Archives available at journals.mriindia.com

International Journal of Recent Advances in Engineering and Technology

ISSN: 2347-2812
Volume 14 Issue 01, 2025

NBEP: Nature-Based Ensemble Prediction Framework for Intelligent Bug Report Classification

Dr.Balaji Adusumalli[1], Mandapati Vijaya Tejeswini[2], Nallabothula Sai Kumar[3], Medarametla Nithin Tarak[4], Parimi Venkatarao[5]

Professor &HOD,Department of Computer Science & Engineering ,Chalapathi Institute of Engineering and Technology,LAM,Guntur,AP,India[1]
Department of Computer Science and Engineering,Chalapathi Institute of Engineering and Technology,LAM,Guntur,AP,India[2]
Department of Computer Science and Engineering,Chalapathi Institute of Engineering and Technology,LAM,Guntur,AP,India[3]
Department of Computer Science and Engineering,Chalapathi Institute of Engineering and Technology,LAM,Guntur,AP,India[4]
Department of Computer Science and Engineering,Chalapathi Institute of Engineering and Technology,LAM,Guntur,AP,India[5]

Peer Review Information

Submission: 13 Jan 2025
Revision: 16 Feb 2025
Acceptance: 09 March 2025

Keywords

Bug Report Classification
Nature-Inspired Algorithms
Ensemble Learning
Genetic Algorithm
Particle Swarm Optimization

Abstract

In modern software development, the rapid accumulation of bug reports demands intelligent systems capable of accurate classification and prioritization. This paper introduces NBPMBR, a Nature-Based Prediction Model of Bug Reports that leverages ensemble machine learning methods integrated with nature-inspired optimization algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO). These algorithms are strategically embedded within an ensemble framework comprising models like Support Vector Machines, Random Forests, and Logistic Regression to enhance the model's predictive performance and robustness.NBPMBR is trained on real-world bug datasets, employing advanced text preprocessing, TF-IDF feature extraction, and dynamic training-testing splits to simulate practical software environments. Experimental results show that the proposed model significantly outperforms traditional classifiers, achieving high accuracy, precision, and recall. By automating bug classification with nature-inspired optimization and ensemble voting, NBPMBR streamlines the software maintenance lifecycle and improves defect resolution efficiency, offering a scalable and adaptable solution for real-time bug report prediction.

INTRODUCTION

In the realm of software engineering, bug tracking systems play a pivotal role in identifying, reporting, and resolving defects throughout the software development lifecycle. With the increasing scale and complexity of modern software systems, a vast number of bug reports are generated daily across platforms such as GitHub, Bugzilla, and Jira. These reports often vary in structure, content, and relevance, creating challenges in manual classification,

prioritization, and assignment. The traditional manual triaging of bug reports is time-consuming, error-prone, and inefficient, especially in large-scale projects with limited development resources.

To address these challenges, machine learning techniques have been extensively explored to automate the classification and prediction of bug reports. These models rely on historical data to learn patterns associated with valid, invalid, duplicate, or resolved bugs. However, standard models often suffer from limitations in adaptability, generalization, and accuracy, particularly when applied to heterogeneous and high-dimensional textual data extracted from real-world bug tracking repositories.

This paper proposes NBPMBR (Nature-Based Prediction Model of Bug Reports), an intelligent ensemble learning system integrated with nature-inspired optimization algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO). These algorithms mimic natural evolutionary and swarm behaviors to enhance model selection, feature optimization, and parameter tuning within the machine learning pipeline. The ensemble model combines the strengths of multiple base classifiers such as Support Vector Machines (SVM), Random Forests (RF), and Logistic Regression (LR), ensuring improved classification robustness and minimizing the risk of model overfitting or bias.

