An Ensemble Learning Framework with Metaheuristic Optimization for Credit Card Fraud Detection

Agung Nugroho, Muhtajuddin Danny, Ismasari Nawangsih
Dept. of Informatics Engineering, Universitas Pelita Bangsa, Bekasi, Indonesia

***Abstract*—Credit card fraud detection is a major challenge in the financial system due to the characteristics of highly unbalanced data. This study proposes an ensemble learning approach combined with hyperparameter optimization using a Genetic Algorithm to improve the performance of fraud transaction detection. The results of the experiment showed that Random Forest achieved the best performance with a perfect Recall of 1.00 and an F1-Score of 0.903, outperforming the Stacking and Bagging models. Although the optimization significantly increases the training time, this method manages to accelerate the inference time to 0.0290 seconds, making it very feasible to apply to real-time banking security systems that require instant validation. This study confirms the effectiveness of integrating ensemble learning and metaheuristic optimization in dealing with the problem of unbalanced data.**

***Keywords*—Fraud detection; ensemble learning; Genetic Algorithm; Random Forest; real-time detection**

I. Introduction

The rapid increase in digital transactions has changed the global economic landscape, but at the same time, widened the gap for cybercriminal activities. Recent reports show that financial losses due to credit card fraud continue to increase as criminals adapt to conventional security protocols [1][2]. Conventional detection is difficult to rely on anymore because criminals always change their action patterns so that they are unreadable and appear like transactions in general [3]. Accurate early detection becomes crucial, as system failures not only impact direct monetary losses but also erode consumer confidence in the digital banking ecosystem as a whole [4].

The main challenge in detecting credit card fraud lies in the characteristics of extreme class imbalance, where the number of legitimate transactions far exceeds the number of fraudulent transactions with a ratio that can reach 1:1000 or more [5]. In addition, fraud behavior is dynamic (concept drift), where attack patterns are constantly changing to avoid rule-based detection. Traditional machine learning models often fail to address this imbalance, tending to result in high false negative rates that are particularly risky for financial institutions [6].

In recent years, researchers have explored the use of single machine learning models and deep learning to address this problem. However, recent studies show that single models such as Support Vector Machine, Logistic Regression, or Decision Tree often overfit or get stuck in local optima when dealing with high-dimensional data [7] and tend to produce high false negatives when faced with class imbalances between normal and fraudulent transactions [5]. Data imbalance is a major challenge because fraudulent transactions amount to less than 1% of the total data [8].

To overcome the limitations of a single model, the ensemble learning approach has become a prime choice due to its ability to combine the results of multiple algorithms to obtain more stable and accurate classification performance [9]. Approaches such as soft voting and ensemble stacking have been shown to improve detection accuracy and lower false positive rates compared to individual models [10]. The combination of several algorithms, such as Random Forest, AdaBoost, and XGBoost, also provides superior results in handling highly variable transaction data [11].

Nonetheless, the performance of the ensemble model relies heavily on the selection of precise hyperparameters and the selection of relevant features. Searching for parameters manually or using conventional grid search methods has proven to be inefficient and often fails to find the optimal configuration within a large search space [12]. Therefore, metaheuristic optimization algorithms such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Butterfly Optimization Algorithm (BOA) are used to automate the process of finding parameters and selecting the most relevant features [13]. This approach not only improves efficiency but also strengthens the model's ability to deal with high-dimensional datasets [14].

In addition, the integration of data balancing techniques such as SMOTE-ENN and K-means resampling has been proven to improve ensemble learning capabilities in dealing with extreme data imbalances [8]. This approach ensures the model not only focuses on the majority class, but is also able to recognize fraudulent transaction patterns with high accuracy [15]. The combination of metaheuristic optimization and ensemble learning results in models that are adaptive to data dynamics and changes in fraud patterns [16].

This study proposes a new framework that integrates Ensemble Learning with Metaheuristic optimization to detect credit card fraud more precisely. Most previous research has focused on improving accuracy without considering computational efficiency or generalizations on dynamic data [17]. Therefore, this study aims to develop an Ensemble Learning Framework with Metaheuristic Optimization that is able to balance detection performance, efficiency, and adaptability to changing transaction patterns.

II. METHODOLOGY

The methodology proposed in this study is systematically designed to build a robust fraud detection model, address the challenge of data imbalance through the SMOTE technique, and optimize performance through an ensemble stacking approach integrated with metaheuristic optimization. The overall workflow, as visualized in the research flowchart in Fig. 1, consists of five main stages: data collection, preprocessing, unbalanced data handling, model development, and matrix evaluation.

