International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 2, pp 8-18, February-2024

https://doi.org/10.47001/IRJIET/2024.802002

Software Reliability Prediction Using Deep Learning and Feature Selection Algorithms

[1]Shahbaa I. Khaleel, [2]Lumia Faiz Salih

[1,2]Department of Software Engineering, College of Computer Science and Mathematics, Mosul University, Mosul, Iraq

Authors E-mail: [1]shahbaaibrkh@uomosul.edu.iq, [2]lumia.21csp2@student.uomosul.edu.iq

***Abstract* - Software reliability is crucial in preventing user issues, financial losses, and reputational damage to companies. Developing accurate models for estimating reliability is imperative. Deep learning, a branch of artificial intelligence, uses neural networks to understand and analyze data, playing a vital role in predicting errors and improving software quality. In this research, Neural Networks (NN), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM) algorithms, along with statistical methods like Chi-square and Regression Coefficient, and intelligent algorithms such as Particle Swarm Optimization (PSO) and Whale Optimization Algorithm (WOA), were employed for feature selection. The results highlighted the superiority of PSO and WOA over traditional methods, with LSTM outperforming other algorithms. Evaluation metrics, including Accuracy, Precision, Recall, and F1-Score, indicated that WOA with LSTM achieved 100% accuracy across datasets. For DS1, accuracy was 97% for all networks, reaching 100% with WOA. DS2 showed accuracy improvement from 80% to 82% with statistical methods and up to 100% with WOA. DS3 demonstrated 99% accuracy with statistical methods and PSO, reaching 100% with WOA. DS4 maintained 99% accuracy with all methods. DS5 exhibited accuracy ranging from 82% to 84%, reaching 100% with WOA. DS6 had accuracy between 78% and 77%, reaching 100% with WOA. This underscores the effectiveness of deep learning, especially with PSO and WOA, in enhancing software reliability.**

Keywords: Software Reliability, Deep Learning, Neural Networks, Feature Selection.

I. INTRODUCTION

In recent years, there has been significant progress in software engineering, expected to continue with the growing size and complexity of applications [1]. The increasing use of applications in various fields, coupled with tight project deadlines, puts pressure on mobile application development teams to rapidly enhance the functions of complex software [2].

The rapid evolution of software reliability models has led individuals to attempt understanding the nature of program failures and measuring software reliability. There are two main approaches for modeling and assessing program reliability: estimation modeling and prediction modeling. These techniques are essential elements in software reliability modeling to determine if the software product meets its goal and is ready for deployment. Both modeling techniques rely on observing and collecting failure data and analyzing it through statistical inference. These models define the stochastic process to describe the behavior of software failure over time [3].

Reliability is crucial for software quality. Despite different methods to describe it, reliability engineering focuses on analyzing and evaluating software quality, emphasizing the ability of a software product to operate without errors [1]. Predicting software defects, or early software defect prediction, has become vital in software engineering, aiming to help teams release high-quality applications on time [4].

Machine learning (ML), a subfield of artificial intelligence (AI), has gained interest in enhancing various aspects of software engineering, aiding programmers and engineers in creating reliable programs[5] .ML, by extracting hidden patterns from data, can effectively predict software issues (Iqbal et al., 2019). Challenges persist in achieving reliability for applications without a universal solution, driving ongoing ML research [6].

II. SOFTWARE RELIABILITY

Software reliability is a critical factor considered in ensuring the quality of software. In other words, software reliability deals with system failures or errors [7]. Failure and error are distinct factors. Defects can be considered errors or faults introduced during the development phase. Failure occurs due to the presence of one or more errors over time. Generally, it is observed that the failure rate decreases with increased execution time [8].

The rapid evolution of software reliability models has led individuals to attempt to understand the nature of software

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 8, Issue 2, pp 8-18, February-2024
https://doi.org/10.47001/IRJIET/2024.802002

failures and why they occur, as well as to measure software reliability. There are two approaches for modeling and evaluating software reliability: estimation modeling and prediction modeling. These techniques are fundamental in modeling software reliability to determine whether the software product meets its objective and is ready for deployment. Both modeling techniques rely on observing and collecting failure data and analyzing it through statistical inference. These models define a stochastic process to describe the behavior of software failure over time [3].

Deep learning provides autonomous learning and hierarchical representation of features across multiple levels. Unlike traditional machine learning methods, this resilience is a result of the powerful nature of deep learning; in short, the entire deep learning structure is utilized for feature extraction and refinement. The initial layers process incoming data with simple feature learning, and the output is sent to higher layers responsible for learning complex features. Thus, deep learning is suitable for handling larger datasets and greater complexity. Deep learning plays a crucial role in predicting software reliability [9].

These techniques can be employed to analyze software data, such as source code, error logs, and tests, to identify factors that may affect program reliability. This information can be used to create models capable of accurately predicting software reliability [10] .This work aims to demonstrate that incorporating machine learning and deep learning techniques during software development can significantly aid in the early prediction of application reliability.

