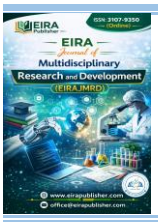

EIRA Journal of Multidisciplinary Research and Development (EIRAJMRD)
Journal homepage: https://eirapublisher.com/eirajmrd/
Email: office.eirapublisher@gmail.com
ISSN: 3107-9350 (Online)
Volume 2, Issue 3, (May-June) 2026
DOI: https://doi.org/10.5281/zenodo.20807097

DEVELOPMENT OF AN ENHANCED STARFISH OPTIMIZER-BASED ADABOOST ENSEMBLE MODEL FOR FINANCIAL FRAUD DETECTION

Iretiolu Yemisi Alabi*[1], Olufemi Olayanju Awodoye[2], Stephen Olatunde Olabiyisi[3], Elijah Olusayo Omidiora[4], Zubair Kamaldeen[5] and Oluwasina Adewumi [6]

[1,3,5,6]Department of Computer Science, Ladoke Akintola University of Technology, Ogbomoso, Oyo State, Nigeria.
[2,4]Department of Computer Engineering, Ladoke Akintola University of Technology, Ogbomoso, Oyo State, Nigeria.

Article history:
Received: 15/04/2026
Accepted: 25/05/2026
Published: 23/06/2026

Keywords: *Financial Fraud Detection, AdaBoost, Enhanced Starfish Optimization Algorithm (SFO)*

***Corresponding Author**:
Iretiolu Yemisi Alabi

Abstract **Original Research Article**

Machine learning has emerged as a powerful and indispensable tool for fraud detection due to its ability to learn complex patterns, adapt to evolving threats and make accurate prediction in real time by leveraging data-driven approaches. Although AdaBoost has proven effective for handling imbalanced fraud datasets, it remains sensitive to noisy data and prone to overfitting. Metaheuristic approaches such as the Starfish Optimization Algorithm (SFOA) have been applied to optimize AdaBoost but suffered from premature convergence. Consequently, this study developed an Enhanced Starfish Optimization Algorithm (ESFOA) with an Elite Preservation Strategy to improve the robustness and performance of the AdaBoost ensemble model for financial fraud detection.

A financial dataset of ten thousand (10,000) obtained from Kaggle's credit card fraud detection which comprises both legitimate and fraudulent transactions was utilized for the study. Preprocessing procedures included handling missing values through imputation, detecting and treating outliers using z-score and Interquartile Range (IQR) methods, performing feature selection and engineering, encoding categorical variables and scaling numerical attributes to ensure balanced and standardized input data. A baseline AdaBoost ensemble model with decision stumps as weak learners was developed, after which the Standard Starfish Optimization Algorithm was enhanced with an Elite Preservation Strategy to optimize key AdaBoost hyperparameters which include the number of estimators, learning rate, and tree depth through a defined fitness function. The enhanced algorithm iteratively evolves candidate solutions through exploration and exploitation mechanisms until convergence was achieved and the optimal parameters were used to retrain the classifier to produce the ESFOA-AdaBoost model, which was further validated using cross-validation techniques for robustness. Finally, the developed model was implemented in MATLAB (R2023a) Software and its performance was evaluated and compared with SFOA-Adaboost and Adaboost using false positive rate, sensitivity, specificity, precision, F1-score, accuracy and detection time.

The false positive rate, sensitivity, specificity, precision, F1-score, accuracy, and detection time for ESFOA-AdaBoost model were 2.22%, 98.90%, 97.78%, 99.05%, 98.81%, 98.57%, and 16.53 seconds, respectively. The corresponding SFOA-AdaBoost model and standard AdaBoost ensemble model were 6.67 and 8.67%, 97.00 and 96.14%, 93.33 and 91.33%, 97.14 and 96.28%, 96.52 and 95.48%, 95.90 and 94.70% and 24.50 and 31.32 seconds, respectively.

This developed ESFOA-AdaBoost ensemble model demonstrated superior effectiveness in financial fraud detection compared to both the conventional SFOA-AdaBoost and AdaBoost models. The developed ESFOA-AdaBoost model can serve as a reliable, scalable solution for real-time financial fraud detection systems.

How to cite this article: Iretiolu Yemisi Alabi, Olufemi Olayanju Awodoye, Stephen Olatunde Olabiyisi, Elijah Olusayo Omidiora, Zubair Kamaldeen and Adewumi Oluwasina. (2026). DEVELOPMENT OF AN ENHANCED STARFISH OPTIMIZER-BASED ADABOOST ENSEMBLE MODEL FOR FINANCIAL FRAUD DETECTION. EIRA Journal of Multidisciplinary Research and Development (EIRAJMRD), 2(3), 76-89.

1.0 INTRODUCTION

The rapid growth of online banking, electronic payments, and e-commerce has made financial fraud detection a critical challenge in modern financial systems. Financial fraud causes significant financial losses, undermines customer trust, increases operational costs, and threatens financial stability (Afjal et al., 2023; Catherine, 2019). Traditional fraud detection approaches have largely relied on rule-based systems and statistical models that identify suspicious transactions through predefined rules or mathematical analysis of transaction patterns (Aprico, 2020). Although widely adopted, these methods often suffer from high false-positive rates, limited adaptability to evolving fraud strategies, and an inability to detect previously unseen fraud schemes without frequent manual updates (Stamler et al., 2014).