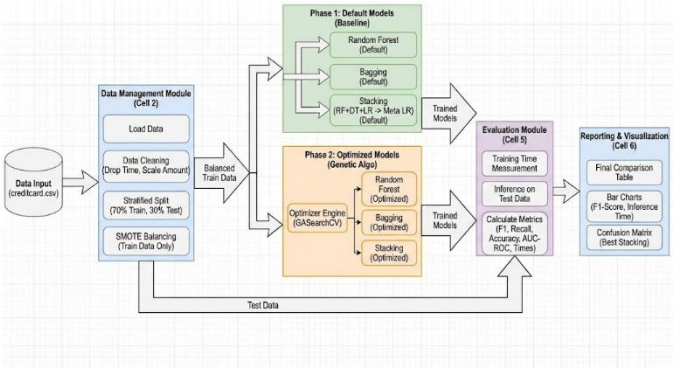

Fig. 1. Research flow.

A. Dataset

The initial stage begins with data input, where the target dataset, i.e., credit card transaction data, is loaded into the analysis environment. This dataset contains credit card transactions that will be the basis for learning for machine learning models. This dataset is sourced from Kaggle, which collected credit card transaction data from cardholders in Europe over two days in September 2013. The data covers a total of 284,807 transactions, of which only 492 transactions were identified as fraud [18]. The characteristics of this dataset are categorized as highly unbalanced due to the significant disparity in proportion between legitimate transactions and fake transactions. This class imbalance is a crucial challenge that is deeply considered in the development of this machine learning model. This dataset was chosen because it has been widely used as a benchmark in many studies on credit card fraud detection, allowing the results of this study to be objectively compared with previous studies [2][19].

B. Data Preprocessing

The preprocessing stage aims to ensure the quality and consistency of data before it is used for model training. First, data cleaning is carried out by removing irrelevant features such as time columns (Time) that do not contribute to fraud patterns, and performing feature scaling using the StandardScaler method to normalize numerical variables such as Amount. After that, the dataset was divided using a stratified split technique with a proportion of 70% training data and 30% test data, to ensure that the distribution of fraud and non-fraud classes remained balanced in each subset [20]. This process is important so that the model does not experience bias against the majority class during training or validation [21].

C. Handling Imbalance Data

One of the most critical challenges in fraud detection is the highly unbalanced nature of data (class imbalance), where the number of fraudulent transactions is much smaller than normal. If left unchecked, the model tends to be biased towards the majority class and fails to detect fraud. To overcome this, the Synthetic Minority Over-sampling Technique (SMOTE) is applied exclusively to the training data only. SMOTE works not by simply duplicating minority data, but by creating new synthetic samples based on interpolation between adjacent minority samples [22]. This approach results in a Balanced Train Data that allows the model to learn the characteristics of fraud more effectively without causing data leakage to the test data. This method helps to expand the search space for fraud patterns without copying data repeatedly [23].

SMOTE helps balance minority class data by synthesizing new samples, but can trigger overfitting if not implemented correctly. In order for the model to have good generalization capabilities, SMOTE is carried out through several stages: calculating the spacing of minority data, determining the percentage of SMOTE and k-value, and finally creating artificial data [24]. Details of this mechanism can be seen in Eq. (1):

$$x_{syn} = x_i + (x_{knn} - x_i) \times \delta \quad (1)$$

where, x_{syn} is the set of synthetic data to be generated, x_i is the data to be replicated, x_{knn} is the data and value closest to the data to be replicated, and δ is a random value between 0 and 1.

D. Model Development

The model development stage utilizes training data that has been balanced and carried out in two experimental phases using the Ensemble Learning approach. The ensemble method is known to improve generalization and robustness of predictions by combining several base models [25].

Phase 1: Default Models (Baseline). This phase aims to build performance benchmarks. The three ensemble architectures of Random Forest, Bagging, and Stacking are trained using the default hyperparameters. Specifically for the Stacking model, the architecture built consists of Random Forest, Decision Tree, and Logistic Regression as base-learners, with additional Logistic Regression serving as meta-learners to aggregate predictions. This approach was chosen because of its ability to combine the advantages of heterogeneous models to produce more stable and accurate predictions than single models [11].

Random Forest – can reduce overfitting by combining multiple decision trees [26][27]. Random Forest is configured with n_estimators=100, random_state=42, and n_jobs=-1. This configuration ensures reproducible results, efficient training, and stable performance in fraud detection.

Bagging Classifier – used to reduce model variance by using bootstrap sampling on the training data. This makes the model more stable and prevents overfitting [28].