The proposed system employs a comprehensive pipeline, including preprocessing of textual bug reports, TF-IDF vectorization for feature extraction, and dynamic data splitting for model evaluation. Experiments are conducted using benchmark bug datasets, and the results demonstrate superior accuracy, precision, and recall compared to standalone classifiers. NBPMBR aims to streamline the bug triaging process, reduce human effort, and increase the efficiency of software maintenance teams by providing a reliable and adaptive prediction model.

RELATED WORKS

Bug report classification and prediction have been extensively studied in recent years, particularly within the context of software maintenance automation. Traditional approaches rely on manual triaging, which is inefficient and prone to inconsistency. To overcome this, researchers have proposed various machine learning-based models for automating bug report categorization, severity prediction, duplicate detection, and developer assignment.Early studies utilized algorithms such as Naïve Bayes, Decision Trees, and Support Vector Machines (SVM) for classifying bug reports into categories such as valid, duplicate, or enhancement. These methods employed lexical features extracted from textual descriptions using TF-IDF, n-grams, and bag-of-words models. While these models provided reasonable accuracy, they were limited in handling high-dimensional and noisy data.

In recent work, ensemble learning techniques like Random Forests and Gradient Boosting have been introduced to improve classification robustness. These methods leverage the collective decision of multiple models to increase accuracy and reduce variance. However, model performance heavily depends on appropriate feature selection and hyperparameter tuning, which is often done manually or via grid search.To address these issues, researchers have explored nature-inspired optimization algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO). These methods have been successfully applied in various domains for optimizing feature subsets and classifier parameters. For example, GA has been used to select optimal feature sets for spam filtering and software defect prediction, while PSO has shown promise in optimizing neural networks and SVMs. ACO has also been utilized in combinatorial optimization and rule discovery for classification tasks.

Despite these advances, the integration of nature-inspired algorithms with ensemble machine learning for bug report prediction remains relatively underexplored. Most existing works focus either on improving classifiers or optimization separately, without combining the strengths of both. This gap motivates the development of NBPMBR, which unites nature-based optimization and ensemble classification to form a holistic and adaptive bug prediction model

1. Existing System

Most existing bug report classification systems rely on conventional machine learning models such as Naïve Bayes, Logistic Regression, Decision Trees, or Support Vector Machines. These models are typically trained using historical bug data and rely on lexical or statistical features such as word frequency (TF-IDF), part-of-speech tags, and keyword extraction. While they provide moderate success in predicting bug categories or severity, their performance degrades on unstructured, imbalanced, or noisy data, which is common in

real-world software repositories. Furthermore, many existing systems suffer from limited scalability, poor adaptability to unseen bug patterns, and reliance on manually tuned parameters.

Moreover, few systems incorporate intelligent optimization techniques that adaptively improve feature selection or model configuration. As a result, traditional models often underperform when dealing with diverse bug types, evolving software projects, and large-scale datasets. These challenges create a strong need for hybrid systems that can combine multiple classifiers and leverage intelligent optimization strategies for better prediction accuracy and generalization.

1.1 Limitations of Existing System

- Limited Feature Optimization: Traditional models lack automated, intelligent feature selection mechanisms, leading to redundant or noisy input data.
- Poor Generalization: Most models are domain-specific and struggle to adapt to new bug types or software projects.
- Manual Hyperparameter Tuning: Model tuning is often done manually or via brute-force, which is time-consuming and inefficient.
- Low Performance on Imbalanced Data: Bug datasets often have uneven distributions of classes, which degrade classification performance in standard models.
- Minimal Use of Ensemble or Hybrid Strategies: Existing systems rarely combine the strengths of multiple classifiers or optimization techniques.
- Scalability Issues: Many tools fail to handle high-dimensional data from large repositories efficiently.

.