To address these limitations, ensemble learning techniques have emerged as effective tools for fraud detection by combining multiple classifiers to improve predictive accuracy and robustness. Among these techniques, AdaBoost enhances classification performance by iteratively training weak learners and assigning greater importance to previously misclassified instances (Alabi *et al*, 2026, Shahraki et al., 2022; Randhawa et al., 2018). However, applying AdaBoost to financial fraud detection presents challenges, including high-dimensional data, severe class imbalance, and sensitivity to hyperparameter settings, all of which can negatively affect model performance and generalization capabilities (Chen et al., 2023; Kégl, 2009). Additionally, real-time fraud detection requires computationally efficient models capable of processing large transaction volumes with minimal delay.

Metaheuristic optimization techniques have gained prominence as effective solutions for hyperparameter tuning in ensemble models. Algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Differential Evolution (DE) employ stochastic search strategies to efficiently explore complex parameter spaces without requiring gradient information (Yang, 2020). These methods improve model performance by automating hyperparameter selection, reducing computational costs, and enhancing generalization ability (Li et al., 2023). More recently, the Starfish Optimizer (SFO), a bio-inspired algorithm that mimics starfish exploration, regeneration, and prey-searching behaviors, has demonstrated superior optimization performance through its balanced exploration and exploitation mechanisms (Zhong et al., 2025). Despite its promising results, limited research has investigated the application of SFO for optimizing boosting algorithms in financial fraud detection, highlighting an important research gap that this study seeks to address (Dey et al., 2025).

2.0 RELATED WORK

Metaheuristic algorithms have also been integrated with advanced learning models. Kumar et al. (2020) optimized a Deep Neural Network using the Whale Optimization Algorithm, whereas Zhou et al. (2020) combined the Dragonfly Algorithm with LightGBM for credit card fraud detection. Although these approaches improved precision and F1-score, they suffered from slow convergence and increased computational complexity when handling large-scale or noisy datasets.

Ensemble learning methods have demonstrated superior performance in fraud detection. Rahman et al. (2021) optimized a k-Nearest Neighbors model using the Artificial Bee Colony algorithm, while Bose et al. (2021) applied Cat Swarm Optimization to a stacking ensemble model. Similarly, Patel and Zhao (2022) optimized AdaBoost using Particle Swarm Optimization, achieving improved fraud detection accuracy. However, these approaches faced challenges such as high computational cost, premature convergence, and scalability issues.

Recent studies by Alabi *et al*, (2026), Chen and Li (2023) and Singh et al. (2024) further confirmed the effectiveness of ensemble classifiers and hyperparameter optimization in fraud detection. Nevertheless, conventional optimization methods such as Grid Search, Genetic Algorithms, and Particle Swarm Optimization often require extensive computational resources and may become trapped in local optima, limiting their ability to identify globally optimal solutions.

Despite the success of ensemble learning and metaheuristic optimization, limited research has explored the use of the Starfish Optimizer (SFO) for AdaBoost hyperparameter tuning in financial fraud detection. Therefore, this study proposes an Enhanced Starfish Optimizer-Based AdaBoost Ensemble Model to improve detection accuracy, convergence speed, and robustness while addressing the limitations of existing optimization approaches in highly imbalanced financial datasets.

3.0 METHODOLOGY

3.1 Research Approach

The study adopted a systematic approach to develop and evaluate an Enhanced Starfish Optimization Algorithm-based AdaBoost (ESFOA–AdaBoost) model for financial fraud detection. A financial transaction dataset containing both legitimate and fraudulent records was obtained from Kaggle's Fraud Detection Repository. Data preprocessing involved handling missing values through imputation, treating outliers using Z-score and Interquartile Range (IQR) methods, performing feature selection and feature engineering, encoding categorical variables, scaling numerical features, and addressing class imbalance to ensure high-quality input data for model training. A baseline AdaBoost classifier was

then developed using decision stumps as weak learners, with instance weights iteratively adjusted according to classification errors.

To improve the predictive performance of AdaBoost, the Standard Starfish Optimization Algorithm (SFOA) was enhanced by incorporating an elite preservation strategy and used to optimize key AdaBoost hyperparameters, including the number of estimators, learning rate, and tree depth. Through iterative exploration and exploitation processes, the Enhanced Starfish Optimization Algorithm (ESFOA) identified optimal parameter combinations, which were subsequently used to retrain the AdaBoost classifier, resulting in the ESFOA–AdaBoost model. The developed model was validated using cross-validation techniques and evaluated using confusion matrix-based metrics such as accuracy, precision, recall (sensitivity), specificity, F1-score, and false positive rate. Its performance was then compared with the baseline AdaBoost model to assess improvements in fraud detection effectiveness, robustness, and generalization capability.

3.2 Data Acquisition

The dataset comprises 10,000 synthetic financial transactions specifically generated for fraud detection research and machine learning model development acquired from www.kaggle.com. It is designed to mimic realistic user behavior alongside fraudulent patterns, thereby providing a secure and controlled environment for experimentation without exposing sensitive real-world data. Each transaction is characterized by several key features, including a unique transaction identifier, user identifier, transaction amount in local currency, and transaction type such as POS, online, ATM, or QR-based transactions.

