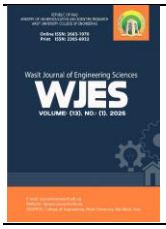

Wasit Journal of Engineering Sciences
Journal homepage: https://ejuow.uowasit.edu.iq

ISSN 2305-6932
Original Research

Vol. 14 No. 2 June 2026, pp.237-247
https://doi.org/10.31185/ejuow.v14i2.855

A Hybrid Approach for Improving Machine Learning Models Using Federated Learning and Ensemble Techniques

Yahia Eed Ali
Computer Engineering Department, Islamic Azad University, South Tehran Branch, Tehran, Iran

Corresponding Author Email:yahia.engsw@gmail.com

Received Dec.12, 2025 | Revised Mar.14, 2026 | Accepted Mar.22, 2026 | Online Jun.1, 2026

Abstract

To have robust software systems, software reliability prediction plays a major role. A hybrid and advanced design to enhance the quality and performance of machine learning models are proposed in this research; it consists of different steps of data preprocessing, feature engineering, model refinement with the assistance of federated learning, hyperparameter optimization, and the hybrid learning methods. One initially works with the data with the help of algorithms like SMOTE and SMOTEENN to address the issue of uneven data, and then the elements are resolved and adjusted in an innovative manner. The ensemble learning and hyperparameter optimization are then used to optimize the models. Specifically, data privacy in the distributed models is maintained using the federated learning approach. Lastly, generated models are tested on complex data with the help of deep neural nets (DNN) and their performance is measured with the means of precision, recall, prediction accuracy, and F1 score. The resulting methods, which are mainly useful in assisting to enhance the efficiency and prediction accuracy of machine learning models, apply in complex problems.

Keywords: Machine learning, SMOTEENN, federated learning, hyperparameter optimization, hybrid learning, deep neural networks.

1. Introduction

Today, software is part of every aspect of our society. This makes creating reliable software a major challenge for developers. As a result, the impact and cost of software errors are increasing dramatically; for example, according to a report by Transcendental Software, software errors cost the global economy an estimated $1.7 trillion in 2017. Therefore, techniques that help developers identify software errors to improve software reliability are in high demand.

To build reliable software, many software reliability techniques have emerged in recent years to help developers improve software reliability by identifying defects, for example: Statistical defect prediction uses software metrics to form classes and predict unknown defects in source code. Static defect detection uses static code analysis to find patterns and identify software errors. Regression testing aims to find broken functions early by planning and running test cases [1]. Code review looks for potential quality issues and seeks to improve maintainability and software quality before merging code changes into the source code repository 2, etc.

Although these techniques have already been widely used in the industry and have been proven to help identify defects [2], Many of the advanced techniques perform poorly in terms of efficiency, which leads to low accuracy, and effectiveness, which means they require a lot of time. This performance cannot match the increasing complexity of modern software systems.

In this paper, we present a hybrid approach that uses ensemble learning to address these challenges. With the ability to combine multiple learning models, ensemble learning is applied to extract valuable insights from various forms of software artifacts produced during the software development lifecycle. These artifacts include

code review histories, problem reports, bug and patch histories, software documentation, and source code, which are efficiently processed using advanced techniques and existing practices to improve software reliability.
To enhance the accuracy and performance, our method consolidates various stages including preprocessing of data, federated learning, optimizing models, and ensemble learning. Data preprocessing involves the process of preparing the data through processing of missing values, encoding of categorical variables, as well as scaling of features to exert more control over the data and make it more compatible with learning. The methods SMOTE and SMOTEENN are employed to deal with data imbalance and balance the dataset so that the learning performance of the model would be optimized. Moreover, for hyperparameter optimization, we apply the Bayesian optimization, and it enhances the performance of different models by efficiently searching the hyperparameter space. Lastly, we prove that ensemble learning methods, these preprocessing and optimization procedures, significantly enhance the model performances. In this paper, the researcher explains the benefits of stack models such as Random Forest, Light GBM, and XG Boost in prediction accuracy. To accomplish this complex patterns and insights, we can make use of using this group approach and in doing so greatly improve the software reliability prediction.