III. METHODOLOGY

This work, shown in outline in Figure 1, demonstrates a comprehensive approach to building and evaluating deep learning models for predicting software reliability. The process encompasses several critical steps, starting with the loading of six distinct datasets (DS1-DS6). The data undergoes normalization using the Min-Max method, ensuring consistent scaling across features. To enhance model efficiency, feature selection techniques such as Chi-Square, Regression Coefficient, Particle Swarm Optimization (PSO), and Whale Optimization Algorithm (WOA) are applied.

Subsequently, the dataset is strategically split to facilitate training, validation, and testing phases. Three types of neural networks—Feedforward Neural Network (NN), Recurrent Neural Network (RNN), and Long Short-Term Memory (LSTM)—are employed to capture intricate patterns and temporal dependencies within the software reliability data. The models are then trained using the preprocessed and feature-selected datasets.

The evaluation phase involves rigorous assessment metrics to gauge the models' performance. Results are printed to provide a clear overview of the predictive capabilities of each model, offering insights into their accuracy and reliability in software reliability prediction. This block diagram encapsulates a holistic methodology, integrating data preprocessing, feature selection, model training, and evaluation to optimize the prediction of software reliability.

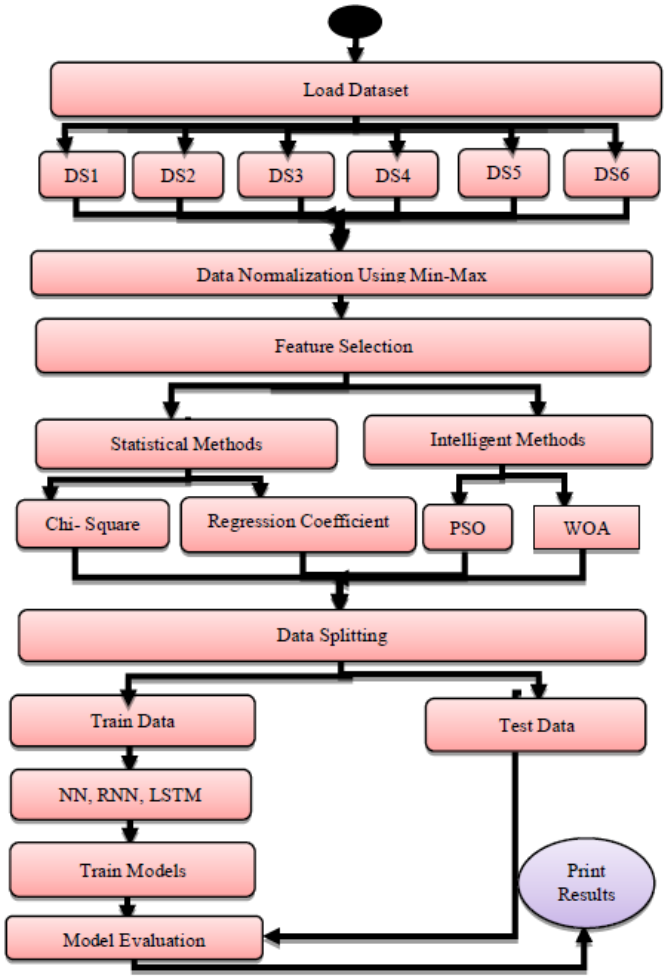

Figure 1: Effectiveness Diagram for the SRDL tool

4.1 Dataset Description

Six datasets, labeled as Dataset1 to Dataset6, were utilized in the research, each providing distinct information for software reliability prediction. The details of each dataset, including the number of features, the number of instances, and the percentage of instances with negative features, are outlined below.

Dataset 1 (DS1): This dataset comprises 39 features, with 516 instances representing unreliable projects and 16,670 instances representing reliable projects. The unreliable instances account for 3% of the total units.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 2, pp 8-18, February-2024

https://doi.org/10.47001/IRJIET/2024.802002

Dataset 2 (DS2): With 22 features, DS2 contains 2,106 instances of unreliable projects and 8,779 instances of reliable projects, constituting 19.3% of the units as unreliable.

Dataset 3 (DS3): Featuring 39 features, DS3 includes 68 instances of unreliable projects and 9,398 instances of reliable projects, making up 0.71% of the units as unreliable.

Dataset 4 (DS4): With 38 features, DS4 consists of 23 instances of unreliable projects and 5,566 instances of reliable projects, representing 0.41% of the units as unreliable.

Dataset 5 (DS5): This dataset, with 22 features, encompasses 326 instances of unreliable projects and 1,783 instances of reliable projects, contributing to 15% of the units being unreliable.

Dataset 6 (DS6): DS6, with 22 features, includes 107 instances of unreliable projects and 415 instances of reliable projects, making up 20% of the units as unreliable.

4.2 Data Normalization Using Min-Max

Min-Max normalization is a type of data scaling technique used to re-scale numerical data within a specific range. It is a linear scaling technique that transforms the original data by mapping it to a new range between a specified minimum and maximum value [11].. The formula for Min-Max normalization is given by:

$$X_{norm} = \frac{X - X_{Min}}{X_{Max} - X_{Min}} \qquad (1)$$

Where (X) is the original data value, (X_{Min}) is the minimum value of the data, (X_{Max}) is the maximum value of the data, andX_{norm} is the normalized value. The resulting normalized values will range between 0 and 1, where 0 represents the minimum limit of the data value, and 1 represents the maximum limit of the data value. Min-Max normalization is beneficial when the range of values in the dataset is large and widely distributed, as it helps improve the performance of machine learning algorithms by reducing the impact of extreme values and enhancing the overall distribution of the data. However, it is important to note that this normalization method may have limitations, such as its inability to handle extreme values or changes in the minimum and maximum values of the dataset.