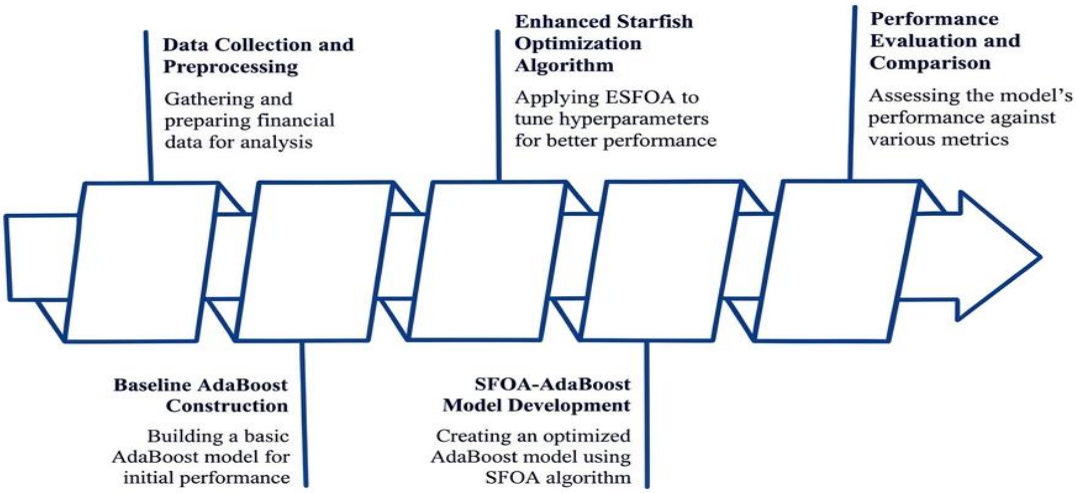

Figure 3.1: Schematic Diagram of Starfish Optimized AdaBoost Model for Fraud Detection

Additional attributes include merchant category, country of occurrence, transaction hour, and risk indicators such as device risk score and IP risk score, all of which contribute to assessing transaction legitimacy.

The dataset also contains a binary label indicating whether a transaction is fraudulent or legitimate, and it incorporates simulated fraud patterns such as unusually high transaction amounts, transactions in unfamiliar countries, night-time activities, rapid successive transactions, newly created accounts, and high-risk devices or IP addresses. 70% of the dataset were used for training and 30% of the dataset were used for testing using random subsampling cross-validation method.

3.3 Data Preprocessing

One of the first challenges addressed during preprocessing was the handling of missing values, which, if left untreated, could lead to biased model outcomes or reduced accuracy. In this research, missing values in numerical fields were imputed using the mean of the respective feature, ensuring that the central tendency of the data was preserved. For categorical features, mode imputation was applied to maintain consistency with the most frequent attribute values. In cases where missing data were extensive correlated, complete row or column deletion was acquired, provided it does not significantly reduce dataset integrity or representation.

Following missing value treatment, outlier detection and correction performed to eliminate anomalies that could distort the training process of the AdaBoost classifier. Outliers were identified using the Z-score method, where data points exceeding a specified standard deviation threshold (typically ±3) are flagged for review. Additionally, the Interquartile Range (IQR) technique was employed to detect extreme values beyond the acceptable range (1.5 × IQR below Q1 or above Q3), especially in transaction amounts. For datasets with high skewness or non-normal distributions, robust scaling techniques were applied to minimize the influence of outliers while retaining critical data patterns.

3.4 Feature Selection and Engineering

Effective feature selection and engineering played a vital role in enhancing the predictive performance of the Starfish Optimized AdaBoost Ensemble Model for financial fraud detection. To begin with, irrelevant or redundant features within the dataset are systematically identified and eliminated to reduce dimensionality and improve model generalization. Correlation analysis was employed to detect features that exhibit high interdependence, typically those with correlation coefficients above 0.85, indicating redundancy. Such features may contribute new information to the model and are either merged or dropped to prevent multicollinearity, which could impair the decision-making process of the ensemble model. Additionally, mutual information analysis was used to evaluate the dependency between each feature and the target variable (fraudulent or not), allowing the selection of features that contribute the most meaningful predictive information.

This initial stage involved identifying and removing irrelevant or redundant features using correlation analysis and mutual information (MI). The Pearson correlation coefficient (r) is calculated between pairs of features to detect multicollinearity using the formula:

$$r_{xy} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

where:

x_i, y_i : individual data points of features x and y

$\bar{x}, \bar{y}$: means of features x and y respectively

r_{xy} : correlation coefficient between x and y

Features with high correlation values (e.g., $|r| > 0.85$) was considered redundant and may be removed. Additionally, Mutual Information (MI) between each feature X and the target variable Y is computed to evaluate how much information a feature contributes to predicting fraud. The MI is defined as:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \left(\frac{p(x,y)}{p(x) \cdot p(y)} \right)$$

where:

$p(x,y)$: joint probability distribution of X and Y

$p(x), p(y)$: marginal probability distributions of X and Y

Features with low mutual information scores are dropped to reduce noise and computational complexity.

Beyond removing irrelevant features, feature engineering was carried out to create new variables that offer deeper insights into transactional behavior patterns. For instance, transaction frequency per user is computed to capture unusual activity over a given time window an indicator often associated with fraud. Similarly, a credit-to-debt ratio feature was engineed, especially in cases where credit limit and usage history are available, to reflect a customer's financial health and risk level. Time-based features such as hour of transaction, day of the week, or time since last transaction are also derived from available timestamps to identify fraud-prone periods. This engineered features not only improve the model's capacity to detect subtle patterns of fraudulent behavior but also enhance interpretability, making the model more suitable for real-world credit risk management applications.

Beyond selection, feature engineering is used to derive new variables that may enhance the model's ability to detect fraudulent activities. Transaction frequency can be calculated as:

$$TF_i = \frac{N_i}{T}$$

where:

TF_i : transaction frequency of user i

N_i : number of transactions made by user i

T: total observation period

Similarly, the credit-to-debt ratio (CDR) can be computed as:

$$CDR_i = \frac{CL_i}{DU_i}$$

where:

CDR_1 : credit-to-debt ratio of user i

CL_1 : credit limit of user i

DU_i : debt used or outstanding credit of user i

Additional engineered features may include time-based indicators such as "hour of day", "day of week", and "elapsed time since last transaction", all derived from timestamp data to capture behavioral patterns. This engineered features not only improve model performance by capturing hidden relationships in the data but also make the model more interpretable for credit risk analysts.