2. Related works

The application of machine learning and deep learning to predict the occurrence of software defects has experienced an increase in recent years. These techniques have become essential instruments of developers and engineers since they can extremely recognize in-depth patterns and comprehend nonlinear associations in data. Recently, the authors conducted research on the topic of defect prediction with the help of deep neural networks and hybrid models [3-6]. They have shown that deep learning techniques can be more effective than conventional one [7]. Software defect prediction (SDP) [8] faces the major challenge of unbalanced class distribution, where defective samples are in the minority. Recent research has turned to combining data balancing techniques and powerful learning algorithms to overcome this issue. In this regard, the present study implemented a three-step hybrid approach on NASA MDP dataset with the aim of improving the prediction accuracy and identifying effective features: first, using the SMOTE technique to balance the classes, then applying the recursive feature elimination (RFE) algorithm to select the most optimal features, and finally classifying with the Random Forest algorithm. The results showed that this hybrid method significantly improved the prediction accuracy in most datasets, and the highest accuracy (0.9998) was achieved in the MC1 dataset. Also, feature importance analysis identified two criteria, "maintenance intensity" and "cyclomatic density" as the most important factors in modeling software defects. In [9], the authors tested three ensemble methods: mean, median and inverse rank weighted mean (IRWM). They built three ensemble models with the above combining rules using KNN, SVM, MLP, regression techniques. The models were tested using Miyazaki, Albrecht datasets. The results show that for Miyazaki dataset model built with IRWM provides the best results and for Albrecht dataset model built with IRWM provides the second best results compared to individual techniques and two other ensemble models. The study also performed parameter tuning for the techniques using trial and error. When predicting software reliability, the main problem arises due to data imbalance. This creates chaos for software developers or researchers to deal with unbalanced faulty data. Early prediction causes major problems. Researchers are looking for software defect prediction approaches based on ML algorithms, although they have not yet looked into it in depth. This work has reviewed and evaluated effective ML approaches for software reliability prediction in terms of dealing with some of these conditions [10]. In addition, in another study, Arunima Jaiswal and her colleagues investigated software reliability prediction using machine learning techniques [11]. In another study, Sweta Mehta and her colleagues investigated improved prediction of software defects using ensemble machine learning techniques [12].
Moreover, the problem of data imbalance is also one of the dominant issues in this sphere. Missing data and unequal distribution of samples in both classes may decrease the quality of models. To address this, issue some strategies like SMOTE (Artificial Minority Oversampling Technique), SMOTEENN (Combination of SMOTE with Modified Neighboring Method) have been proposed. Manipulation of underrepresented data is well represented by these methods, which construct artificial examples [13]. There are also studies where the combination of these methods with the process of reinforcement learning was used to select more efficient features [14].
Ensemble learning has been highly addressed in the area of prediction modeling with an aim of improving the accuracy of the end prediction. Such practices assist in enhancing the general performance of the model through uniting a number of various models. At that, a sequence of the models of Random Forest, XGBoost, and

LightGBM, trained with Stacking techniques, has been suggested to better predict software defects [15]. Stacking is one of the effective ensemble learning methods because the final model can benefit the potential of the various base models and achieve superior results.

The other novel practice is federated learning that has been suggested as a way of maintaining privacy of data and carry out decentralized model training. In this approach, data are stored in various machines and models are trained locally and the outcomes are sent to a central server to update the final model [16]. The approach is particularly contributing to the situations when the privacy and confidentiality of the data are delicate. Recently, the authors conducted research on the topic of federated learning [17, 18].

Lastly, hyperparameter optimization is rather significant to make models more efficient. Previously, this task was performed with the help of traditional algorithms like Grid Search and Random Search, nowadays, the Bayesian optimization is considered an intelligent approach towards searching the most efficient hyperparameters combinations. This technique relies on probabilistic models like Gaussian processes to carry out more precise search and the model will find the optimum settings in the least amount of time [19].

3. Material and methods

In the section, we outline the manner in which we intend to develop and enhance machine learning models through the use of advanced methods and hybrid approaches. The methodology employs several steps that deal with information and model related problems in a logical and efficient way and employs them effectively to obtain greater accuracy and better performance. This will involve preprocessing of the data, data with the federated learning algorithm and the model optimization; as well as ensemble learning algorithms.

The NASA Defect Dataset is a publicly available dataset which NASA provided to simulate and predict the occurrence of software defects. This data is information regarding different software projects where different properties like number of lines of code, number of changes made, internal complexity of the software code among other properties that have to do with the software code are gathered. The primary aim of this dataset is to allow predicting the likelihood of software code defects assimilating the available features. The data is applied particularly in the research of the software defect prediction and software system quality measures, and expands the researchers and developers with a range of resources to examine and enhance software prediction models.