Stacking Classifier – a combination of predictions from several base algorithms (base learners) through meta-learners. The purpose of this method is to take advantage of each algorithm so that its performance is more optimal (see Algorithm 1).

Algorithm 1: Stacking Model

Input: Training data after SMOTE balancing
Output: Trained stacking model
1: **Begin**
2: Initialize base learners $B=\{h_1, h_1, \ldots, h_3\}$
3: Train base learners: each h_k is trained on D'_{train} to generate predictions $\hat{y}_k$
4: Construct meta-features: for each sample x_i
$$M(x_i) = [h_1(x_i),\ h_2(x_i), \ldots, h_k(x_i)]$$
5: Train the meta-learner g using the pairs $(M(x_i), y_i)$
6: Final prediction: for a new sample x_j, the stacking prediction is $S(x_j) = g([h_1(x_j),\ h_2(x_j), \ldots, h_k(x_j)])$
7: **End**

Phase 2: Optimized Models (Genetic Algorithm). This phase focuses on improving the performance of the baseline model through hyperparameter optimization. The Optimizer Engine used is based on Genetic Algorithm (GA) through GASearchCV. GA mimics the natural selection process to iteratively evolve a population of parameter combinations to find the optimal configuration that maximizes fitness functions (e.g., F1-Score) for Random Forest, Bagging, and Stacking models.

GA works on the principle of natural selection, where the best combination of parameters is obtained through the process of selection, crossover, and mutation. The fitness function value used is 5-fold cross-validation. This algorithm allows the search for global solutions in complex parameter spaces, resulting in models with better generalizations of new data [16]. The combination of ensemble learning with metaheuristic optimization, such as GA, has been proven to be able to reduce false positive rates without sacrificing inference time significantly [29].

TABLE I. GA HYPERPARAMETER SETTINGS

Parameter	Setting
Population × Generations	6 × 3
Selection	Tournament
Crossover / Mutation	0.8 / 0.1
CV Strategy	5-fold stratified
Fitness	F1-score
Seed	42

Table I presents the detailed configuration of the Genetic Algorithm employed for hyperparameter optimization. The GA operated with a population size of six individuals across three generations, yielding a total of 90 model evaluations when accounting for 5-fold cross-validation. Tournament selection was adopted to maintain competitive pressure among candidate solutions, while crossover (0.8) and mutation (0.1) probabilities were set to balance global exploration and local exploitation of the search space. Elitism ensured that the best solution from each generation was preserved, contributing to rapid convergence as observed in the experimental results. Stratified 5-fold cross-validation was used as the fitness evaluation strategy to maintain class imbalance proportions and reduce overfitting risk. A fixed random seed (42) was applied to guarantee experimental reproducibility.

E. Matrix Evaluation

The final stage is a comprehensive evaluation within the Evaluation Module. It is important to emphasize that all trained models (both from Phase 1 and Phase 2) were evaluated using the original Test Data that was not touched by the SMOTE process. This ensures performance measurements reflect the model's ability to deal with unbalanced real-world data. Evaluation not only measures computational efficiency (training time and inference time), but also performance metrics that are crucial for unbalanced data cases. In addition to Accuracy, the analysis relies heavily on the metrics of Precision, Recall, F1-Score (harmonized average of precision and recall), and Area Under the ROC Curve (AUC-ROC) to provide a complete picture of the model's ability to minimize false negatives (fraud that escapes) and false positives (false alarms). The final results are presented in the form of comparison tables, bar graphs, and confusion matrices. This analysis helps identify the trade-offs between accuracy and efficiency in the implementation of a real-time credit card fraud detection system [30]. All equations can be defined in Eq. (2) to Eq. (5):

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} x\ 100 \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$Precission = \frac{TP}{TP+FP} \quad (4)$$

$$F1-score = 2\ \frac{precission * recall}{precission + recall} \quad (5)$$

TP and TN show the amount of positive and negative data that were correctly predicted. Meanwhile, FP and FN described misclassifications in each of these categories. The accuracy metric measures the extent to which target and non-target samples are precisely predicted, reflecting the ability of the test data to represent patterns in the training data [31]. There are two key elements that affect the level of accuracy, namely the amount of data and the imbalance conditions in the dataset used.

III. RESULTS

This study evaluated the performance of ensemble learning, namely Random Forest, Bagging, and Stacking in detecting credit card fraud through comparison of standard conditions and Genetic Algorithm (GA) optimization. Data imbalances were addressed with SMOTE on the training set, while the test data remained in the original distribution to maintain the relevance of reality. The evaluation focused on the Recall and F1-Score metrics in addition to accuracy, precision and AUC-ROC, given the importance of detecting fraudulent transactions on unbalanced data.