3.5 Data Encoding

Data encoding is a technique used in preparing categorical variables for input into the Starfish Optimized AdaBoost Ensemble Model, which inherently requires numerical data. This research categorical variables present in the dataset such as transaction type, customer status was transformed using one-hot encoding or label encoding, depending on the nature of the variable. One-hot encoding was applied to nominal variables with no intrinsic ordering, creating binary columns for each category to ensure equal weight and prevent misleading relationships. For example, a categorical feature like "transaction type" with values such as "purchase," "withdrawal," and "transfer" was converted into three separate binary columns. On the other hand, label encoding was used for ordinal categorical features, where category labels are replaced with unique integers that reflect their inherent order. This encoding techniques ensure that the categorical data are effectively represented in a numerical format, allowing the AdaBoost model to process them efficiently without introducing bias or invalid assumptions about the relationships between categories.

In One-Hot Encoding (OHE), let a categorical variable C have n unique categories:

$$C = \{c_1, c_2, \dots, c_n\}$$

One-hot encoding transforms these into an n-dimensional binary vector $\mathbf{v}$, where for a given observation with category c_j, the encoded vector is:

$$\mathbf{v}_i = [0,0,\dots,1,\dots,0] \text{ where the 1 is in the } j^{\text{th}} \text{ position}$$

For example, if $C = \{$ purchase, withdrawal, transfer $\}$, and the transaction is a "withdrawal", then:

$$\text{OHE(withdrawal)} = [0,1,0]$$

In Label Encoding (LE), s Label encoding maps each unique category $c_f \in C$ to a unique integer.

$$LE(c_j) = j \text{ where } j \in \{0,1,\dots,n-1\}$$

This is suitable when the categorical variable has an inherent order (ordinal data). For instance, if $C = \{$low, medium, high$\}$, a valid encoding would be:

$$LE(\text{ low }) = 0, LE(\text{ medium }) = 1, LE(\text{ high }) = 2$$

3.6 Normalization

Normalization ensures all numerical features contribute equally to the learning process of the Starfish Optimized AdaBoost Ensemble Model. This research, Min-Max normalization was applied to rescale numerical features to a common range, typically [0,1], thereby eliminating bias caused by differing value magnitudes. This is particularly important in fraud detection tasks, where features such as transaction amounts may span a wide scale and dominate the learning process if left unnormalized. This transformation was performed using the Min-Max normalization formula:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

where:

- x is the original value of the feature
- x_{mln} and $x_{\max}$ are the minimum and maximum values of the feature across the dataset
- x' is the normalized value

By transforming all numerical features (transaction amount, time intervals, frequency counts) into the same scale, the model becomes more stable and converges more efficiently during training.

3.7 Formulation of the Enhanced Starfish Optimization Algorithm (ESFO)

The Starfish Optimization Algorithm (SFOA) described in Algorithm 3.1 is a population-based metaheuristic inspired by the natural behavior of starfish in searching for food and regenerating damaged limbs. The algorithm begins with population initialization using the equation

$$x_i = LB + \text{rand}(0,1)(UB - LB),$$

which ensures uniform distribution of candidate solutions within the search space. Each starfish represents a potential solution, and its fitness is evaluated using the objective function $f(x_i)$. The global best starfish is identified using

$$x^* = \arg\min_{x_i} f(x_i),$$

which serves as the leader guiding the optimization process. This initialization procedure is identical in both Algorithm 3.1 and Algorithm 3.2.

In the exploration phase of the standard SFOA, starfish positions are updated using the equation

$$x_i^{t+1} = x_i^t + \alpha \text{rand}(0,1)\left(x_j^t - x_k^t\right)$$

This equation promotes global exploration by allowing each starfish to move based on the relative difference between randomly selected starfish. The exploration coefficient $\alpha \in (0,1)$ regulates the step size of movement. Larger values of α encourage broader exploration, while smaller values encourage fine adjustments. This mechanism helps prevent premature convergence but does not guarantee retention of high-quality solutions. The exploitation phase of Algorithm 3.1 enhances convergence by guiding starfish toward the global best solution using the equation

$$x_i^{t+1} = x_i^{t+1} + \beta \text{rand}(0,1)(x^* - x_i^t).$$

This equation strengthens local search by reducing the distance between each starfish and the optimal solution. The exploitation coefficient $\beta \in (0,1)$ controls convergence speed. However, this approach does not explicitly protect elite solutions from being modified in subsequent iterations. Consequently, previously discovered optimal solutions may be lost during population updates.

Another important component of the standard SFOA is the regeneration phase, which introduces additional randomness using the equation

$$x_i^{t+1} = x_i^{t+1} + \gamma \text{rand}(0,1)(UB - LB)$$

This regeneration mechanism increases diversity and helps the algorithm escape local optima. The regeneration coefficient $\gamma \in (0,1)$ determines the magnitude of random perturbation. While this improves exploration capability, it may also disrupt convergence stability. Therefore, although regeneration enhances diversity, it may reduce solution consistency. To overcome this limitations, Algorithm 3.2 introduces the Enhanced Starfish Optimization Algorithm (ESFOA) by integrating an elite preservation strategy. After initialization, elite starfish are selected using the equation

$$E = \{x_1, x_2, \dots, x_{N_e}\},$$

where N_e represents the number of elite starfish. These elite solutions correspond to the lowest fitness values. Unlike the standard SFOA, this elite starfish is preserved without modification. This ensures that the best solutions are retained across generations.