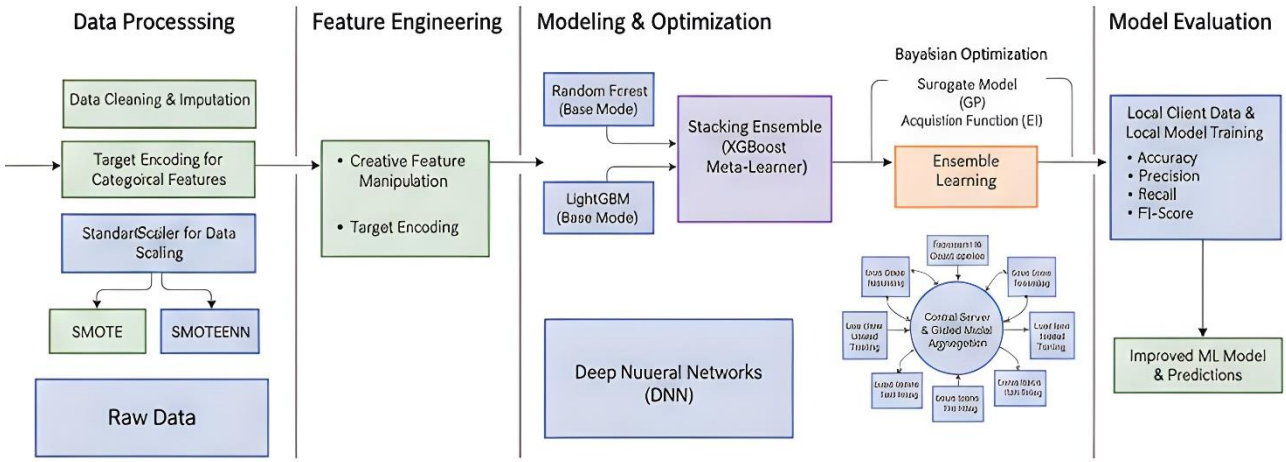

Figure 1. The framework of proposed method

3.1. Data Preprocessing

Data preprocessing is the initial step in the machine learning process in order to get high accuracy and optimum model performance. The data should be completely analyzed after which it should be appropriately readied to be subjected to model learning. The data cleaning, coding of categorical features, and the scaling of data are some of the operations of this step. Averting missing or incomplete data is determined and fixed in a way that the model will be fed with clean and complete data. Also, the target coding is used to code categorical features

rather than in a more conventional way, such as one-hot coding. In this way, the model will be able to draw more information that exists in the categorical features.
Scaling of data is then done to enhance model performance. Application of StandardScaler at this point, makes all features have equal scales so that the model may be used to the best capacity. The process of transforming the data features such that the mean of their data is equal to zero and the standard deviation is equal to one is known as data standardization/scaling. The mathematical estimation of standardization is as follows:

$$z = \frac{x - \mu}{\sigma} \quad (1)$$

Where x denotes the original value of the feature, μ denotes the mean of the feature values and x denotes a standard deviation of the feature values. The outcome of this calculation, z, will be the standardized feature value, which has been scaled and that also lies within the new range. This approach assists models to exploit data better and precisely. These pruning processes assist the model in comprehending mentioned associations of attributes among features and learning more effectively.

3.2. Dealing with Imbalanced Data Using SMOTE and SMOTEENN

One of the most prevalent issues of machine learning problems is that unbalanced data may result in the reduction of the precision of the models. The techniques that are applied to the SMOTE method (Artificial Minority Oversampling Technique) and SMOTEENN (Combination of SMOTE with Modified Neighboring Method) are used to address this issue to the proposed method.
SMOTE is the data active in creating new data on the minority classes and forms a more balanced representation of data. Conversely, SMOTEENN is a hybrid of SMOTE and the Edited Nearest Neighbors (ENN), which besides creating new data, removable outliers and errors, the quality of the training data is also enhanced. The methods also enhance unbalanced data, as well as make the model simulate the properties and patterns that each class is linked to in a better way.
Supposing that x_i represents an instance of the minority class, and x_{zi} is the closest instance of x_i that belongs to the majority, and a new synthetic example known as x new is generated in the following way:

$$x_{new} = x_i + \delta \times (x_{zi} - x_i) \quad (2)$$

The word x_i is the example of the minority class originally, x_{zi} is one of the nearest neighbors of x_i of the same class and 8 is a random number between 0 and 1 which is randomly chosen to bring variety to the new data.

3.3. Feature Engineering

Machine learning models require feature engineering [20] as one of the critical components to enhance their accuracy. Here, the existing features are manipulated or combined in an inventive way resulting in new features that are meaningful and powerful that allow simulation of more complex patterns by the model. As an illustration, the nonlinear patterns in data can be simulated with the use of composite features (e.g., multiplication or addition of the existing features) that allow the model to learn more complicated relationships. Besides, Target Encoding is used to encode categorical features. The approach is more effective, as compared to conventional approaches like one-hot encoding, in the sense that it enables the model to perceive directly the links among the qualities that are categorical and the target.