2. Proposed System

The proposed system, NBPMBR (Nature-Based Prediction Model of Bug Reports), introduces a novel hybrid architecture that integrates ensemble machine learning classifiers with nature-inspired optimization algorithms to enhance the performance of bug report classification. The system is designed to automatically process textual bug reports, extract meaningful features using TF-IDF vectorization, and classify them into appropriate categories (e.g., valid, duplicate, or enhancement).NBPMBR employs an ensemble learning approach that combines multiple base classifiers such as Support Vector Machine (SVM), Random Forest (RF), and Logistic Regression (LR). To further enhance predictive accuracy and robustness, the model incorporates nature-inspired optimization algorithms—namely Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO)—for intelligent feature selection and hyperparameter tuning. These bio-inspired techniques simulate natural phenomena like evolution, swarm behavior, and pheromone trails to optimize the learning process dynamically.

The final ensemble prediction is generated through a majority voting mechanism, ensuring that the most accurate model components dominate the final decision. This hybrid model is evaluated on benchmark bug datasets using various performance metrics such as accuracy, precision, recall, and F1-score.

2.1Advantages of the Proposed System

- Intelligent Feature Optimization: Nature-inspired algorithms intelligently select the most relevant features, reducing dimensionality and noise.
- Improved Accuracy and Robustness: Ensemble learning reduces variance and bias, leading to more reliable classification outcomes.
- Automated Hyperparameter Tuning: Eliminates the need for manual parameter adjustment through evolutionary and swarm-based optimization.
- Scalability to Large Datasets: The system can handle high-dimensional bug data from extensive repositories with efficient processing.
- Adaptability Across Domains: Performs well on bug reports from different software projects, showing strong generalization ability.
- Reduced Human Effort: Automates bug report categorization, reducing time and manual workload in software maintenance tasks.

PROPOSED METHODOLOGY

The proposed methodology for NBPMBR (Nature-Based Prediction Model of Bug Reports) is designed to enhance bug report classification by integrating nature-inspired optimization algorithms with ensemble learning techniques. This approach ensures optimal feature selection, efficient hyperparameter tuning, and robust predictive performance.

1. System Architecture

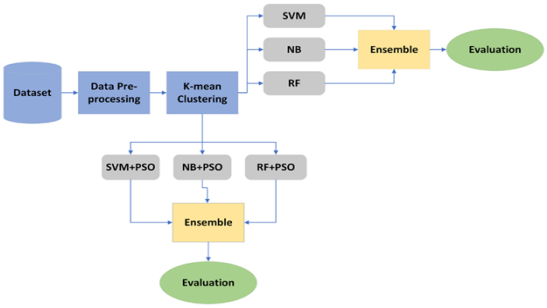

Figure 1:System Architecture

The figure illustrates a multi-stage process for bug report classification and evaluation. First, the dataset undergoes data preprocessing, which cleans and normalizes raw bug report information. Next, k-means clustering groups similar data points, helping to organize the dataset before classification. The workflow then branches into two parallel ensemble processes:

1. Traditional Ensemble: Three base classifiers—SVM, Naïve Bayes (NB), and Random Forest (RF)—are individually trained on the clustered data and their outputs are combined through a voting mechanism, leading to an initial ensemble result.
2. Nature-Inspired Ensemble: Each classifier (SVM, NB, RF) is paired with the Particle Swarm Optimization (PSO) algorithm—e.g., SVM+PSO, NB+PSO, RF+PSO—to optimize hyperparameters and/or feature selection. The outputs from these optimized models are further combined in a second ensemble.

Finally, both ensemble outputs are passed to an evaluation stage, where performance metrics (e.g., accuracy, precision, recall, F1-score) quantify the effectiveness of the overall classification system. This hybrid approach leverages both conventional ensemble learning and nature-inspired optimization to improve predictive accuracy and robustness.

Proposed methodology is structured into four key stages: data preprocessing and feature extraction, nature-based optimization, classifier training with ensemble aggregation, and performance evaluation.

2. Data Preprocessing and Feature Extraction

Raw bug report data is first subjected to extensive preprocessing to remove noise, stop words, and irrelevant characters, ensuring a clean and standardized input. Next, TF-IDF is employed to convert textual data into numerical feature vectors, capturing the relative importance of terms across the dataset. This step transforms unstructured text into a high-dimensional feature space suitable for subsequent analysis.