The exploration equation in ESFOA remains mathematically similar to that of the standard SFOA, expressed as

$$x_i^{t+1} = x_i^t + \alpha \text{rand}(0,1)\left(x_j^t - x_k^t\right)$$

However, this update is applied only to non-elite starfish, while elite starfish remain unchanged. This selective update ensures that high-quality solutions are not degraded. By preserving elite starfish, the algorithm improves stability and convergence reliability. Thus, the exploration phase becomes more efficient compared to the standard SFOA. The exploitation phase in ESFOA also follows a similar structure using the equation

$$x_i^{t+1} = x_i^{t+1} + \beta \text{rand}(0,1)(x^* - x_i^t)$$

However, the key difference is that the global best solution x^* is always preserved due to elite preservation. In contrast, the standard SFOA may lose the best solution during regeneration or random updates and ESFOA guarantees fitness improvement by equation 3.19. Mathematically, this ensures that

$$x^* \in E \subseteq P^{t+1}$$

Where P^{t+1} is the updated population. This guarantees monotonic improvement in solution quality.

The most significant enhancement in ESFOA is the elite reinsertion mechanism, defined as

$$P^{t+1} = \text{Best}_N(\{x_1^{t+1}, x_2^{t+1}, \dots, x_N^{t+1}\} \cup E)$$

This equation combines the updated population with the elite set and selects the best N solutions. This ensures that elite solutions are never lost during optimization. In contrast, the standard SFOA lacks such a mechanism. As a result, ESFOA achieves superior convergence stability. Another key difference between the two algorithms lies in the absence of the regeneration equation in ESFOA. The standard SFOA includes the regeneration equation

$$x_i^{t+1} = x_i^{t+1} + \gamma \text{rand}(0,1)(UB - LB),$$

which introduces uncontrolled randomness. In ESFOA, this term is removed and replaced with elite preservation and reinsertion. This reduces unnecessary randomness while maintaining solution diversity through exploration and exploitation equations. Consequently, ESFOA achieves a better balance between exploration and exploitation.

Algorithm 3.2: Enhanced Starfish Optimization Algorithm (ESFOA)

Step 1: Population Initialization

i. Randomly initialize the starfish population within the defined search space.
ii. Evaluate the fitness of each starfish using the objective function.
iii. Identify the best-performing starfish as the global best solution.
iv. Set the iteration counter.

Step 2: Elite Preservation Strategy

i. Rank all starfish according to their fitness values.
ii. Select the top-performing starfish as the elite set.
iii. Preserve the elite solutions without modification for the next iteration.

Step 3: Starfish Exploration Phase

i. Update the positions of non-elite starfish through random interactions with other starfish.
ii. Encourage exploration of new regions within the search space to enhance global search capability.

Step 4: Starfish Exploitation Phase

i. Move non-elite starfish toward the current global best solution.
ii. Refine promising solutions and improve convergence toward the optimum.

Step 5: Boundary Control

i. Check all updated positions and ensure they remain within the allowable search boundaries.
ii. Correct any solutions that violate the predefined limits.

Step 6: Fitness Evaluation

Evaluate the fitness of all updated starfish based on the objective function.

Step 7: Elite Reinsertion

i. Combine the updated population with the preserved elite solutions.
ii. Rank the combined population according to fitness.
iii. Select the best-performing starfish to form the population for the next iteration.

Step 8: Global Best Update

Identify and update the current global best solution from the population.

Step 9: Termination Check

i. Increment the iteration counter.
ii. Repeat Steps 2–8 until the maximum number of iterations or convergence criterion is reached.

Step 10: Output Generation

Return the global best solution as the optimal solution obtained by the Enhanced Starfish Optimization Algorithm.

3.8 Development of the Enhanced Starfish-Optimized AdaBoost Model (ESFO-AdaBoost)

The development of the Enhanced Starfish-Optimized AdaBoost Ensemble Model (ESFO-AdaBoost) begins with the formulation of the financial transaction dataset.

$$D = \{(X_n, y_n)\}_{n=1}^{N_s}$$

In this dataset, X_n represents the feature vector of the n-th financial transaction, and $y_n \in \{0,1\}$ denotes the corresponding class label indicating legitimate or fraudulent activity. The primary objective was to develop an optimized AdaBoost ensemble classifier capable of accurately

distinguishing fraudulent transactions. Since AdaBoost performance depends heavily on its hyperparameters, optimization was necessary to achieve maximum detection accuracy. Therefore, the Enhanced Starfish Optimization Algorithm (ESFOA) is integrated to optimize AdaBoost parameters efficiently.

In the ESFO-AdaBoost model, each starfish represents a candidate AdaBoost parameter vector defined as

$$x_i = \theta_i = \left[N_{\text{est},i}, \eta_i, d_i\right]$$

Here, $N_{\text{est},i}$ represents the number of estimators, η_i is the learning rate, and d_i denotes the decision tree depth. This encoding transforms hyperparameter tuning into a multidimensional optimization problem. Each starfish position corresponds to a complete AdaBoost configuration. Consequently, the optimization algorithm directly influences model performance through parameter adjustments. The population initialization process is performed using the equation

$$x_i = LB + \text{rand}(0,1)(UB - LB)$$

ensuring random distribution within parameter bounds. This guarantees diversity in the initial population of candidate AdaBoost models. Discrete parameters such as $N_{\text{est},i}$ and d_i are converted using rounding operations. This ensures valid integer values for tree-based classifiers. The initial population forms the basis for ESFOA optimization.