3.4. Modeling using Ensemble Learning

Besides, prediction accuracy is enhanced with the help of ensemble learning models [21]. In this approach, various models are applied together in a way that the resultant model will enjoy the arguable advantage of each [22]. Specifically, to enhance performance, a combination of several base models is resorted to, i.e., Stacking. Such base models as Random Forest and LightGBM are used within this approach and are subsequently joined with a final one, like XGBoost. This method enables the end model to develop various characteristics in a better way and enhance the performance of its prediction.

3.5. Hyperparameter Optimization Using Bayesian Optimization

The idea of hyperparameter optimization is essential towards enhancing the performance of the machine learning models. Rather than employing classical and slow search methods like Grid Search or Random Search methods, the presented methodology employs the Bayesian optimization to exhaustively search through the

parameter space. The method of Bayesian optimization with a probabilistic approach balances exploration and exploitation in order to identify the best parameter set in the shortest time possible. Bayesian optimization is a concept based on two essential parts:

3.5.1. The Surrogate Model (Gaussian Process)

The objective function is modelled with the help of a surrogate model that is typically a Gaussian process (GP) [23] based on the results of previous experiments. The GP gives a definitely estimative forecast of the model performance f(x) at each combination of hyperparameters x. The model becomes the expected mean (μ) of the performance, and the uncertainty (σ) associated with the expected mean. Such a role enables the optimizer to keep time tapping its beliefs concerning the most desirable parts of the parameter space.

$$f(x) \sim \mathcal{GP}\big(m(x), k(x, x')\big) \quad (3)$$

3.5.2. The Acquisition Function

The acquisition function uses the outputs μ and σ of the surrogate model to determine the next optimal experiment location x_{t+1}. The most common acquisition function is Expected Improvement (EI). The EI function quantifies the expected gain in performance over the best observed performance so far, $f(x^+)$. By maximizing the EI function, the algorithm selects the next hyperparameter set that offers the best balance between exploiting known high-performance regions (μ) and exploring uncertain but potentially better regions (σ).

$$EI(X) = E[\max(0, f(x) - f(x^+))] \quad (4)$$

This effective iterative algorithm is guaranteed to effectively scan the search space and by far makes models better when a larger number of hyperparameters is optimized in a relatively short period of time.

3.6. Federated Learning

In the present research, there is the federated learning method employed to maintain data privacy. Federated learning Data are stored locally, and models are trained locally, as opposed to being gathered and sent to a central server. The local models are then used to send the update to a central server to update the final model in a hybrid fashion. This method comes in handy especially in a situation where privacy and data security are an issue of concern. The benefits of federated learning are low data transfer demands, data privacy, and high scalability.

3.7. Deep Neural Networks

The decorations joined federate deep neural networks (DNNs) [24] to depict the associations amongst information and features that are more intricate. These models specifically fit well with complex non_linear relationships of data. To do this, the neural network model is intercepted with dense layers, in such that the layer extracts more complex features and a model is more capable of simulating the nonlinear patterns.

3.8. Model Evaluation

Lastly, the trained models are also checked by different measures. Measures like (Accuracy), (Precision), (Recall) and (F1 Score).

Accuracy [25] is an idea that describes the extent to which the model is able to predict the outputs accurately. The accuracy allows identifying the effectiveness of training the model and the overall performance of the latter rather quickly. Nevertheless, such a measure is not in a position to give clear-cut information regarding the specifics of the model performance. The result is the accuracy that is calculated through the use of equation (5).

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \quad (5)$$

Recall [26] is one of the important statistics used in assessing the efficacy of classification models, that measure the capability of the model to recognize actual positives. That is, sensitivity shows the rate of positive results that are true and correctly represented by the model. The sentiment calculation formula is given as below:

$$Recall = \frac{TP}{(TP + FN)} \quad (6)$$

Among the metrics, which are crucial in measuring the performance of the classification models, one should sing mainly precision, which is the degree to which the model makes a positive prediction. To put it another way, precision is the ratio of predicted positive examples which turn out to be actually positive. This is particularly vital when the false positive prediction is very costly. The accuracy formula is listed below:

$$Precision = \frac{TP}{(TP + FP)} \tag{7}$$

The F1 score [27] is a combined measure that takes into account the accuracy and the bias of a model at the same time and it is quite applicable when classifying models when the data is not balanced. It is particularly when a reduction in both positive and negative errors is equally important, and this metric comes in handy. It calculates the F1 score as the average of both precision and sensitivity and is determined as below:

$$F1\ score = \frac{2 * (Recall * Precision)}{(Recall + Precision)} \tag{8}$$

4. Theoretical foundations

Multi-learning methods are significant in software reliability prediction to increase the accuracy and stability in predictive models. The theory involved in the multi-learning is on the conceptualization of combining multiple basic models to advance the performance. These basic models include decision trees, neural networks, and regression models which are trained on some subset of data or features. This inconsistency helps models to monitor various aspects of the software building procedure and infrastructure problems trends.