The analysis was conducted using public data from Kaggle on European cardholder credit card transactions as of September 2013. The two-day data collection consisted of 284,807 transactions with 492 of them being fraudulent. With a percentage of fraud classes that only reaches 0.172%, this dataset is classified as highly unbalanced. Fig. 2 presents a distribution graph for each of these classes.

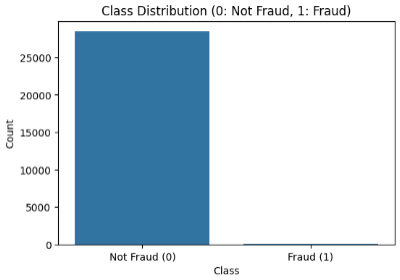

Fig. 2. Class data distribution.

The data is then divided into test data (30%) and training data (70%). In the training data, the SMOTE technique was applied to balance the classes. Initially, out of 199,364 training data, there were only 344 samples of fraud classes. However, after the balancing process, the number of each class (fraudulent and non-fraudulent) became equally large, at 199,020 samples. Details of the comparison before and after SMOTE can be seen in Table II and the graph in Fig. 3.

TABLE II. DATA TRAIN WITH SMOTE

Label (y)	*Row data*	*Row data with SMOTE*
0	199020	199020
1	344	199020

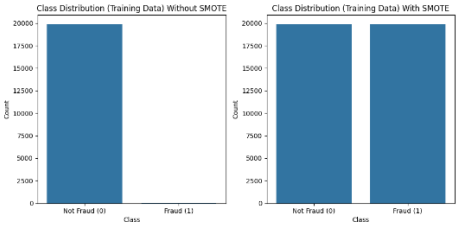

Fig. 3. Comparison data train with SMOTE.

A. Baseline Model Performance

Based on the initial experiment without optimization (baseline), there was a significant difference in performance between the three ensemble algorithms. Referring to Table III, the Stacking model shows the most superior performance with a perfect Precision of 1.00 and an F1-Score of 0.88, which indicates a high ability to minimize false positives. Random Forest (RF) also shows competitive performance with an Accuracy of 0.9995 and a Recall of 0.928, making it a fairly reliable baseline model. In contrast, the Bagging model showed the lowest performance with a Recall of only 0.50 and an F1-Score of 0.48, which indicates its inability to detect half of the total fraud cases that exist.

TABLE III. BASELINE MODEL PERFORMANCE (WITHOUT GENETIC ALGORITHM OPTIMIZATION)

Model	Accuracy	Recall	Precision	F1-Score	AUC-ROC
Random Forest	0.999532	0.928571	0.812500	0.866667	0.999883
Bagging	0.998245	0.500000	0.466667	0.482759	0.783282
Stacking	0.999649	0.785714	1.000000	0.880000	0.999908

B. Optimization of Genetic Algorithms for Model Performance

The application of the Genetic Algorithm aims to find the optimal combination of hyperparameters in each ensemble model. Based on the data in Table IV, the average fitness value for Random Forest increased from 0.9989 in the initial generation to 0.9998 in the 2nd generation with a decreasing standard deviation, indicating that the algorithm successfully converged to the optimal solution. The Random Forest and Stacking models show relatively rapid convergence, with consistent max fitness values since the first generation, while the Bagging model requires a wider variety of parameters before achieving optimal performance.

TABLE IV. RESULTS OF GENETIC ALGORITHM OPTIMIZATION

Model	Generation	Evaluations	Fitness (Mean)	Fitness Std	Fitness Max	Fitness Min
Random Forest (Optimized)	0	6	0.998947	0.00183917	0.999849	0.994836
	1	12	0.999778	0.00005692	0.999849	0.999699
	2	12	0.999841	0.00001184	0.999849	0.999824
Bagging (Optimized)	0	6	0.998746	0.00089185	0.998971	0.998394
	1	12	0.998959	0.00008410	0.999122	0.998846
	2	12	0.999063	0.00008763	0.999197	0.998971
Stacking (Optimized)	0	6	0.999774	0.00000561	0.999824	0.999699
	1	12	0.999820	0.00000936	0.999824	0.999799
	2	12	0.999824	0.00000000	0.999824	0.999824

The application of Genetic Algorithm-based hyperparameter optimization has a significant positive impact, especially on the Random Forest and Stacking models. Based on Table V, Random Forest (Optimized) managed to achieve a perfect Recall of 1.00 (up from 0.928) and an increase in F1-Score to 0.903, making it the best model for detecting all fraudulent transactions without false negatives. The Stacking (Optimized) model also experienced an increase in Recall to 0.857 from the previous 0.785, with a very high AUC-ROC of 0.99995. However, a different phenomenon occurs in the Bagging model, where the optimization actually lowers its performance (the F1-Score drops to 0.387), which indicates that the hyperparameter configuration found by GA may be causing overfitting or not matching the characteristics of the model.