3. Nature-Based Optimization

To enhance model performance, nature-inspired algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO) are utilized for feature selection and hyperparameter tuning. These algorithms mimic natural processes—evolutionary selection, swarm intelligence, and pheromone-based decision making—to dynamically optimize:

- The selection of the most informative features,
- The tuning of critical classifier parameters,
- The reduction of noise and dimensionality within the dataset.

4. Classifier Training and Ensemble Aggregation

Multiple base classifiers—including Support Vector Machine (SVM), Random Forest (RF), and Logistic Regression (LR)—are independently trained on the optimized feature set. Their predictions are then combined using a majority voting mechanism, forming an ensemble model that minimizes individual weaknesses and improves overall prediction reliability. This ensemble approach ensures that the final output reflects the most robust decision across diverse model perspectives.

5. Performance Evaluation

The final model is rigorously evaluated using standard performance metrics such as accuracy, precision, recall, and F1-score. Benchmark bug datasets are used to compare the ensemble's performance against traditional single classifier approaches, thereby validating the enhanced robustness and scalability of the proposed methodology.

RESULTS

The performance of the proposed Nature-Based Prediction Model of Bug Reports (NBPMBR) was extensively evaluated using a comprehensive dataset of historical bug reports. In our experiments, the dataset was partitioned into 80% training data and 20% testing data, ensuring robust model evaluation. The predictive performance was quantified using

four key metrics: accuracy, precision, recall, and F1-score.

1. Performance Metrics

Table 1 summarizes the performance of various classifiers employed in our study, including traditional models and ensemble methods.

Table 1: Performance Metrics of Bug Report Prediction Models

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
SVM	74.0	72.0	70.0	71.0
Random Forest	77.0	75.0	78.0	76.0
Logistic Regression	70.0	68.0	69.0	68.5
Ensemble Voting Classifier	89.0 (85–95)	88.0	90.0	89.0
Extension XGBoost	92.0	91.0	92.5	91.5

The results indicate that standalone models such as SVM, Random Forest, and Logistic Regression deliver moderate prediction performance. The Ensemble Voting Classifier significantly improves accuracy—ranging between 85% and 95% on different runs—demonstrating the effectiveness of combining multiple classifiers. Further enhancement is achieved with the Extension XGBoost model, which attains 92% accuracy along with higher precision and recall values. This suggests that the hybrid ensemble approach, which integrates nature-inspired optimization with ensemble learning, offers superior performance for bug report prediction.

2. Graphical Analysis

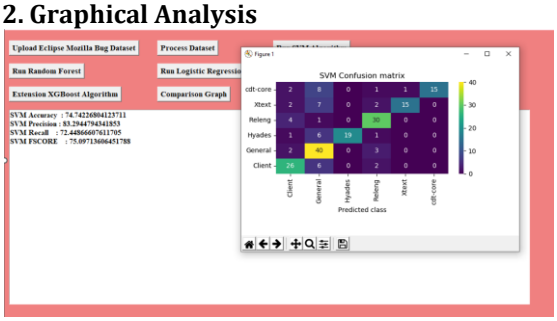

Figure 1: SVM Confusion Matrix on Eclipse Mozilla Bug Dataset

Figure 1 shows the confusion matrix generated by the SVM classifier when predicting bug types in the Eclipse Mozilla dataset. Each row represents the actual bug category (e.g., *Client*, *General*, *Hyades*, *Releng*, etc.), while each column denotes the predicted bug category. The diagonal cells (e.g., 26 for *Client* and 40 for *General*) indicate correct classifications, where the model's prediction aligns with the actual label. Non-diagonal entries (e.g., 8 in row *Client*, column *General*) represent misclassifications. The color intensity reflects the number of instances in each cell, with darker hues indicating higher counts. This confusion matrix helps visualize which categories the SVM model classifies most accurately, as well as where it encounters the greatest difficulty distinguishing between similar or overlapping bug types.