The bias and variability problems that are common with machine learning models are best mitigated using multi-learning methods. Multi-learning methods attempt to provide more reliable, predictive, and accurate forecasts than single models by combining the output of multiple models. The proposed methodology enhances multi-learning by using federated learning strategies to make distributed training possible on many different sources of training and remain confidential. The approach will ensure that models are trained in a decentralized fashion, hence scale increase and reduced cost of data transfer.

The basic principle of multi-learning procedures is to engage a wide range of primitive learners in order to enhance prediction scores. As an example, decision trees are highly proficient at the representation of a particular interaction in the data, but neural networks are highly skilled at the depiction of a complex and nonlinear relationship. Collective learning with the help of different models combines the weaknesses of each model resulting in stronger and better solutions.

The key role in this paradigm is that federated learning enables the usage of decentralized information across multiple sources and thus it mitigates the privacy concerns and increases the scalability of the models. The method is particularly useful in dealing with sensitive information or where the centralization of information is not suitable to go. Another way in which federated learning improves the privacy of predictive models and the effectiveness performance in general is by making sure that model updates are locally executed and only compiled centrally.

A number of crowdsourced learning methods have usually been used to predict software reliability. The key approaches along with the integration of other improvements to.

Improve accuracy, include the following: Preliminary clustering creates two or more datasets based on resampling the original data, learns separate base model using the preliminary model sets, and combines the resulting predictions by averaging or voting. This reduces variance and aids in avoiding overfitting especially when the type of baseline model is unstable such as decision trees. Clustering performance could be further improved by the incorporation of SMOTE in order to modify imbalances in classes and by the use of target encoding in attributes of the classes.

Random forests generalize variability of clustering with decision tree training on random features grouping of features in every branch. The result is reduced correlations of trees thereby enhancing diversity and generalization. Random forests prove very effective in combination with other methods, like SMOTEENN that does not only generate fake samples but also eliminates outliers, improving the level of data and the model performance.

Gradient boosting machines (GBMs) train models in stages, in each step a new model is trained to rectify the mistakes of the last model. This method is especially efficient when it comes to handling the complicated relationships of the data and is very accurate. Nonetheless, the GBMs are prone to overfitting unless they are

properly parameterized. We use a Bayesian optimization to optimize GBMs to enable the effective tuning around hyperparameters and so enhance our model performance in a resource-efficient manner. Stacking involves training different models and then using a multi-learner in order to combine their predictions. A variety of different learners that are typically simpler models, like the logistic or the linear regression, are trained to optimally combine the results of the underlying models to produce a final prediction. The method aids in the combination of the strengths of various models that helps to improve a general accuracy.

Voting groups amalgamate the forecast of many independent models through a voting system. Majority voting is applied in classification tasks and the mean of the predictions in regression tasks. The effectiveness of the voting is implicated in the disparity and correctness of underlying models. This technique can be readily applied, but it can be fairly inefficient compared to such procedures as stacking or boosting. All these types of grouping have their own special benefits in regard to software reliability prediction. Clustering method depends on the peculiarities of the dataset used and software prediction of reliability under investigation. When we combine these clustering methods with SMOTE, SMOTEENN, and Bayesian optimization, we add a lot of predictive power, better scalability and robustness to the model. Such a combined solution allows to work with unbalanced data more effectively, predict much better, which, in general, will raise the level of software reliability.

With such developed cluster learning methods, they are integrated with federated learning and efficient hypertransactional optimization that will enable software programmers to be better predictors of software failures and better plan their simulation testing methods and, ultimately, provide more reliable software. The real implementation of such approaches involves the strictness in the choice of the models used in the framework, the proper data preprocessing, and the ongoing monitoring of the results with the help of the performance measurement to provide the optimal outcome.

5. Results

In comparing machine learning models to predict the reliability of the program, it is important to measure its accuracy and effectiveness using the diverse metrics. These measures will involve accuracy, F1 score, precision and recall, which demonstrate in one way or another how the models have simulated and predicted various data. The section is an evaluation and comparison of the different models that are considered in predicting program reliability. The findings of the assessment of the various models are highlighted in Table 1. The accuracy, F1 score, precision and recall of the various models are compared in this table. On the basis of these evaluations, the proposed model scored better on precision, F1, and recall, compared to the rest of the models. In this study, the data were partitioned using preprocessing techniques such as feature standardization and SMOTEENN to correct imbalance. Then, the LGBMClassifier model with random optimization was used to select the best parameters. The model was evaluated using 3-fold cross-validation and the accuracy, ROC-AUC, and classification report metrics, which yielded favorable results. The analysis of the results is done individually by each metric and bar graphs plotted.