4.3 Feature Selection

Feature selection is the process of choosing relevant and informative data from the initial dataset to create a set of features for the purpose of data analysis or executing a specific task. Feature selection involves identifying specific attributes or variables that are most crucial for a particular analysis or machine learning algorithm [12]The following is an overview of the methods used in feature selection:

Chi-Square: The Chi-square (χ^2) feature selection method is a statistical technique used to assess the suitability of features in a dataset for a specific target variable. It measures the independence between the feature and the target by comparing the observed frequencies of their occurrences with the expected frequencies under the assumption of independence [13]. The Chi-square test is employed to statistically verify hypotheses. In this type of significance test, it determines whether the observed frequencies obtained through measurement in practice differ from the theoretical frequencies that align with the given null hypothesis. Statistically, the Chi-square statistic is the sum of squared differences between observed frequencies (actual, experimentally determined) and expected frequencies (those proportional to the formulated null hypothesis), divided by the expected frequency for each cell. The formula for the Chi-square statistic used in the Chi-square test is given by [14]:

$$\mathbf{X^2} = \sum \frac{(O-E)^2}{E} \qquad (2)$$

Where $\mathbf{X^2}$ is the Chi-square statistic, O is the observed frequency, and E is the expected frequency corresponding to the formulated null hypothesis. Figure (2) shows the block diagram for chi-square feature selection.

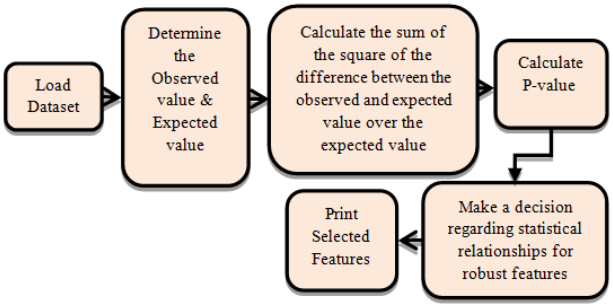

Figure 2: Chi- Square Feature Selection Block Diagram

Regression Coefficient: The proposed feature selection method, called Kernel F-Score, is a fundamental and straightforward technique that measures the discrimination between two classes with real values. In the F-Score method, F-score values are calculated for each feature in the dataset according to equation (3). To determine the features from the entire dataset, the threshold value is obtained by calculating the average F-score value for all features. If the F-score value for any feature is greater than the threshold, this feature is added to the feature space; otherwise, it is removed. Considering training samples, k=1,...,m, where the number of positive and negative n_+, n_-predictions is:

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 2, pp 8-18, February-2024

https://doi.org/10.47001/IRJIET/2024.802002

$$F(i) = \frac{\left(\bar{X}_i^{(+)} - \bar{X}_i\right)^2 + \left(\bar{X}_i^{(-)} - \bar{X}_i\right)^2}{\frac{1}{n_+ - 1}\sum_{k=1}^{n_+}\left(\bar{X}_{k,i}^{(+)} - \bar{X}_i^{(+)}\right)^2 + \frac{1}{n-1}\sum_{n-1}^{n_-}\left(X_{k,i}^{(-)} - \bar{X}_i^{(-)}\right)^2} \quad (3)$$

Here, $\bar{X}_i, \bar{X}_i^{(+)}, \bar{X}_i^{(-)}$represent the average of the $\bar{X}_{k,i}^{(+)}$feature for the entire dataset, positive, and negative classes, respectively. $\bar{X}_{k,i}^{(-)}$ are the ith feature for positive example k and negative example k, respectively. The numerator emphasizes the distinction between the positive and negative classes, while the denominator specifies the grouping within each of these two classes. A larger F-score indicates a higher likelihood that the feature is more discriminative. However, a drawback of the F-Score method is that it does not consider the mutual information between features in its calculations [15]. Figure (3) shows the regression Coefficient feature selection block diagram.

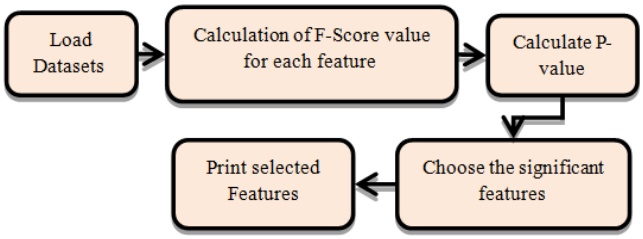

Figure 3: Regression Coefficient Feature Selection Block Diagram

Particle Swarm Optimization: is a popular metaheuristic optimization algorithm inspired by the social behavior of birds and fish. In the context of feature selection, PSO can be effectively utilized to find an optimal subset of features that maximizes a given objective function [16]. Feature selection using Particle Swarm Optimization (PSO) involves several sequential steps, as follows:

1. Initialization of the Swarm: The community is initialized with a set of particles flying in the binary search space, constrained between 0 and 1, aiming to find the best solution. Each particle takes its current position, current velocity, and the best personal solution (Pbest). fter configuring the location, the best location (Pbest), initially the same as the starting location, is assigned as it is the only and best location for the particle and also the Global Best Solution (Gbest).
2. Calculation of the Mean: Calculate the mean for the exchanged information between features and the target.
   $mean = np.mean(feature_{scores})$ (4)
3. Updating Velocity and Position: Update the velocity using the equation:
   $Vi(t) = Vi(t-1) + c1r1\big(Pbest - Xi(t-1)\big) + c2r2\big(Gbest - Xi(t-1)\big)$ (5)
4. Determine the new position using:
   $Xi(t) = Xi(t-1) + Vi(t)$ (6)
5. Fitness Function Calculation: Calculate the fitness function, in this case, the accuracy, using the equation:
   $Accuracy\ fitness = (FP + FN)/(TP + FP + FN + TN)$ (7)
   Update the values of the best personal solution (Pbest) and the best global solution (Gbest).
6. Termination: Repeat the above steps for each particle.

The algorithm repeats itself according to the specified number of iterations. The entire process is iterative, and the algorithm is illustrated using a box plot in Figure (4).

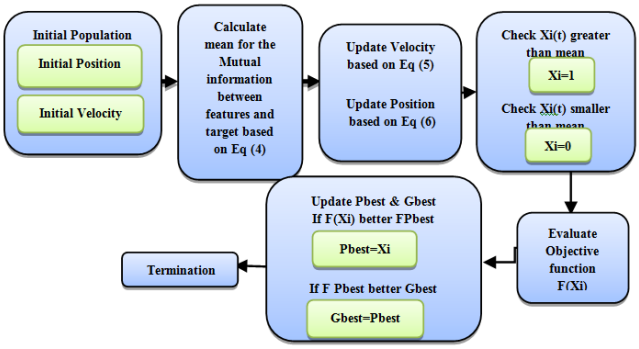

Figure 4: PSO Block Diagram

Whale Optimization Algorithm: is a nature-inspired optimization algorithm that draws inspiration from the social behavior of humpback whales. In the context of feature selection, WOA can be employed to find an optimal subset of features that maximizes a given objective function[17].When applying the Whale Optimization Algorithm (WOA), there are several fundamental steps to be followed, as outlined below:

1. Initialization and Parameter Setting: Start by defining and initializing the basic algorithm parameters, such as the swarm size (N), maximum number of iterations (Max Iteration), dimensions of the search space, and values for P, A, C, Xrand, t, a, l. The search space is typically within [0, 1]. P is a random number between [0, 1] used for updating the whales' positions, while A is used for searching for prey. If P is less than 0.5, compare A, and if A is greater than 1, the whale is considered far from prey, and equations (8) to (9) are applied to determine a new position for the whale. If A is less than 1, equations (10) to (11) are used to encircle the prey.
   $D = |C \times X_{rand} - X|$ (8)
   $X(t+1) = X_{rand} - A \times D$ (9)
   $D = |C \times X'(t) - X(t)|$ (10)
   $X(t+1) = X'(t) - A \times D$ (11)
2. Predation: When P is greater than 0.5, a spiral shape is formed to encircle the prey using equations (11) to (12).

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 2, pp 8-18, February-2024

https://doi.org/10.47001/IRJIET/2024.802002

$$D = |X'(t) - X(t)| \quad (12)$$

$$X(t+1) = D \times e^{b1} \times Cos(2\pi l) + X'(t) \quad (13)$$

3. Cost Function and Fitness Evaluation Use a cost function (Cost) as a fitness measure to evaluate the quality of solutions. The cost for each whale is assigned based on its current performance. The cost function is designed based on the specific goal and problem being addressed, directing the movement and updating of whale positions based on cost values. The cost equation is given by: `Cost = α × error + β × (Number of feature/Len of feature)` (Equation 13). Where α is 0.99, β is 1-α, and error is calculated using Equations (14) and (15).

$$Cost = \alpha \times error + \beta \times (Number\ of\ feature / Len\ of\ feature) \quad (14)$$

$$Accuracy = np.sum(ytest == ypred) / number\ of\ test \quad (15)$$

$$error = 1 - Accuracy \quad (16)$$

4. Termination: If convergence criteria are met, terminate the algorithm and return the best-found solutions, as illustrated in Figure (5).

4.4 Deep Learning Models

Neural Network (NN): Neural Networks are a fundamental component of machine learning, inspired by the structure and functioning of the human brain. Comprising interconnected nodes or neurons organized into layers, NNs excel in learning complex patterns and relationships from data. They consist of an input layer, hidden layers that perform computations, and an output layer. Training involves adjusting the weights between neurons based on the error in predictions, allowing the network to generalize to new, unseen data [18].

Recurrent Neural Network (RNN): RNNs are a specialized class of neural networks designed to handle sequential data by incorporating feedback loops. This enables them to capture temporal dependencies in sequences, making them well-suited for tasks like natural language processing and time series analysis. RNNs maintain a memory of previous inputs, allowing them to consider context and exhibit dynamic behavior crucial for tasks where the order of information matters [19].