Fig. 4 and Fig. 5 reinforce the numerical results in the comparison table, where Random Forest Optimized shows the most significant F1-Score improvement over other models. The Stacking model also showed consistent improvements, while the Bagging Optimized experienced a decrease in performance. This confirms that the effectiveness of GA optimization is highly dependent on the characteristics of the base model used in the ensemble.

TABLE V. MODEL PERFORMANCE COMPARISON (DEFAULT VS. OPTIMIZED)

Model	Accuracy	Recall	Precision	F1-Score	AUC-ROC
Random Forest	0.999532	0.928571	0.812500	0.866667	0.999883
Random Forest (Optimized)	0.999649	**1.000000**	0.823529	**0.903226**	**0.999908**
Bagging	0.998245	0.500000	0.466667	0.482759	0.783282
Bagging (Optimized)	0.997776	0.428571	0.352941	0.387097	0.783282
Stacking	0.999649	0.785714	**1.000000**	0.880000	0.999908
Stacking (Optimized)	0.999649	0.857143	0.923077	0.888889	**0.999954**

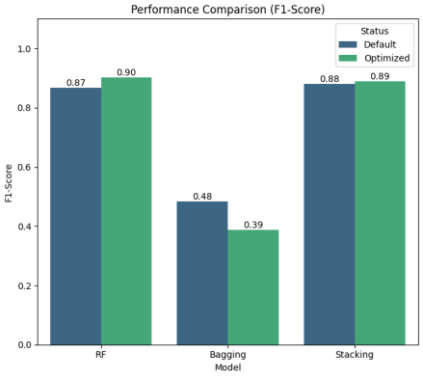

Fig. 4. Performance comparison (F1-score).

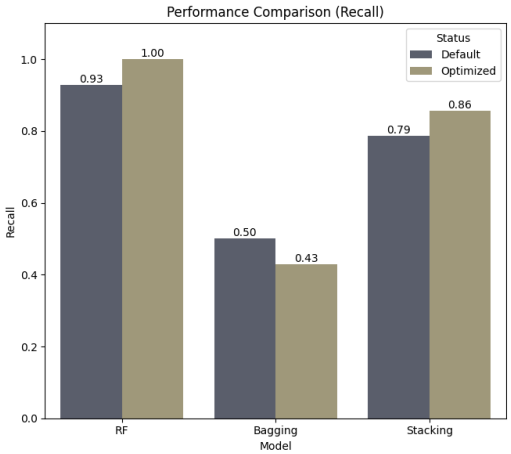

Fig. 5. Performance comparison (Recall).

C. Genetic Algorithm Convergence Analysis

Based on Fig. 6, it can be seen that fitness values increase rapidly in the early generation and then tend to stabilize in the next generation. This pattern shows that the Genetic Algorithm is able to find near-optimal solutions in a relatively small number of generations. The stable convergence also indicates that the GA parameters used have been quite effective in balancing the exploration and exploitation of the solution space.

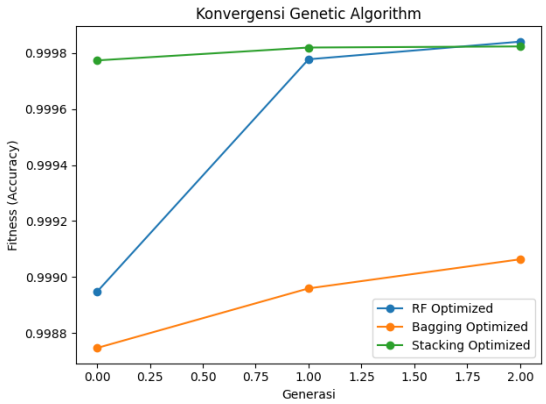

Fig. 6. Genetic Algorithm convergence.

D. Confusion Matrix Analysis and Classification Error Patterns

The confusion matrix analysis in Fig. 7 shows that the optimized Random Forest and Stacking models are able to significantly reduce the number of false negatives, which is a crucial aspect of the fraud detection system. The lack of fraudulent transactions that are misclassified as normal transactions indicates that the optimized model is more sensitive to the minority class. However, there are still a small number of false positives to be aware of, as they can impact the user experience in a real system.