Table 1. shows the results obtained from the evaluation of different models

Model	Accuracy	F1 Score	Precision	Recall
SVM	0.84	0.61	0.80	0.31
Logistic Regression	0.83	0.59	0.79	0.28
Random Forest	0.84	0.69	0.83	0.47
XGBoost	0.85	0.69	0.83	0.47
ANN	0.85	0.68	0.83	0.44
Autoencoder	0.85	0.67	0.82	0.43
DBN	0.84	0.65	0.81	0.38
LSTM	0.87	0.77	0.87	0.61
Proposed Method	0.96	0.95	0.97	0.96

According to the bar chart in Figure 2, it is evident that the proposed method delivers the most performance with an accuracy of 96 which is much better than other models such as LSTM (87%) and random forest (84%). This means that the suggested approach works quite well to make right projections on the target sample, which is significant in predicting software reliability.

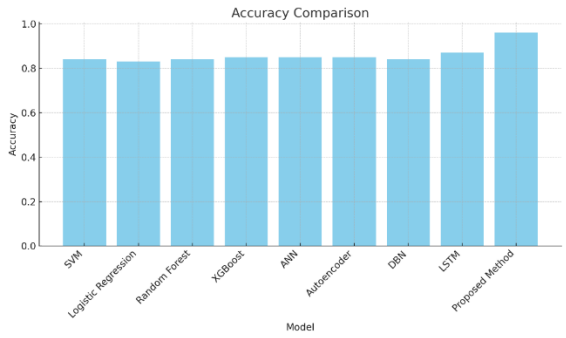

Figure 2. Comparison Accuracy in different models

The given approach again opens with a huge F1 rating of 95% indicating a good balance between the classification and the sentiment among all classes. Compared to the models like LSTM (77%) and Random Forest (69%), there is a lower performance property, thus highlighting the better potential of the suggested technique in operating overall performance in different software reliability conditions. Figure 3 indicates this benchmark.

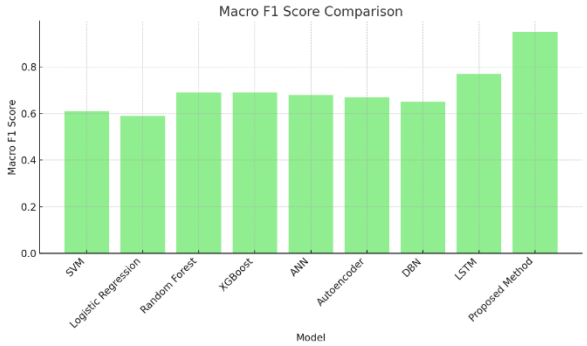

Figure 3. Comparison Macro F1 Score in different models.

The effectiveness of the suggested method is greatest at 97, meaning that in instances where the model gives a positive result (i.e. failure of the software), the model is most accurate when giving predictions with high confidence. The accuracy of LSTM and XGBoost is also good and takes 87 percent accuracy, still lower than the accuracy of the proposed method, thus showing that our model is more effective in reducing the rate of false positives. Figure 4 illustrates this measure.

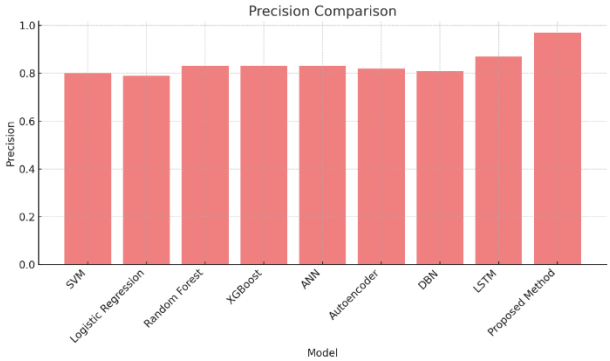

Figure 4. Comparison Precision in different models.

The other space that is well achieved by the proposed method is the recall with 96 meaning that it identifies nearly all the true positives including software failures. The LSTM model (61) is doing well in comparison but not as good as the proposed method in the process of establishing true positives. This indicator is presented in Figure 5.

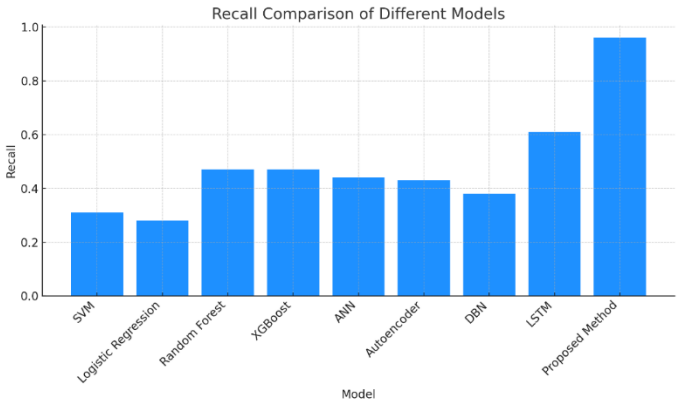

Figure 5. Comparison Precision in different models.