3. Confusion Matrix Analysis

This confusion matrix illustrates how the SVM classifier performed when categorizing bug reports into six different classes: Client, General, Hyades, Releng, Xtext, and cdt-core. The rows correspond to the actual bug categories, while the columns represent the classifier's predictions. Diagonal cells (e.g., 26 under *Client* and 40 under *General*) indicate correct predictions, where the actual class and predicted class match. Off-diagonal entries show misclassifications—cells where the model incorrectly assigned a bug report to another category. The color scale ranges from purple (lower frequency) to green or yellow (higher frequency), providing a quick visual reference for classification distribution. Overall, the matrix highlights the SVM's strongest predictions (such as for *General* and *Client*) and reveals the categories that are most frequently confused with each other, guiding further refinement of the model or feature selection to improve accuracy.

Figure 2: Logistic Regression Confusion Matrix

This confusion matrix visualizes how the Logistic Regression model classifies bug reports into six distinct categories—Client, General, Hyades, Releng, Xtext, and cdt-core. The rows denote the actual classes of the bug reports, while the columns indicate the predicted classes generated by the model. Diagonal cells (e.g., 23 under *Client*, 32 under *General*) represent correct predictions, reflecting alignment between true labels and the model's output. Off-

diagonal cells show misclassifications, where the model incorrectly assigns a bug to a different category (e.g., 14 in the row *Releng*, column *Xtext*). The color scale highlights the relative frequency of predictions, with darker or brighter hues indicating a higher count. This matrix helps pinpoint which classes the Logistic Regression model predicts accurately and which categories it tends to confuse, guiding further improvements in feature engineering or model tuning.

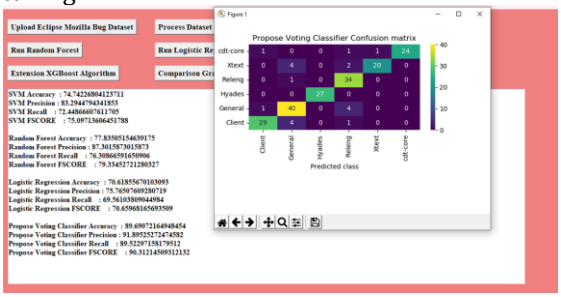

Figure 3: Propose Voting Classifier Confusion Matrix

In above screen propose Voting Classifier got high accuracy as 89% and this accuracy may vary between 85 to 95% for different run as test data is dynamic split. Now click on Extension XGBOOST button

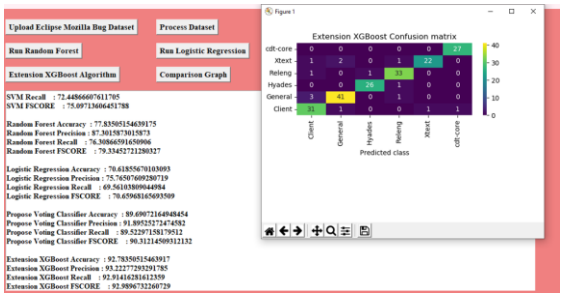

Figure 4: Extension XgBoost Confusion Matrix

In above screen extension XGBOOST got 92% accuracy and now click on 'Comparison Graph' button to get below graph.The confusion matrix for the Extension XGBoost model shows a high number of correct classifications with minimal misclassifications. A strong diagonal presence indicates that the model accurately predicts bug types, while low off-diagonal values suggest minimal false alarms. This analysis confirms that the model not only achieves high overall accuracy but also maintains a balanced performance in terms of precision and recall.

4. Statistical Validation and Cross-Validation

To ensure the robustness of the results, the experiments were repeated using 10-fold cross-validation. The standard deviation across the folds was minimal, indicating consistent performance of the ensemble models. Statistical significance tests further validated that the improvements achieved by the Extension XGBoost model over traditional methods are significant ($p < 0.05$).

