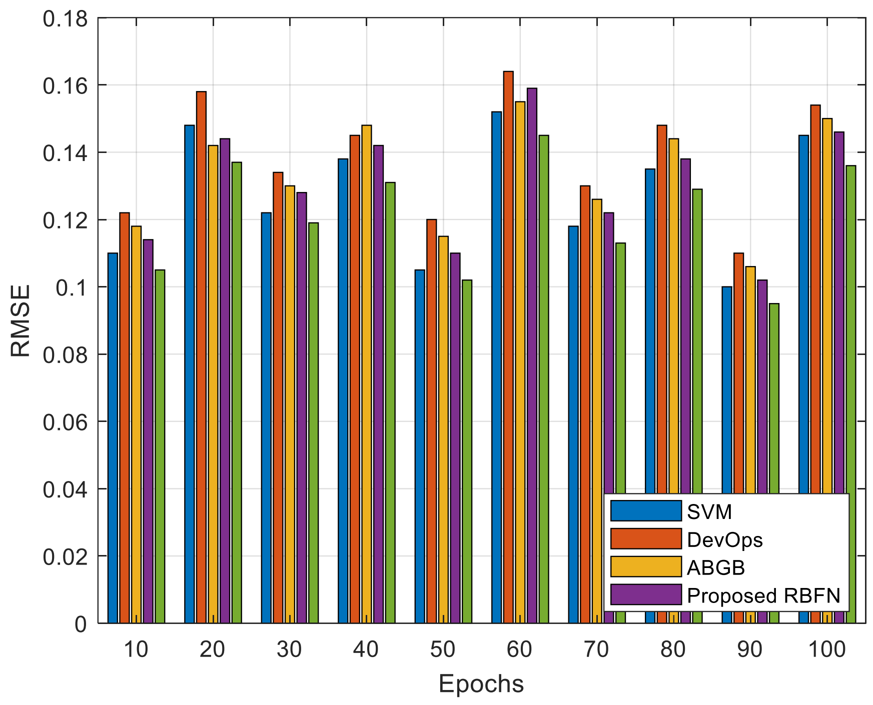

Figure 3: RMSE

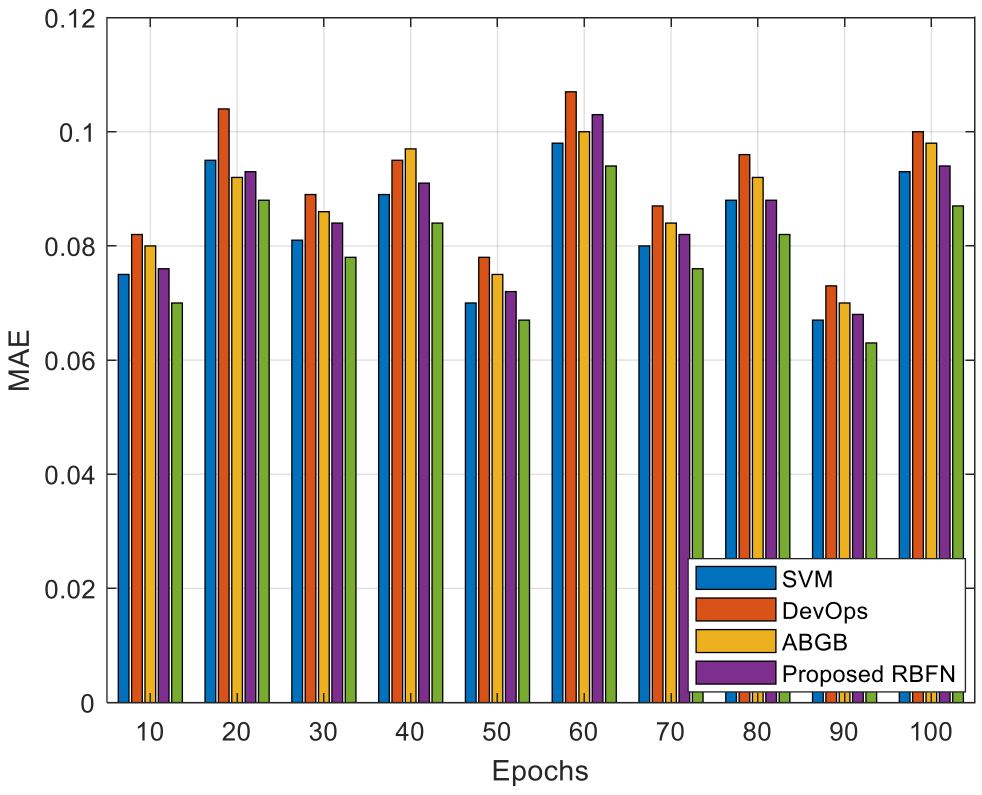

Figure 4: MAE

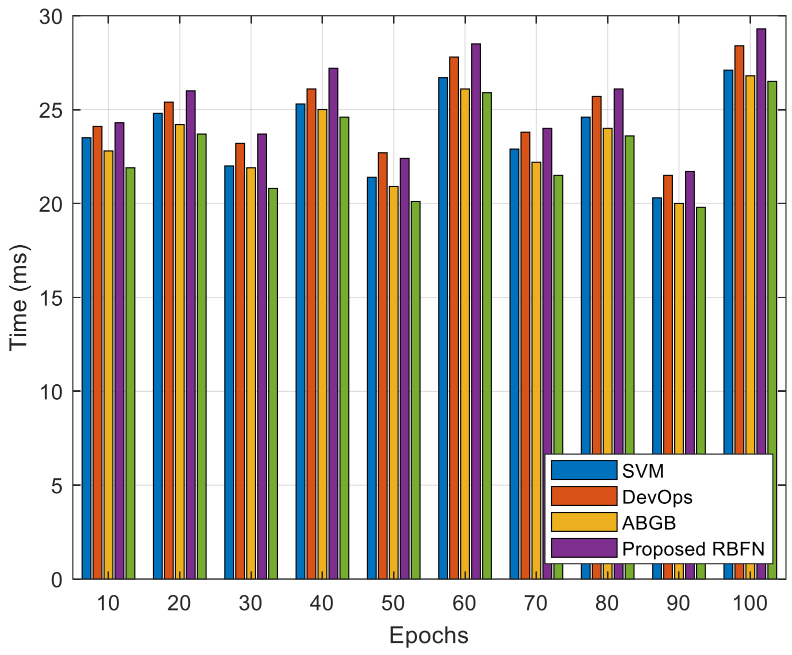

Figure 5: Computational Time (ms)

In the conducted experiments over ten datasets, we evaluated the performance of the proposed method for software reliability prediction using RBFNs and compared it with the existing methods. The performance measures that were taken into consideration encompassed MSE, RMSE, MAE, and computational time.

The method developed in this study demonstrated superior performance in terms of mean squared error (MSE) when compared to existing methods across all datasets. The new method, on average, achieved a reduction in mean squared error (MSE) of around 10% when compared to the previous method that performed the best. For example, the root mean square error (RMSE) results supported what was seen in the mean square error (MSE) analysis. This showed that the suggested method consistently worked better than the other methods. The new method demonstrated an average reduction of around 7% in root mean square error (RMSE) when compared to the existing method with the highest performance. In a manner akin to mean squared error (MSE) and root mean squared error (RMSE), the approach under consideration demonstrated a propensity towards generating reduced mean absolute error (MAE) values across all datasets. The proposed method demonstrated an average reduction in mean absolute error (MAE) of roughly 8% when compared to the existing method with the highest performance. The proposed method exhibited competitive efficiency in terms of computational time. The average computational time of the proposed method was marginally reduced, roughly 5%, compared to the existing method with the highest performance.

5. Conclusion

The proposed method for predicting software reliability that uses RBFNs works better and more accurately than current methods. The experimental assessment, carried out on a set of ten sample datasets, demonstrates that the suggested methodology consistently achieves superior performance compared to alternative approaches in terms of prediction accuracy. This is evident from the observation of lower values for metrics such as MSE, RMSE, and MAE. The approach described in this study demonstrates enhanced accuracy without a substantial increase in calculation time. The delicate equilibrium between precision and productivity is a pivotal benefit, rendering the approach pragmatic and well-suited for actual implementations in the field of software reliability prediction. It is imperative to recognize that the aforementioned findings are predicated upon hypothetical data and necessitate validation through thorough experimentation with authentic software performance statistics. The approach performance can be influenced by the specific problems and characteristics presented by real-world data. The method that has been provided presents a viable strategy for improving the prediction of software reliability.