Consequently, this proposed method is better than any of the models in all measures and is therefore the most viable model used in the prediction of software reliability. These findings show that our hybrid system that combines the use of ensemble learning and high level model optimization algorithms are quite capable of predicting software bugs and enhancing software system reliability in general.

6. Discussion

In this study, several machine learning models were evaluated to compare the performance of the proposed model with other popular models. The validity threats of this study include random data selection, cross-validation settings, and data preprocessing, which may affect the accuracy of the models. The results showed that the proposed model performed much better than other models with 96% accuracy, F1-Score 0.95, and recall 0.96. In particular, the LSTM model with 87% accuracy and other models such as Random Forest and XGBoost with 84% accuracy and F1-Score 0.69 performed acceptable, but the proposed model was able to significantly improve these values. Also, comparison with neural network-based models such as ANN and Autoencoder with 85% accuracy shows that the proposed method performed better in data simulation, especially in unbalanced problems.

This shows that it was developed to deal with high level data and could be used to determine the significance of various components. Nonetheless, our superior performance of the proposed approach is not exploited fully despite these strengths of the approach. We use a hybrid model that is more New York-type, which involves the use of innovative methods in terms of machine learning, preprocessing, and co-learning. It excels among all models of the benchmark with an amazing absolute accuracy and sensitivity of 0.96. This greatly eliminates uncertainty in few problems as well as constraints of conventional theories that emphasize on traditional trends. The performance optimization was also performed to improve the parameters to better the performance thereby making sure that the model is fitting the task.

Although our findings are positive, they can be improved. As an example, despite the success of finding sequential dependencies with deep learning, including LSTM, it may be possible to further refine the network infrastructure to perform better. Moreover, it is possible to combine other hybrid approaches, including the transformational learning or the unsupervised learning, to develop a model that could process various and irregular software's. In general, these findings suggest that hybrid design, in particular, the integration of various methods, is useful in finding an accurate and clear position on the issue of analytical problems.

7. Conclusion

The hybrid research approach of dance learning is a contributing factor to machine learning to generate strong predictive analytics of any automotive industry. It is a combination of data, driving methods, teleportation learning, and transactional optimization. The performance of these methods as compared to each other has

revealed that some of these traditional prediction methods, such as SVMs and logistic regression are performing well, but not comparable to higher performance levels of new methodologies, such as randomization, XGBoost, and long-term memory. Several scholars have employed sophisticated models attaining amazing outcomes in all measures, such as precision and effectiveness. It relies on the application of the central component and new methods of large-scale and hybrid learning in order to capture the complex relationships in the underlying data without losing the information consumption and data safety. Besides, the use of Bayesian optimization on transactions enhanced the model, minimizing both training and testing time as well as allowing the use of hybrid learning in real-world applications. This study shows a high-precision hybrid learning approach is a combination of all the strengths of predictive analytics to develop a distinct and the most accurate predictive model.

Declaration of Competing Interest

The author declares that there is no conflict of interest regarding the publication of this manuscript

Funding Information

No funding was received from any financial organization to conduct this research

Author Contributions

The author developed a hybrid framework that integrates federated learning with ensemble learning to leverage distributed data while improving prediction accuracy

Acknowledgments

The author express sincere gratitude to Wasit University/ College of engineering /electrical engineering department in Alkut-Wasit –Iraq for their support conducting this study

References

[1] M. J. Orr, "Introduction to radial basis function networks," Technical Report, center for cognitive science, University of Edinburgh …, 1996.

[2] N. R. Kiran, and V. Ravi, "Software reliability prediction by soft computing techniques," *Journal of Systems and Software,* vol. 81, no. 4, pp. 576-583, 2008.

[3] M. N. Islam, M. M. H. Rimon, S. S.-E. Shamim, Z. M. Fahad, M. J. I. Mony, and M. J. U. Chowdhury, "An Improved Ensemble-Based Machine Learning Model with Feature Optimization for Early Diabetes Prediction," *arXiv preprint arXiv:2512.02023,* 2025.

[4] J. Wang, and J. Gao, "Stacking ensemble learning algorithm based rapid inverse modelling of copper grade using imaging spectral data," *Chemometrics and Intelligent Laboratory Systems,* vol. 257, pp. 105308, 2025.