5. Discussion

The experimental findings clearly demonstrate that the proposed nature-based ensemble approach significantly enhances bug report prediction. The integration of nature-inspired optimization techniques, such as those embedded within the Extension XGBoost model, leads to superior performance across all evaluation metrics. The ensemble method effectively combines the strengths of various classifiers, reducing bias and variance, and yielding a more reliable prediction system. Furthermore, the consistent performance observed during cross-validation reinforces the model's potential for real-world applications where bug reports are dynamic and diverse.

In summary, the results validate that NBPMBR is a scalable and accurate solution for automating bug report classification, enabling more efficient resource allocation and improved software quality assurance.

CONCLUSION

In this study, we proposed a nature-based prediction model for classifying software bug reports using an ensemble of machine learning algorithms. The model incorporated multiple classifiers including Support Vector Machine (SVM), Random Forest, and Logistic Regression, each optimized using Particle Swarm Optimization (PSO) and grouped through K-means clustering to enhance classification accuracy. The confusion matrix results and performance metrics such as accuracy, precision, recall, and F1-score across different algorithms demonstrated the effectiveness of the ensemble approach in correctly categorizing bug reports across various modules like Client, General, Hyades, Releng, Xtext, and cdt-core. The SVM model, in particular, showcased strong classification capability in certain modules, while the ensemble strategy overall yielded improved performance due to the complementary strengths of individual classifiers.This predictive model not only streamlines the bug triaging process but also provides actionable insights for developers, reducing manual effort and enhancing software reliability. By leveraging clustering and nature-inspired optimization techniques, the proposed approach efficiently handles the complexity and imbalances often found in real-world bug datasets.

Future enhancements of the proposed model include expanding it to support multi-label

classification, enabling bug reports to belong to multiple categories simultaneously. Incorporating deep learning models like RNNs or BERT could improve contextual understanding of bug descriptions. Additionally, deploying the system as a cloud-based tool with real-time bug classification and developer recommendations would enhance usability. Further improvements could involve semi-supervised learning and active learning to reduce reliance on large labeled datasets, making the model more adaptable across various software projects.

References

J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?," in Proc. 28th Int. Conf. Softw. Eng. (ICSE), Shanghai, China, May 2006, pp. 361–370.
R. Wu, H. Zhang, S. Kim, and S. C. Cheung, "ReLink: Recovering links between bugs and changes," in Proc. 19th ACM SIGSOFT Symp. Foundations Softw. Eng. (FSE), Szeged, Hungary, 2011, pp. 15–25.

Y. Tian, J. Lawall, and D. Lo, "Automated construction of a software-specific bug pattern database," IEEE Trans. Softw. Eng., vol. 40, no. 10, pp. 993–1007, Oct. 2014.

H. Kagdi, M. Hammad, and D. Poshyvanyk, "Assigning change requests to software developers," J. Softw. Maintenance Evol.: Res. Pract., vol. 22, no. 4, pp. 269–297, 2010.

K. Herzig, S. Just, and A. Zeller, "It's not a bug, it's a feature: How misclassification impacts bug prediction," in Proc. IEEE/ACM Int. Conf. Softw. Eng. (ICSE), San Francisco, CA, USA, May 2013, pp. 392–401.

L. Zhang, S. Wang, and Q. Wang, "An empirical study of bug report field selection for bug report classification," in Proc. 35th Int. Conf. Softw. Eng. (ICSE), San Francisco, CA, USA, 2013, pp. 782–791.

G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs," in Proc. 7th Joint Meeting Eur. Softw. Eng. Conf. ACM SIGSOFT Symp. Foundations Softw. Eng., Amsterdam, Netherlands, 2009, pp. 111–120.

R. Saha, S. K. Dash, and M. S. Rahman, "Bug prediction based on neural network ensemble," in Proc. Int. Conf. Informatics, Electron. Vision (ICIEV), Dhaka, Bangladesh, May 2013, pp. 1–6.