Long Short-Term Memory (LSTM): LSTM are a type of RNN designed to address the challenges of learning long-range dependencies in sequences. They incorporate memory cells and gates that regulate the flow of information, mitigating the vanishing gradient problem encountered by traditional RNNs. LSTMs excel in capturing context over extended sequences, making them highly effective in tasks requiring an understanding of context, such as language translation and speech recognition [20].

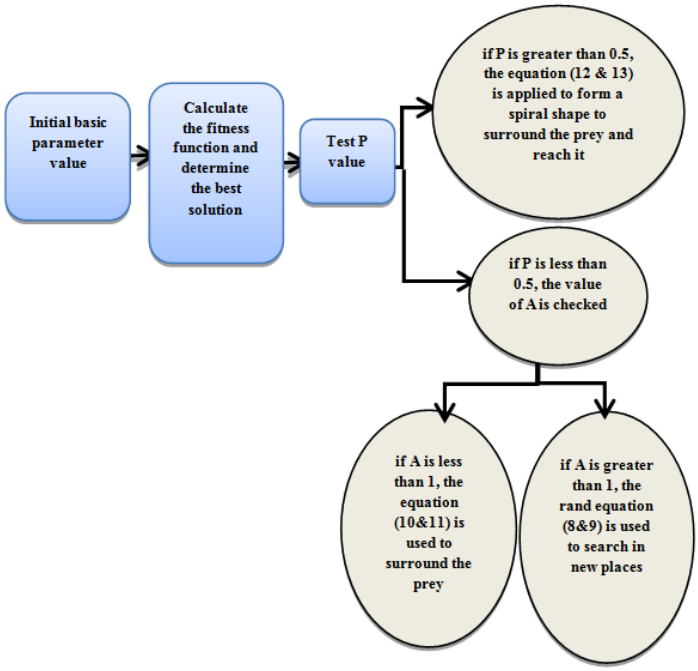

Figure 5: WOA block diagram

4.5 Evaluation Metrics

In Figure 2, can observe a confusion matrix table typically employed to evaluate the effectiveness of a binary classification model. This table comprises the real positives of the target variable marked as True (1) and False (0), alongside the predicted positives of the model designated as Positive (1) and Negative (0). The confusion matrix facilitates the calculation of diverse performance metrics, including accuracy, precision, recall, and F1 score, offering a thorough evaluation of the model's classification capabilities.

1. The confusion matrix is typically divided into four quadrants, which correspond to four possible outcomes:
2. True Positive (TP): The model correctly predicts a positive outcome when the actual outcome is positive.
3. True Negative (TN): The model correctly predicts a negative outcome when the actual outcome is negative.
4. False Positive (FP): The model predicts a positive outcome when the actual outcome is negative (also known as a Type I error).
5. False Negative (FN): The model predicts a negative outcome when the actual outcome is positive (also known as a Type II error) [21].

	Predicted: NO	Predicted: YES
Actual: NO	TN = ??	FP = ??
Actual: YES	FN = ??	TP = ??

Figure 1: Binary Classification Confusion Matrix

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 2, pp 8-18, February-2024

https://doi.org/10.47001/IRJIET/2024.802002

The values of TP, TN, FP, and FN can be used to calculate various performance metrics shown in Table (1):

Table 1: The Elements of the Evaluation Process (Variables, Definitions, and Equations)

Variable	Definition	Equation
Accuracy	The percentage of accurately anticipated data from tests is easily determined by dividing all accurate forecasts by all predictions.	$Accuracy = \frac{Tp + TN}{TP + TN + FP + FN}$
Precision	the proportion of outstanding instances among all anticipated ones from a specific class	$Precision = \frac{TP}{TP + FP}$
Recall	the ratio of the total number of occurrences to the proportion of instances that were supposed to be members of a class	$Recall = \frac{TP}{TP + FN}$
F1-Score	The phrase is used to describe a test's accuracy. The maximum F1-score is 1, which denotes outstanding recall and precision, while the lowest F1-score is 0.	$F1 - Score = 2 \times \frac{percision \times recall}{Percison + recall}$

V. RESULTS

Table (2) provides performance metrics for various algorithms (NN, RNN, LSTM) before feature selection using different techniques (Chi-Square, Regression Coefficient, PSO, WOA). The metrics include Accuracy, Precision, Recall, and F1 Score.

Before feature selection, all three algorithms (NN, RNN, LSTM) consistently demonstrate high performance across the board, with Accuracy, Precision, Recall, and F1 Score all equal to 0.97 or 1.00. This indicates that the models have excellent predictive capabilities on the given data.

It's worth noting that the application of feature selection algorithms (Chi-Square, Regression Coefficient, PSO, WOA) may impact these metrics by selecting subsets of features that contribute the most to the model's performance. If you have specific questions about the impact of feature selection on these metrics or if you'd like further analysis, please provide additional details or questions.