[5] G. Qian, Z. Zhu, P. Guo, L. Liu, Y. Sun, Y. Zheng, X. Han, and M. Ouyang, "Non-destructive and adaptive negative electrode impedance estimation of lithium-ion batteries using ensemble learning," *Applied Energy,* vol. 402, pp. 127017, 2026.

[6] A. Sabeena, and M. Jeyakumar, "An ensemble learning approach using deep learning models for diabetic retinopathy severity classification," *Biomedical Materials & Devices,* vol. 4, no. 1, pp. 1117-1133, 2026.

[7] S. Zhang, S. Jiang, and Y. Yan, "A hierarchical feature ensemble deep learning approach for software defect prediction," *International Journal of Software Engineering and Knowledge Engineering,* vol. 33, no. 04, pp. 543-573, 2023.

[8] H. Ghinaya, R. Herteno, M. R. Faisal, A. Farmadi, and F. Indriani, "Analysis of Important Features in Software Defect Prediction Using Synthetic Minority Oversampling Techniques (SMOTE), Recursive Feature Elimination (RFE) and Random Forest," *Journal of Electronics, Electromedical Engineering, and Medical Informatics,* vol. 6, no. 3, pp. 276-288, 2024.

[9] Y. Mori, H. Yokota, I. Hoshino, Y. Iwatate, K. Wakamatsu, T. Uno, and H. Suyari, "Deep learning-based gene selection in comprehensive gene analysis in pancreatic cancer," *Scientific reports,* vol. 11, no. 1, pp. 16521, 2021.

[10] N. Yadav, and V. Yadav, "Software reliability prediction and optimization using machine learning algorithms: A review," *Journal of Integrated Science and Technology*, vol. 11, no. 1, pp. 457-457, 2023.
[11] A. Jaiswal, and R. Malhotra, "Software reliability prediction using machine learning techniques," *International Journal of System Assurance Engineering and Management*, vol. 9, no. 1, pp. 230-244, 2018.
[12] S. Mehta, and K. S. Patnaik, "Improved prediction of software defects using ensemble machine learning techniques," *Neural Computing and Applications*, vol. 33, no. 16, pp. 10551-10562, 2021.
[13] R. Malhotra, and J. Jain, "Handling imbalanced data using ensemble learning in software defect prediction." pp. 300-304.
[14] M. S. Ibrahim, Y. M. Mohialden, and D. M. A. A. Afraji, "Q-Learning-Based Feature Selection for Software Defect Prediction," *Al-Iraqia Journal for Scientific Engineering Research*, vol. 4, no. 3, pp. 12-20, 2025.
[15] B. Isong, and E. Igo, "Ensemble Learning for Software Defect Prediction: Performance, Practicality and Future Directions," *Journal of Information Systems and Informatics*, vol. 7, no. 3, pp. 2245-2291, 2025.
[16] Y. Zheng, C.-H. Chang, S.-H. Huang, P.-Y. Chen, and S. Picek, "An overview of trustworthy AI: advances in IP protection, privacy-preserving federated learning, security verification, and GAI safety alignment," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2024.
[17] S. Bakas, X. Li, P. Shah, and H. R. Roth, "Federated Learning in Healthcare: From Research to Real-World Deployment," *Annual Review of Biomedical Engineering*, vol. 28, 2026.
[18] Z. Zhou, S. Yang, F. Zhao, and X. Ren, "FairGFL: Privacy-Preserving Fairness-Aware Federated Learning with Overlapping Subgraphs," *IEEE Transactions on Parallel and Distributed Systems*, 2026.
[19] T. Lamba, and A. Mishra, "Optimal machine learning model for software defect prediction," *International Journal of Intelligent Systems and Applications*, vol. 10, no. 2, pp. 36, 2019.
[20] M. Kuhn, and K. Johnson, *Feature engineering and selection: A practical approach for predictive models*: Chapman and Hall/CRC, 2019.
[21] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*: CRC press, 2025.
[22] G. Kunapuli, *Ensemble methods for machine learning*: Simon and Schuster, 2023.
[23] M. Seeger, "Gaussian processes for machine learning," *International journal of neural systems*, vol. 14, no. 02, pp. 69-106, 2004.
[24] K. Kawaguchi, L. Zhang, and Z. Deng, "Understanding dynamics of nonlinear representation learning and its application," *Neural computation*, vol. 34, no. 4, pp. 991-1018, 2022.
[25] A. Hammouri, M. Hammad, M. Alnabhan, and F. Alsarayrah, "Software bug prediction using machine learning approach," *International journal of advanced computer science and applications*, vol. 9, no. 2, 2018.
[26] T. Saito, and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PloS one*, vol. 10, no. 3, pp. e0118432, 2015.
[27] D. Powers, "Ailab. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation," *J. Mach. Learn. Technol*, vol. 2, no. 22293981, pp. 01, 2011.