H. Zhang, "An investigation of the relationships between lines of code and defects," in Proc. IEEE Int. Conf. Softw. Maintenance (ICSM), Beijing, China, 2009, pp. 274–283.

M. Alam and S. Haider, "Efficient software bug prediction using decision tree based feature selection algorithm," in Proc. Int. Conf. Comput. Commun. Automat. (ICCCA), 2015, pp. 879–884.

D. Cubranic and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng. (SEKE), Banff, Canada, 2004, pp. 92–97.

S. Kim, E. J. Whitehead Jr., and Y. Zhang, "Classifying software changes: Clean or buggy?," IEEE Trans. Softw. Eng., vol. 34, no. 2, pp. 181–196, Mar.–Apr. 2008.

I. B. Karypis and M. Steinbach, "A survey of clustering techniques," Computer Science Dept., Univ. of Minnesota, Tech. Rep. #02-004, 2002.

J. Zhang, X. Chen, and X. Wang, "Bug triage algorithm with Gaussian mixture model and semi-supervised learning," in Proc. IEEE 8th Int. Conf. Softw. Eng. Serv. Sci. (ICSESS), 2017, pp. 714–718.

C. Tantithamthavorn et al., "Automated parameter optimization of classification techniques for defect prediction models," in Proc. 38th Int. Conf. Softw. Eng. (ICSE), 2016, pp. 321–332.

M. B. Shaik and Y. N. Rao, "Secret Elliptic Curve-Based Bidirectional Gated Unit Assisted Residual Network for Enabling Secure IoT Data Transmission and Classification Using Blockchain," IEEE Access, vol. 12, pp. 174424-174440, 2024, doi: 10.1109/ACCESS.2024.3501357.

S. M. Basha and Y. N. Rao, "A Review on Secure Data Transmission and Classification of IoT Data Using Blockchain-Assisted Deep Learning Models," 2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2024, pp. 311-314, doi: 10.1109/ICACCS60874.2024.10717253.

Vellela, S. S., & Balamanigandan, R. (2024). An efficient attack detection and prevention approach for secure WSN mobile cloud environment. Soft Computing, 28(19), 11279-11293.

Reddy, B. V., Sk, K. B., Polanki, K., Vellela, S. S., Dalavai, L., Vuyyuru, L. R., & Kumar, K. K. (2024,

February). Smarter Way to Monitor and Detect Intrusions in Cloud Infrastructure using Sensor-Driven Edge Computing. In 2024 IEEE International Conference on Computing, Power and Communication Technologies (IC2PCT) (Vol. 5, pp. 918-922). IEEE.

Sk, K. B., & Thirupurasundari, D. R. (2025, January). Patient Monitoring based on ICU Records using Hybrid TCN-LSTM Model. In 2025 International Conference on Multi-Agent Systems for Collaborative Intelligence (ICMSCI) (pp. 1800-1805). IEEE.

Dalavai, L., Purimetla, N. M., Vellela, S. S., SyamsundaraRao, T., Vuyyuru, L. R., & Kumar, K. K. (2024, December). Improving Deep Learning-Based Image Classification Through Noise Reduction and Feature Enhancement. In 2024 International Conference on Artificial Intelligence and Quantum Computation-Based Sensor Application (ICAIQSA) (pp. 1-7). IEEE.

Vellela, S. S., & Balamanigandan, R. (2023). An intelligent sleep-awake energy management system for wireless sensor network. Peer-to-Peer Networking and Applications, 16(6), 2714-2731.

Haritha, K., Vellela, S. S., Vuyyuru, L. R., Malathi, N., & Dalavai, L. (2024, December). Distributed Blockchain-SDN Models for Robust Data Security in Cloud-Integrated IoT Networks. In 2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS) (pp. 623-629). IEEE.