Each starfish configuration was used to train an AdaBoost classifier defined by the ensemble equation

$$F(x) = \sum_{m=1}^{N_{\text{ext}}} \alpha_m h_m(x)$$

In this equation, $h_m(x)$ represents the weak learner, and α_m is its corresponding weight. The weight of each weak learner was computed using

$$\alpha_m = \frac{1}{2}\ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$$

where ϵ_m represents classification error. This formulation allows AdaBoost to emphasize misclassified samples. As a result, the ensemble model becomes more accurate over successive iterations. The fitness of each starfish was evaluated using the classification error equation Error

$$_i = \frac{1}{N_s}\sum_{n=1}^{N_s} 1(y_n \neq \hat{y}_n)$$

The equation shows that the classification error equation use to evaluate the performance of each candidate solution (starfish) during optimization.

Error $_i$ = Classification error of the $_i th$ starfish (candidate solution).
N_s = Total number of samples in the dataset.
y_n = Actual (true) class label of sample n.
$\hat{y}_n$ = Predicted class label of sample n.
$1(y_n \neq \hat{y}_n)$ = indicator function:

$$1(y_n \neq \hat{y}_n) = \begin{cases} 1, \text{ if the sample is misclassified} \\ 0, \text{ if the samples are correctly classified} \end{cases}$$

The equation counts the number of incorrectly classified samples and divides it by the total number of samples. Therefore, it gives the proportion of misclassified instance.

A lower error value indicates better classification performance. In your enhanced Starfish Optimizer, this classification error serves as the fitness function, where the optimizer searches for parameter settings that minimize the error and improve the AdaBoost fraud detection model.

This equation computes the proportion of misclassified transactions. The fitness function is defined as $f(x_i) = \text{Error}_i$, where lower values indicate better fraud detection performance. This formulation converts the optimization objective into a minimization problem. Thus, ESFOA aims to minimize classification error by optimizing AdaBoost parameters.

The elite preservation strategy was introduced to retain high-quality solutions throughout the optimization process. The elite set is defined as

$$E^t = \left\{x_1^t, x_2^t, \ldots, x_{N_c}^t\right\}$$

where N_e represents the number of elite starfish. This elite starfish corresponds to the best-performing AdaBoost configurations. Unlike standard optimization methods, elite solutions are preserved without modification. This ensures that optimal parameter configurations are never lost. The exploration phase of ESFOA updates non-elite starfish using the equation

$$x_i^{t+1} = x_i^t + \text{arand}(0,1)\left(x_j^t - x_k^t\right)$$

This equation allows starfish to explore new regions of the search space. The exploration coefficient $\alpha \in (0,1)$ controls the magnitude of movement. Random selection of starfish x_j^t and x_k^t ensures diversity. This mechanism helps prevent premature convergence.

The exploitation phase enhances convergence by guiding starfish toward the best solution using

$$x_i^{t+1} = x_i^{t+1} + \beta\text{rand}(0,1)(x^* - x_i^t)$$

Here, x^* represents the global best starfish. The exploitation coefficient $\beta \in (0,1)$ controls convergence strength. This equation ensures gradual movement toward optimal AdaBoost parameters. As a result, the algorithm achieves improved optimization accuracy. Boundary control was applied using

$$x_i^{t+1} = \min(\max(x_i^{t+1}, LB), UB)$$

to maintain valid parameter values. These prevents hyperparameters from exceeding predefined limits. Discrete parameters are rounded to ensure valid classifier

configurations. This step guarantees feasibility of solutions. It also maintains algorithm stability.

The fitness reevaluation phase retrains AdaBoost using updated parameters and recomputes fitness values. This ensures that each starfish represents an updated classifier configuration. The fitness function remains defined as

$$f(x_i^{t+1}) = \text{Error}_{\ i}^{\ t+1}$$

Continuous fitness evaluation allows progressive improvement. Thus, ESFOA iteratively refines AdaBoost parameters. Elite reinsertion was performed using the equation

$$P^{t+1} = \text{Best}_N\left(P_{\text{updated}}^{t+1} \cup E^t\right)$$

This combines updated starfish and elite starfish into a single population. The Best $_N$ operator selects the top-performing solutions. This guarantees retention of elite solutions. Consequently, convergence stability is improved. The global best solution was updated using the equation

$$x^* = \arg\ \min f(x_i)$$

which selects the best performing AdaBoost configuration. This ensures continuous improvement in fraud detection performance. The optimization process continues until the termination condition $t = T$ was reached. At termination, the optimized parameter vector is defined as $\theta^* = [N_{est}^*, \eta^*, d^*]$. This vector represents the optimal AdaBoost hyperparameters. The optimized AdaBoost model was trained using the equation

$$F^*(x) = \sum_{m=1}^{N_{st}^*} \alpha_m h_m(x)$$

This optimized model provides enhanced fraud detection performance. The integration of elite preservation ensures consistent improvement in parameter optimization. The ESFO-AdaBoost model achieves faster convergence and higher accuracy. Therefore, the enhanced optimization framework provides a robust and efficient solution for financial fraud detection.