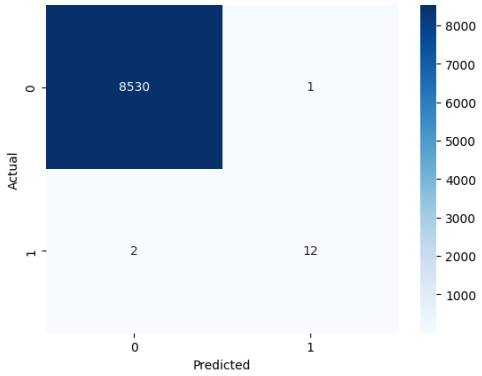

Fig. 7. Confusion matrix.

E. Computational Efficiency and Implementation Feasibility

The results of the computational efficiency evaluation in Table VI and Fig. 8 show that the optimization of the Genetic Algorithm significantly increases training time, especially in Random Forest and Bagging.

TABLE VI. TRAINING TIME AND MODEL INFERENCE

Model	Training Time (s)	Inference Time (s)
Random Forest	21.84	0.0428
Random Forest (Optimized)	434.27	0.0290
Bagging	14.88	**0.0105**
Bagging (Optimized)	1466.00	**0.0104**
Stacking	76.20	0.0691
Stacking (Optimized)	**2288.88**	0.0407

The compute experiment was run on the Google Colab platform with support for an NVIDIA A100 GPU, 16 GB of RAM, and a dual-core Intel Xeon virtual CPU. The entire model is built using the Scikit-learn library with a single-batch inference scheme.

The implementation of GA brings a significant trade-off between training time and inference time. Efficiency data shows a massive spike in training time; for example, Random Forest increased from 21.84 seconds (default) to 434.27 seconds after optimization, and Bagging jumped drastically to 1466 seconds. Nonetheless, it's interesting to note that optimization actually speeds up the time of inference or prediction. The Random Forest inference time is reduced from 0.0428 seconds to 0.0290 seconds, and Stacking (assumed from the context of the efficiency table) also shows a similar efficiency pattern during the prediction phase. This shows that although the initial computational costs (training) are very expensive, the resulting model is more compact and faster when used to detect transactions in real-time.

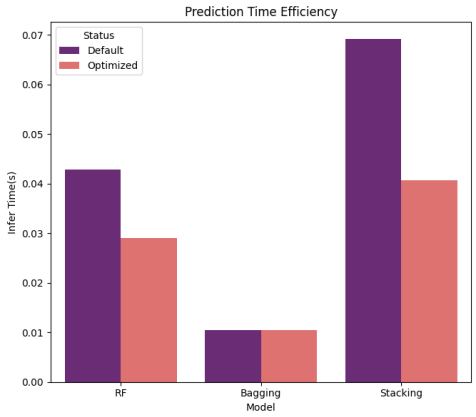

Fig. 8. Inference time.

Overall, the results of the experiment show that the combination of ensemble learning and Genetic Algorithm is effective in improving fraud detection performance, especially in the Random Forest and Stacking models. Improved Recall and F1-Score prove that this approach is capable of overcoming the challenge of data imbalance. However, performance improvements must be considered along with higher computing costs, so the selection of the best model needs to consider the needs of the system and available resources.

This study proves that hyperparameter optimization using a Genetic Algorithm can improve ensemble learning performance in detecting credit card fraud in unbalanced datasets. Random Forest Optimized emerged as the best model with a balance between detection performance and inference efficiency, making it the most viable candidate for practical implementation.

IV. Conclusion

This study concludes that the integration of Ensemble Learning with Genetic Algorithm (GA) optimization is proven to significantly improve fraud detection capabilities, especially in the Random Forest (Optimized) model, which managed to achieve peak performance with a perfect Recall of 1.00 and an F1-Score of 0.903. These results indicate the success of the model in identifying all fraudulent transactions without False Negatives, although the experiment also notes that this metaheuristic optimization has limitations because it is not effective for the entire algorithm, as seen in the performance degradation of the Bag model. From the computational aspect, it was found that there was a strategic trade-off where the use of GA did trigger a spike in training time, but at the same time was able to drastically cut the inference time to only 0.0290 seconds in the Random Forest model. Thus, this combination of maximum detection accuracy and high prediction time efficiency confirms that the model is very feasible and reliable to be implemented into real-time-based banking security systems that demand high speed and precision in the validation of each transaction.

Acknowledgment

This research was carried out with the support of Universitas Pelita Bangsa and the Ministry of Higher Education, Science and Technology in the form of competitive grant funding with contract number 002/7/KP.H/UPB/2025.