Table 2: DS1 Results

Algorithm	Before Feature Selection			Chi-Square			Regression Coefficient			PSO			WOA		
	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM
Accuracy	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	1	1	1
Precision	0.98	0.97	0.97	0.97	0.97	0.97	0.97	0.98	0.97	0.97	0.97	0.97	1	1	1
Recall	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	1	1	1
F1 Score	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	1	1	1

The results for DS2 before feature selection and after applying Chi-Square, Regression Coefficient, PSO, and WOA feature selection algorithms are presented in the Table (3). Before feature selection, all three algorithms (NN, RNN, LSTM) exhibit

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 8, Issue 2, pp 8-18, February-2024
https://doi.org/10.47001/IRJIET/2024.802002

decent performance on DS2. The Accuracy, Precision, Recall, and F1 Score for NN, RNN, and LSTM are consistently high, ranging from 0.8 to 1.00. After applying feature selection techniques, particularly with PSO and WOA, there is a noticeable improvement in performance. The Accuracy, Precision, Recall, and F1 Score reach near-perfect values of 0.99 or 1.00 for all algorithms. This suggests that the selected features enhance the models' ability to predict outcomes on DS2 significantly. In summary, feature selection using PSO and WOA on DS2 has a positive impact on model performance, resulting in highly accurate and precise predictions across all evaluated metrics. If you have any specific questions or need further analysis, feel free to ask.

Table 3: DS2 Results

Algorithm	Before Feature Selection			Chi-Square			Regression Coefficient			PSO			WOA		
	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM
Accuracy	0.80	0.80	0.80	0.82	0.82	0.82	0.81	0.81	0.81	0.99	0.99	0.96	1.00	1.00	1.00
Precision	0.82	0.81	0.81	0.82	0.83	0.82	0.81	0.82	0.81	0.99	0.99	0.95	1.00	1.00	1.00
Recall	0.96	0.97	0.97	0.99	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	1.00	1.00	1.00
F1 Score	0.88	0.88	0.88	0.9	0.9	0.9	0.89	0.9	0.89	0.99	0.99	0.97	1.00	1.00	1.00

The results for DS3 before feature selection and after applying Chi-Square, Regression Coefficient, PSO, and WOA feature selection algorithms are summarized in the Table (4).

Before feature selection, all three algorithms (NN, RNN, LSTM) already demonstrate excellent performance on DS3, with high Accuracy, Precision, Recall, and F1 Score values ranging from 0.99 to 1.00.

After applying feature selection techniques, particularly with PSO and WOA, there is minimal change in performance since the models were already performing at near-perfect levels. The Accuracy, Precision, Recall, and F1 Score remain consistently high, with some metrics achieving perfect scores (1.00).

In conclusion, DS3 seems to be a dataset where the initial feature set already allows the models to achieve optimal performance. Feature selection techniques may not have a significant impact on improving the models' performance further. If you have any specific questions or need additional analysis, feel free to ask.

Table 4: DS3 Results

Algorithm	Before Feature Selection			Chi-Square			Regression Coefficient			PSO			WOA		
	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM
Accuracy	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	1.00

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 2, pp 8-18, February-2024

https://doi.org/10.47001/IRJIET/2024.802002

Precision	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	1.00	1.00
Recall	0.99	0.99	1.00	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F1 Score	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	1.00	1.00

The results for DS4 before feature selection and after applying Chi-Square, Regression Coefficient, PSO, and WOA feature selection algorithms are summarized in the Table (5). Before feature selection, all three algorithms (NN, RNN, LSTM) exhibit exceptional performance on DS4, with high Accuracy, Precision, Recall, and F1 Score values, all ranging around 0.99. After applying feature selection techniques, particularly with PSO and WOA, there is minimal change in performance since the models were already performing at near-perfect levels. The Accuracy, Precision, Recall, and F1 Score remain consistently high, with some metrics achieving perfect scores (1.00). DS4 appears to be a dataset where the initial feature set already allows the models to achieve optimal performance. Feature selection techniques may not have a significant impact on improving the models' performance further. If you have any specific questions or need additional analysis, feel free to ask.

Table 5: DS4 Results

Algorithm	Before Feature Selection			Chi-Square			Regression Coefficient			PSO			WOA		
	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM
Accuracy	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Precision	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Recall	0.99	0.99	1.00	0.99	0.99	0.99	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
F1 Score	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99

The results for DS5 before feature selection and after applying Chi-Square, Regression Coefficient, PSO, and WOA feature selection algorithms are presented in the Table (6). Before feature selection, the initial performance of the algorithms (NN, RNN, LSTM) on DS5 is relatively good, with Accuracy, Precision, Recall, and F1 Score values ranging between 0.82 and 0.93.After feature selection, especially with PSO and WOA, there is notable improvement in the model performance. Accuracy, Precision, Recall, and F1 Score have increased across the board; with some metrics achieving perfect scores (1.00). This indicates that feature selection has successfully enhanced the models' ability to capture relevant information from the dataset. DS5 benefits significantly from feature selection, particularly with PSO and WOA, leading to improved model performance. If you have any specific questions or need further analysis, feel free to ask.