3.9 Implementation of the Developed Model

The implementation of the study was done in MATLAB R2023a which requires a structured combination of machine learning tools and optimization routines. The study was utilized the Statistics and Machine Learning Toolbox for training the AdaBoost ensemble and evaluating classification performance, and the Optimization Toolbox to implement the Starfish Optimization Algorithm (SFO) for hyperparameter tuning. Additional support from the Parallel Computing Toolbox may be required to accelerate fitness evaluations during iterations, especially with large datasets or during cross-validation. The implementation was conducted on 64-bit Operating System of Windows 11 with 13th Gen Intel(R) Core (TM) i7-1355U (1.70 GHz) processor and 16.0 GB (15.7 GB usable) of installed RAM.

Algorithm 3.3: ESFOA-Optimized AdaBoost Ensemble Model for Financial Fraud Detection

Step 1: Encode each starfish as an AdaBoost hyperparameter vector consisting of the number of estimators, learning rate, and tree depth.

Step 2: Randomly initialize the starfish population within the predefined parameter bounds.

Step 3: Train an AdaBoost classifier using the hyperparameters represented by each starfish.

Step 4: Evaluate the performance of each AdaBoost model using a fitness function based on classification error.

Step 5: Rank all starfish according to their fitness values and preserve the top-performing solutions using an elite preservation strategy.

Step 6: Perform the exploration phase by updating non-elite starfish positions to search new regions of the solution space.

Step 7: Perform the exploitation phase by guiding starfish toward the current best solution to refine promising hyperparameter combinations.

Step 8: Apply boundary control to ensure all parameter values remain within their allowable ranges.

Step 9: Retrain AdaBoost using the updated hyperparameters and re-evaluate the fitness of each starfish.

Step 10: Reinsert the preserved elite solutions into the updated population and retain the best-performing starfish.

Step 11: Update the global best solution based on the current fitness values.

Step 12: Repeat the optimization process until the maximum number of iterations or convergence criterion is reached.

Step 13: Obtain the optimal AdaBoost hyperparameters and train the final ESFOA-AdaBoost model for financial fraud detection.

Step 14: Evaluate the final model using performance metrics such as accuracy, precision, recall (sensitivity), specificity, F1-score, and false positive rate.

3.10 Performance Evaluation of the Developed Model

The performance evaluation of the developed model for financial fraud detection was carried out using a comprehensive set of metrics. False Positive Rate (FPR), Sensitivity (Recall), Specificity, Precision, F1-score, Accuracy, and Detection Time. This metrics were derived from the confusion matrix, which outlines the prediction results in terms of four components: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). Each metric highlighted a different aspect of model performance, especially in scenarios where fraud cases represent a small portion of the dataset (class imbalance). Understanding each metric allowed for a balanced assessment of both the correctness and efficiency of the model.

i. False Positive Rate (FPR) was a crucial metric in fraud detection, representing the proportion of legitimate (non-fraud) cases incorrectly identified as fraud. A high FPR could lead to customer dissatisfaction and operational inefficiencies. It is calculated as:

$$\text{FPR} = \frac{FP}{FP + TN}\ X\ 100$$

A lower FPR was desired, indicating the model is effective in minimizing false alarms while still identifying actual fraud instances.

ii. Sensitivity (Recall) measured the model's ability to correctly identify fraudulent cases. It is critical in fraud detection, where failing to identify fraud (false negatives) can result in significant financial losses. The formula for sensitivity is:

$$\text{Sensitivity (Recall)} = \frac{TP}{TP+FN}\ \text{X}\ 100$$

A high sensitivity implied that the model was effective in catching most of the actual fraud cases, even if it sometimes flags non-fraud cases.

iii. Specificity reflected the model's effectiveness in correctly identifying non-fraudulent (legitimate) transactions. It is the complement of the false positive rate and is defined as:

$$\text{Specificity} = \frac{TN}{TN+FP}\ X\ 100$$

High specificity ensured that legitimate customers were not wrongly flagged as fraudulent, which is important for maintaining trust and operational efficiency in credit systems.

iv. Precision, or Positive Predictive Value, indicated the proportion of predicted frauds that are actually fraud. It evaluated how reliable the model's positive predictions are, and is calculated as:

$$\text{Precision} = \frac{TP}{TP + FP}\ X\ 100$$

A high precision implied fewer false alarms, meaning that when the model predicts fraud, it is usually correct. This is particularly beneficial for reducing investigation costs and improving trust in the model.

The F1-score provided a balanced measure by combining precision and recall into a single metric it was the harmonic mean of precision and recall and is especially useful in cases of class imbalance:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

A high F1-score reflected both good detection of fraud and minimal false positives, ensuring a balanced and robust performance.

v. Accuracy measured the overall proportion of correct predictions (both fraud and non-fraud) made by the model:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}\ X\ 100$$

Detection Time (DT) is defined as the average computational time taken by the model to detect fraud per transaction or batch, can be expressed as:

$$DT = \frac{1}{N}\sum_{i=1}^{N} t_i$$

Where:

t_i = Detection time for the i^{th} transaction

N = Total number of transactions processed

This Equation (3.42) measures the average processing time required by the fraud detection model, providing an indicator of the model's computational efficiency in real-time or batch fraud detection systems.

3.11 Comparison of the developed model with the standard SFOA-AdaBoost and the existing AdaBoost Ensemble models. The performance of the developed model was compared with the set of metrics in equ 3.10, that is false positive rate, sensitivity, specificity, precision, F1-score, Accuracy and Detection time.