References

[1] T. A. Gaav, H. U. Adoga, and T. Moses, "Recent Advances in Credit Card Fraud Detection: An Analytical Review of Frameworks, Methodologies, Datasets, and Challenges," J. Futur. Artif. Intell. Technol., vol. 2, no. 3, pp. 343–369, Sep. 2025, doi: 10.62411/faith.3048-3719-251.

[2] A. R. Khalid, N. Owoh, O. Uthmani, M. Ashawa, J. Osamor, and J. Adejoh, "Enhancing Credit Card Fraud Detection: An Ensemble Machine Learning Approach," Big Data Cogn. Comput., vol. 8, no. 1, p. 6, Jan. 2024, doi: 10.3390/bdcc8010006.

[3] Dipali Sanjay Khedkar and Prof. Rashmi Kulkarni, "Credit Card Fraud Detection using Machine Learning," Int. J. Adv. Res. Sci. Commun. Technol., pp. 1–8, Apr. 2025, doi: 10.48175/IJARSCT-25301.

[4] C.-T. Chen, C. Lee, S.-H. Huang, and W.-C. Peng, "Credit Card Fraud Detection via Intelligent Sampling and Self-supervised Learning," ACM Trans. Intell. Syst. Technol., vol. 15, no. 2, pp. 1–29, Apr. 2024, doi: 10.1145/3641283.

[5] K. K. Mohbey, M. Z. Khan, and A. Indian, "Credit Card Fraud Prediction Using XGBoost," Int. J. Inf. Retr. Res., vol. 12, no. 2, pp. 1–17, Jul. 2022, doi: 10.4018/IJIRR.299940.

[6] S. Sheokand and S. Beniwal, "Hybrid metaheuristic optimization based credit card fraud detection," 2025, p. 050004. doi: 10.1063/5.0299060.

[7] F. K. Alarfaj, I. Malik, H. U. Khan, N. Almusallam, M. Ramzan, and M. Ahmed, "Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms," IEEE Access, vol. 10, pp. 39700–39715, 2022, doi: 10.1109/ACCESS.2022.3166891.

[8] I. D. Mienye and Y. Sun, "A Deep Learning Ensemble With Data Resampling for Credit Card Fraud Detection," IEEE Access, vol. 11, pp. 30628–30638, 2023, doi: 10.1109/ACCESS.2023.3262020.

[9] M. R. Baker, Z. N. Mahmood, and E. H. Shaker, "Ensemble Learning with Supervised Machine Learning Models to Predict Credit Card Fraud Transactions," Rev. d'Intelligence Artif., vol. 36, no. 4, pp. 509–518, Aug. 2022, doi: 10.18280/ria.360401.

[10] A.-A. Al-Maari, M. Abdulnabi, Y. Nathan, A. Ali, U. Ali, and M. Khan, "Optimized Credit Card Fraud Detection Leveraging Ensemble Machine Learning Methods," Eng. Technol. Appl. Sci. Res., vol. 15, no. 3, pp. 22287–22294, Jun. 2025, doi: 10.48084/etasr.10287.

[11] S. Kodati, N. Nallametti, K. Veeranjaneyulu, N. Sreekanth, B. M. Rao, and G. Jagadesh, "An Ensemble Machine Learning Approach for Enhancing Credit Card Fraud Detection," J. Neonatal Surg., vol. 14, no. 4, pp. 32–40, Mar. 2025, doi: 10.52783/jns.v14.2436.

[12] S. S. Reddy, K. Amrutha, V. MNSSVKR Gupta, K. VSSR Murthy, and V. V. R. M. Rao, "Optimizing Hyperparameters for Credit Card Fraud Detection with Nature-Inspired Metaheuristic Algorithms in Machine

Learning," J. Inst. Eng. Ser. B, vol. 106, no. 6, pp. 2005–2030, Dec. 2025, doi: 10.1007/s40031-025-01207-2.

[13] P. Sure, S. Pandey, T. Parnami, Gaurang, and A. Saxena, "Feature Selection Techniques for Enhancing Credit Card Fraud Detection Performance: A Hybrid Metaheuristic Approach Using Nature-Inspired Algorithms," Int. J. Res. Appl. Sci. Eng. Technol., vol. 12, no. 2, pp. 1185–1194, Feb. 2024, doi: 10.22214/ijraset.2024.58549.

[14] Y. Wang, "A Data Balancing and Ensemble Learning Approach for Credit Card Fraud Detection," in 2025 4th International Symposium on Computer Applications and Information Technology (ISCAIT), IEEE, Mar. 2025, pp. 386–390. doi: 10.1109/ISCAIT64916.2025.11010591.