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 8, Issue 2, pp 8-18, February-2024
https://doi.org/10.47001/IRJIET/2024.802002

Table 6: DS5 Results

Algorithm	Before Feature Selection			Chi-Square			Regression Coefficient			PSO			WOA		
	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM
Accuracy	0.82	0.82	0.82	0.86	0.87	0.87	0.86	0.87	0.87	0.99	0.98	0.91	1.00	1.00	1.00
Precision	0.90	0.90	0.88	0.90	0.89	0.90	0.89	0.89	0.88	0.99	1.00	0.91	1.00	1.00	1.00
Recall	0.89	0.89	0.93	0.93	0.96	0.95	0.94	0.96	0.97	0.99	0.98	0.99	1.00	1.00	1.00
F1 Score	0.89	0.90	0.91	0.92	0.92	0.93	0.92	0.92	0.92	0.99	0.99	0.95	1.00	1.00	1.00

The results for DS6 before feature selection and after applying Chi-Square, Regression Coefficient, PSO, and WOA feature selection algorithms are presented in the Table (7). Before feature selection, the initial performance of the algorithms (NN, RNN, LSTM) on DS6 shows varying levels of accuracy, precision, recall, and F1 Score. The values range from 0.72 to 0.98. After feature selection, especially with PSO and WOA; there is a noticeable improvement in the model performance. Accuracy, Precision, Recall, and F1 Score have increased across the board; with some metrics achieving perfect scores (1.00). This indicates that feature selection has successfully enhanced the models' ability to capture relevant information from the dataset. In summary, DS6 benefits significantly from feature selection, particularly with PSO and WOA, leading to improved model performance. If you have any specific questions or need further analysis, feel free to ask.

Table 7: DS6 Results

Algorithm	Before Feature Selection			Chi-Square			Regression Coefficient			PSO			WOA		
	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM	NN	RNN	LSTM
Accuracy	0.78	0.77	0.78	0.90	0.89	0.89	0.84	0.84	0.84	0.90	0.90	0.90	1.00	1.00	1.00
Precision	0.98	0.98	0.78	0.92	0.92	0.91	0.85	0.85	0.85	0.96	0.96	0.94	1.00	1.00	0.97
Recall	0.73	0.72	0.98	0.96	0.96	0.96	0.98	0.98	0.98	0.91	0.91	0.91	1.00	1.00	1.00
F1 Score	0.84	0.83	0.87	0.94	0.94	0.94	0.91	0.91	0.91	0.93	0.93	0.92	1.00	1.00	0.98

ISSN (online): 2581-3048
Volume 8, Issue 2, pp 8-18, February-2024
https://doi.org/10.47001/IRJIET/2024.802002

The performance of the algorithms (NN, RNN, LSTM) before feature selection exhibited variability across datasets, with distinct levels of effectiveness reflected in metrics such as Accuracy, Precision, Recall, and F1 Score. This diversity underscored the unique challenges posed by each dataset. However, following the application of feature selection algorithms, notably PSO and WOA, a consistent and substantial enhancement in model performance was observed across all datasets. The selected features proved instrumental in augmenting the algorithms' capability to identify pertinent patterns, culminating in elevated values for accuracy, precision, recall, and F1 Score. This underscores the efficacy of feature selection, particularly through PSO and WOA, in optimizing the performance of machine learning models across diverse datasets.

VI. CONCLUSIONS

The research findings demonstrate varying performance across different datasets, with some data exhibiting excellent results regardless of the employed techniques, while others require additional optimization for outstanding performance. Notably, LSTM, RNN, and NN models consistently achieved high performance based on evaluation metrics like Precision, Recall, and F1 Score, showcasing their effectiveness in data classification. Feature selection techniques such as chi-square and regression coefficient did not significantly impact the models' performance, indicating their robustness. Intelligent techniques required fewer epochs for training compared to statistical techniques, suggesting their superiority. WOA technology proved highly effective, particularly in feature selection, where all metrics reached 1.000, highlighting its efficacy. The LSTM model, although time-consuming to train, demonstrated significant performance improvement when combined with the PSO technique and Gaussian Membership Function. Each dataset exhibited varying optimal techniques, with statistical methods and intelligent techniques excelling in different scenarios. Overall, the study provides valuable insights into the performance of diverse models and techniques across multiple datasets.

ACKNOWLEDGEMENT

The authors would like to thank University of Mosul for Support.

REFERENCES

[1] Z. Tian, J. Xiang, S. Zhenxiao, Z. Yi, and Y. Yunqiang, "Software Defect Prediction based on Machine Learning Algorithms," *2019 IEEE 5th Int. Conf. Comput. Commun. ICCC 2019*, pp. 520–525, 2019, doi: 10.1109/ICCC47050.2019.9064412.

[2] M. Pandey, R. Litoriya, and P. Pandey, "Validation of Existing Software Effort Estimation Techniques in Context with Mobile Software Applications," *Wirel. Pers. Commun.*, vol. 110, no. 4, pp. 1659–1677, 2020, doi: 10.1007/s11277-019-06805-0.

[3] H. P. S. D. V. Bansal, "Analysis of Software Reliability Growth Models for Quantitative Evaluation of Software Reliability and Goodness of Fitness Metrics," *Int. J. Sci. Res.*, vol. 6, no. 4, pp. 1540–1548, 2017, doi: 10.21275/ART20172698.

[4] A. Alsaeedi and M. Z. Khan, "Software Defect Prediction Using Supervised Machine Learning and Ensemble Techniques: A Comparative Study," *J. Softw. Eng. Appl.*, vol. 12, no. 05, pp. 85–100, 2019, doi: 10.4236/jsea.2019.125007.