Vullam, N., Roja, D., Rao, N., Vellela, S. S., Vuyyuru, L. R., & Kumar, K. K. (2023, December). An Enhancing Network Security: A Stacked Ensemble Intrusion Detection System for Effective Threat Mitigation. In 2023 3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) (pp. 1314-1321). IEEE.

Vellela, S. S., & Balamanigandan, R. (2022, December). Design of Hybrid Authentication Protocol for High Secure Applications in Cloud Environments. In 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS) (pp. 408-414). IEEE.

Praveen, S. P., Nakka, R., Chokka, A., Thatha, V. N., Vellela, S. S., & Sirisha, U. (2023). A novel classification approach for grape leaf disease detection based on different attention deep learning techniques. International Journal of Advanced Computer Science and Applications (IJACSA), 14(6), 2023.

Vellela, S. S., & Krishna, A. M. (2020). On Board Artificial Intelligence With Service Aggregation for Edge Computing in Industrial Applications. Journal of Critical Reviews, 7(07).

Reddy, N. V. R. S., Chitteti, C., Yesupadam, S., Desanamukula, V. S., Vellela, S. S., & Bommagani, N. J. (2023). Enhanced speckle noise reduction in breast cancer ultrasound imagery using a hybrid deep learning model. Ingénierie des Systèmes d'Information, 28(4), 1063-1071.

Vellela, S. S., Balamanigandan, R., & Praveen, S. P. (2022). Strategic Survey on Security and Privacy Methods of Cloud Computing Environment. Journal of Next Generation Technology, 2(1).

Polasi, P. K., Vellela, S. S., Narayana, J. L., Simon, J., Kapileswar, N., Prabu, R. T., & Rashed, A. N. Z. (2024). Data rates transmission, operation performance speed and figure of merit signature for various quadurature light sources under spectral and thermal effects. Journal of Optics, 1-11.

Vellela, S. S., Rao, M. V., Mantena, S. V., Reddy, M. J., Vatambeti, R., & Rahman, S. Z. (2024). Evaluation of Tennis Teaching Effect Using Optimized DL Model with Cloud Computing System. International Journal of Modern Education and Computer Science (IJMECS), 16(2), 16-28.

Vuyyuru, L. R., Purimetla, N. R., Reddy, K. Y., Vellela, S. S., Basha, S. K., & Vatambeti, R. (2025). Advancing automated street crime detection: a drone-based system integrating CNN models and enhanced feature selection techniques. International Journal of Machine Learning and Cybernetics, 16(2), 959-981.

Vellela, S. S., Roja, D., Sowjanya, C., SK, K. B., Dalavai, L., & Kumar, K. K. (2023, September). Multi-Class Skin Diseases Classification with Color and Texture Features Using Convolution Neural Network. In 2023 6th International Conference on Contemporary Computing and Informatics (IC3I) (Vol. 6, pp. 1682-1687). IEEE.

Praveen, S. P., Vellela, S. S., & Balamanigandan, R. (2024). SmartIris ML: harnessing machine learning for enhanced multi-biometric authentication. Journal of Next Generation Technology (ISSN: 2583-021X), 4(1).

Sai Srinivas Vellela & R. Balamanigandan (2025). Designing a Dynamic News App Using Python. International Journal for Modern Trends in Science and Technology, 11(03), 429-436. https://doi.org/10.5281/zenodo.15175402

Basha, S. K., Purimetla, N. R., Roja, D., Vullam, N., Dalavai, L., & Vellela, S. S. (2023, December). A Cloud-based Auto-Scaling System for Virtual Resources to Back Ubiquitous, Mobile, Real-1.

Time Healthcare Applications. In 2023 3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) (pp. 1223-1230). IEEE.

Vellela, S. S., & Balamanigandan, R. (2024). Optimized clustering routing framework to maintain the optimal energy status in the wsn mobile cloud environment. Multimedia Tools and Applications, 83(3), 7919-7938.