[15] D. H. M. de Souza and C. J. Bordin, "Ensemble and Mixed Learning Techniques for Credit Card Fraud Detection," pp. 1–7, 2021, [Online]. Available: http://arxiv.org/abs/2112.02627

[16] E. Ileberi and Y. Sun, "A Hybrid Deep Learning Ensemble Model for Credit Card Fraud Detection," IEEE Access, vol. 12, pp. 175829–175838, 2024, doi: 10.1109/ACCESS.2024.3502542.

[17] T. M. Museba and K. V. Vanhoof, "An Adaptive Heterogeneous Ensemble Learning Model for Credit Card Fraud Detection," Adv. Sci. Technol. Eng. Syst. J., vol. 9, no. 3, pp. 1–11, May 2024, doi: 10.25046/aj090301.

[18] "Kaggle." [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

[19] N. Baisholan, J. E. Dietz, S. Gnatyuk, M. Turdalyuly, E. T. Matson, and K. Baisholanova, "A Systematic Review of Machine Learning in Credit Card Fraud Detection Under Original Class Imbalance," Computers, vol. 14, no. 10, p. 437, Oct. 2025, doi: 10.3390/computers14100437.

[20] M. Azim Mim, N. Majadi, and P. Mazumder, "A soft voting ensemble learning approach for credit card fraud detection," Heliyon, vol. 10, no. 3, p. e25466, Feb. 2024, doi: 10.1016/j.heliyon.2024.e25466.

[21] L. Bonde and A. K. Bichanga, "Improving Credit Card Fraud Detection with Ensemble Deep Learning-Based Models: A Hybrid Approach Using SMOTE-ENN," J. Comput. Theor. Appl., vol. 2, no. 3, pp. 383–394, Feb. 2025, doi: 10.62411/jcta.12021.

[22] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.

[23] K. H. Ahmed, S. Axelsson, Y. Li, and A. M. Sagheer, "A credit card fraud detection approach based on ensemble machine learning classifier with hybrid data sampling," Mach. Learn. with Appl., vol. 20, p. 100675, Jun. 2025, doi: 10.1016/j.mlwa.2025.100675.

[24] Asniar, N. U. Maulidevi, and K. Surendro, "SMOTE-LOF for noise identification in imbalanced data classification," J. King Saud Univ. - Comput. Inf. Sci., vol. 34, no. 6, pp. 3413–3423, 2022, doi: 10.1016/j.jksuci.2021.01.014.

[25] A. Mohammed and R. Kora, "A comprehensive review on ensemble deep learning: Opportunities and challenges," J. King Saud Univ. Comput. Inf. Sci., vol. 35, pp. 757–774, 2023, doi: 10.1016/j.jksuci.2023.01.014.

[26] R. Sharma and D. Minhas, "Comparative Performance of Random Forest versus Gradient Boosting Machines in Detecting Financial Fraud," in 2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT), IEEE, Aug. 2024, pp. 1027–1030. doi: 10.1109/ICCPCT61902.2024.10673140.

[27] S. S. Suganya, S. Nishanth, and D. Mohanadevi, "Ensemble Learning Approaches for Fraud Detection in Financial Transactions," in 2023 2nd International Conference on Automation, Computing and Renewable Systems (ICACRS), IEEE, Dec. 2023, pp. 805–810. doi: 10.1109/ICACRS58579.2023.10404382.

[28] L. Yuningsih, G. A. Pradipta, D. Hermawan, P. D. W. Ayu, D. P. Hostiadi, and R. R. Huizen, "IRS-BAG-Integrated Radius-SMOTE Algorithm with Bagging Ensemble Learning Model for Imbalanced Data Set Classification," Emerg. Sci. J., vol. 7, no. 5, pp. 1501–1516, Oct. 2023, doi: 10.28991/ESJ-2023-07-05-04.

[29] S. Iqbal, K. M. Awan, S. Kamal, and Z. U. Rehman, "Interpretable Ensemble Learning Models for Credit Card Fraud Detection," Appl. Sci., vol. 15, no. 22, p. 12073, Nov. 2025, doi: 10.3390/app152212073.

[30] I. Almubark, "Advanced Credit Card Fraud Detection: An Ensemble Learning Using Random Under Sampling and Two-Stage Thresholding," IEEE Access, vol. 12, pp. 192079–192089, 2024, doi: 10.1109/ACCESS.2024.3519335.

[31] M. Gong, "A N OVEL P ERFORMANCE M EASURE FOR M ACHINE," Int. J. Manag. Inf. Technol., vol. 13, no. 1, pp. 1–19, 2021, doi: 10.5121/ijmit.2021.13101.