[5] A. Iqbal *et al.*, "Performance analysis of machine learning techniques on software defect prediction using NASA datasets," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 5, pp. 300–308, 2019, doi: 10.14569/ijacsa.2019.0100538.

[6] O. Barack and L. Huang, "Assessment and Prediction of Software Reliability in Mobile Applications," *J. Softw. Eng. Appl.*, vol. 13, no. 09, pp. 179–190, 2020, doi: 10.4236/jsea.2020.139012.

[7] K. Kumaresan and P. Ganeshkumar, "Software reliability modeling using increased failure interval with ANN," *Cluster Comput.*, vol. 22, pp. 3095–3102, 2019, doi: 10.1007/s10586-018-1942-4.

[8] E. E. Ogheneovo, "Software Dysfunction: Why Do Software Fail?," *J. Comput. Commun.*, vol. 02, no. 06, pp. 25–35, 2014, doi: 10.4236/jcc.2014.26004.

[9] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions," *IEEE Access*, vol. 10, pp. 36429–36463, 2022, doi: 10.1109/ACCESS.2022.3151903.

[10] L. Qiao, X. Li, Q. Umer, and P. Guo, "Deep learning based software defect prediction," *Neurocomputing*, vol. 385, pp. 100–110, 2020, doi: 10.1016/j.neucom.2019.11.067.

[11] X. Jin, J. Zhang, J. Kong, T. Su, and Y. Bai, "A Reversible Automatic Selection Normalization (RASN) Deep Network for Predicting in the Smart Agriculture System," *Agronomy*, vol. 12, no. 3, pp. 1–20, 2022, doi: 10.3390/agronomy12030591.

[12] R. Corizzo, E. Zdravevski, M. Russell, A. Vagliano, and N. Japkowicz, "Feature extraction based on word embedding models for intrusion detection in network traffic," *J. Surveillance, Secur. Saf.*, pp. 140–150, 2020, doi: 10.20517/jsss.2020.15.

[13] M. Wang, "Online and Offline Feature Screening and Applications," *Florida State Univ. Coll. Arts Sci. Online*, 2021.

[14] K. Vichova, P. Taraba, and T. Belantova, "Risk

ISSN (online): 2581-3048
Volume 8, Issue 2, pp 8-18, February-2024
https://doi.org/10.47001/IRJIET/2024.802002

management of the project and the use of software in sme," *WSEAS Trans. Bus. Econ.*, vol. 17, pp. 551–559, 2020, doi: 10.37394/23207.2020.17.54.

[15] K. Polat and S. Güneş, "A new feature selection method on classification of medical datasets: Kernel F-score feature selection," *Expert Syst. Appl.*, vol. 36, no. 7, pp. 10367–10373, 2009, doi: 10.1016/j.eswa.2009.01.041.

[16] I. N. Muisyo, C. M. Muriithi, and S. I. Kamau, "Enhancing low voltage ride through capability of grid connected DFIG based WECS using WCA-PSO tuned STATCOM controller," *Heliyon*, vol. 8, no. 8, p. e09999, 2022, doi: 10.1016/j.heliyon.2022.e09999.

[17] O. Almomani, "SS symmetry Detection System Based on PSO , GWO , FFA and," *A Featur. Sel. Model Netw. Intrusion Detect. Syst. Based PSO, GWO, FFA GA Algorithms*, vol. 33, no. 32, pp. 1–22, 2020.

[18] P. S. Kulkarni, S. N. Londhe, and M. C. Deo, "Journal of Soft Computing in Civil Engineering Artificial Neural Networks for Construction Management: A Review ARTICLE INFO ABSTRACT," *J. Soft Comput. Civ. Eng.*, vol. 1, no. 2, p. 70, 2017, [Online]. Available: http://creativecommons.org/licenses/by/4.0/

[19] M. Hibat-Allah, M. Ganahl, L. E. Hayward, R. G. Melko, and J. Carrasquilla, "Recurrent neural network wave functions," *Phys. Rev. Res.*, vol. 2, no. 2, pp. 1–18, 2020, doi: 10.1103/PhysRevResearch.2.023358.

[20] J. Liu and R. Wang, "Research performance prediction for scientists using LSTM neural networks," *CEUR Workshop Proc.*, vol. 2871, pp. 194–201, 2021.

[21] I. C. Dipto, M. A. Rahman, T. Islam, and H. M. M. Rahman, "Prediction of Accident Severity Using Artificial Neural Network: A Comparison of Analytical Capabilities between Python and R," *J. Data Anal. Inf. Process.*, vol. 08, no. 03, pp. 134–157, 2020, doi: 10.4236/jdaip.2020.83008.

Citation of this Article:

Shahbaa I. Khaleel, Lumia Faiz Salih, "Software Reliability Prediction Using Deep Learning and Feature Selection Algorithms" Published in *International Research Journal of Innovations in Engineering and Technology - IRJIET*, Volume 8, Issue 2, pp 8-18, February 2024. Article DOI https://doi.org/10.47001/IRJIET/2024.802002

****