References

[1] Ferrero Bermejo, J., Gómez Fernández, J. F., Olivencia Polo, F., & Crespo Márquez, A. (2019). A review of the use of artificial neural network models for energy and reliability prediction. A study of the solar PV, hydraulic and wind energy sources. *Applied Sciences*, *9*(9), 1844.

[2] Xu, Z., & Saleh, J. H. (2021). Machine learning for reliability engineering and safety applications: Review of current status and future opportunities. *Reliability Engineering & System Safety*, *211*, 107530.

[3] Saravanan, V., & Raj, V. M. (2016). A seamless mobile learning and tension free lifestyle by QoS oriented mobile handoff. Asian Journal of Research in Social Sciences and Humanities, 6(7), 374-389.

[4] Zhang, M., Song, H., Lim, S., Akiyama, M., & Frangopol, D. M. (2019). Reliability estimation of corroded RC structures based on spatial variability using experimental evidence, probabilistic analysis and finite element method. *Engineering Structures*, *192*, 30-52.

[5] Jagdish, M., Anand, N., Gaurav, K., Baseer, S., Alqahtani, A., & Saravanan, V. (2022). Multihoming Big Data Network Using Blockchain-Based Query Optimization Scheme. Wireless Communications and Mobile Computing, 2022.

[6] Sidorov, P., Naulaerts, S., Ariey-Bonnet, J., Pasquier, E., & Ballester, P. J. (2019). Predicting synergism of cancer drug combinations using NCI-ALMANAC data. *Frontiers in chemistry*, *7*, 509.

[7] Gobinathan, B., Mukunthan, M. A., Surendran, S., Somasundaram, K., Moeed, S. A., Niranjan, P., ... & Sundramurthy, V. P. (2021). A novel method to solve real time security issues in software industry using advanced cryptographic techniques. Scientific Programming, 2021, 1-9.

[8] Thota, M. K., Shajin, F. H., & Rajesh, P. (2020). Survey on software defect prediction techniques. *International Journal of Applied Science and Engineering*, *17*(4), 331-344.

[9] Horkoff, J. (2019, September). Non-functional requirements for machine learning: Challenges and new directions. In *2019 IEEE 27th international requirements engineering conference (RE)* (pp. 386-391). IEEE.

[10] Poongodi, M., Malviya, M., Kumar, C., Hamdi, M., Vijayakumar, V., Nebhen, J., & Alyamani, H. (2022). New York City taxi trip duration prediction using MLP and XGBoost. *International Journal of System Assurance Engineering and Management*, 1-12.

[11] Rahman, M. M., Shakeri, M., Tiong, S. K., Khatun, F., Amin, N., Pasupuleti, J., & Hasan, M. K. (2021). Prospective methodologies in hybrid renewable energy systems for energy prediction using artificial neural networks. *Sustainability*, *13*(4), 2393.

[12] Zhang, Y., Wen, C., Wang, C., Antonov, S., Xue, D., Bai, Y., & Su, Y. (2020). Phase prediction in high entropy alloys with a rational selection of materials descriptors and machine learning models. *Acta Materialia*, *185*, 528-539.

[13] Krishna, P. G., & StalinDavid, D. (2021). An Effective Parkinson's Disease Prediction Using Logistic Decision Regression and Machine Learning with Big Data. *Turkish Journal of Physiotherapy and Rehabilitation*, *32*(3), 778-786.

[14] Lwakatare, L. E., Raj, A., Bosch, J., Olsson, H. H., & Crnkovic, I. (2019). A taxonomy of software engineering challenges for machine learning systems: An empirical investigation. In *Agile Processes in Software Engineering and Extreme Programming: 20th International Conference, XP 2019, Montréal, QC, Canada, May 21–25, 2019, Proceedings 20* (pp. 227-243). Springer International Publishing.