4.0 RESULTS AND DISCUSSION

Evaluating the performance of the AdaBoost, SFOA–AdaBoost, and ESFOA–AdaBoost models for financial fraud detection. Comparative analysis was conducted using multiple evaluation metrics, including False Positive Rate (FPR), Sensitivity, Specificity, Precision, F1-score, Accuracy, and Detection Time. In addition, the convergence behavior of the optimization algorithms was analyzed to evaluate the effectiveness of the Elite Preservation Strategy integrated into the Enhanced Starfish Optimization Algorithm (ESFOA). Table 1 presents the combined evaluation results of the conventional AdaBoost model, the SFOA–AdaBoost model, and the ESFOA - AdaBoost framework at the optimum threshold value.

Table 1: Comparative Evaluation Results of AdaBoost, SFOA–AdaBoost, and ESFOA–AdaBoost

Technique	FPR (%)	Sensitivity (%)	Specificity (%)	Precision (%)	F1-Score (%)	Accuracy (%)	Detection Time (sec)
AdaBoost	8.67	96.14	91.33	96.28	95.48	94.70	31.32
SFOA–AdaBoost	6.67	97.00	93.33	97.14	96.52	95.90	24.50
ESFOA–AdaBoost	2.22	98.90	97.78	99.05	98.81	98.57	16.53

The ESFOA-AdaBoost model achieved the best overall performance with an accuracy of 98.57%, F1-score of 98.81%, and sensitivity of 98.90%. The model also produced the lowest false positive rate of 2.22% and the shortest detection time of 16.53 seconds. The results indicate that integrating elite preservation into the Starfish Optimization Algorithm improved both predictive accuracy and computational efficiency. The results indicate that the ESFOA–AdaBoost framework consistently achieved superior performance across all evaluation metrics when compared with both the standard AdaBoost and SFOA–AdaBoost models.

False Positive Rate (FPR) is an important metric in fraud detection because it measures the proportion of legitimate transactions incorrectly classified as fraudulent. High false positive rates can negatively affect customer trust and increase operational costs due to unnecessary transaction verification. The conventional AdaBoost model recorded an FPR of 8.67%, indicating a relatively high level of false alarms. With the integration of the Starfish Optimization Algorithm, the FPR reduced to 6.67%, demonstrating that optimization improved the discriminative capability of the classifier. However, the ESFOA–AdaBoost framework achieved the lowest FPR of 2.22%, representing a substantial reduction in false positive predictions. This improvement can be attributed to the Elite Preservation Strategy integrated into ESFOA. By retaining high-quality candidate solutions throughout the optimization process, ESFOA avoids loss of optimal hyperparameter configurations and improves convergence reliability. Consequently, the optimized AdaBoost model is better able to distinguish between legitimate and fraudulent transaction patterns.

Sensitivity, also referred to as recall, measures the ability of the model to correctly identify fraudulent transactions. In financial fraud detection systems, high sensitivity is critical because undetected fraudulent transactions can lead to severe financial losses. The standard AdaBoost model achieved a sensitivity of 96.14%, while SFOA–AdaBoost improved this value to 97.00%. The ESFOA–AdaBoost framework further increased sensitivity to 98.90%, indicating superior fraud detection capability. The improvement demonstrates that optimized hyperparameter selection significantly enhances the ability of the ensemble model to learn minority fraud patterns within highly imbalanced datasets. The elite retention mechanism in ESFOA further strengthens this capability by maintaining high-performing solutions during successive optimization iterations.

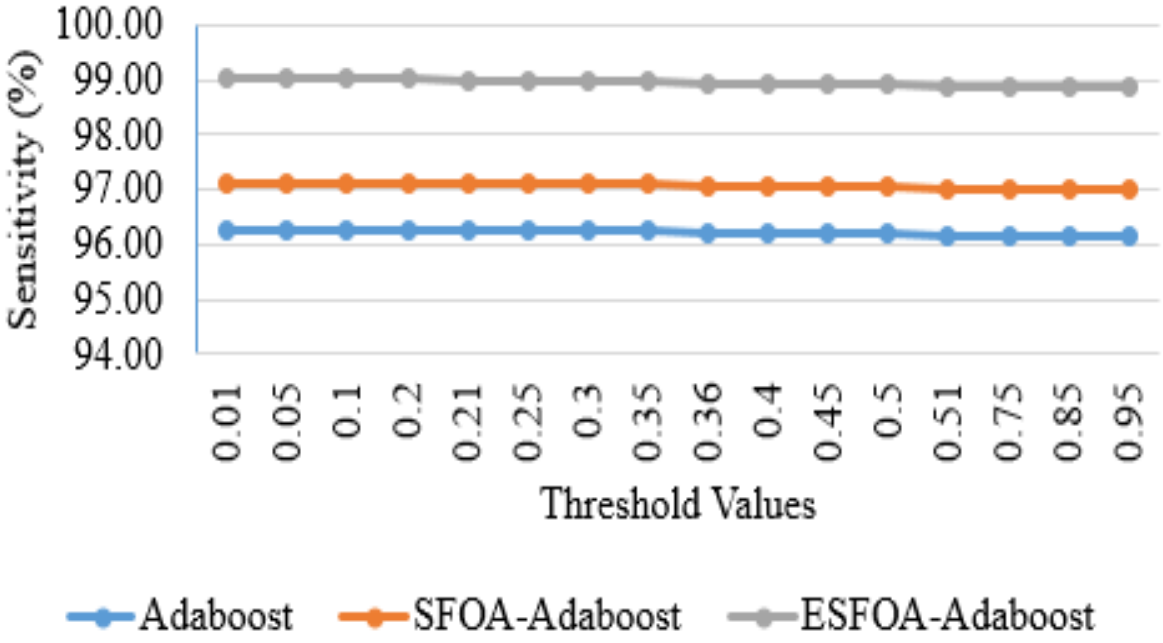

Figure 1: Sensitivity against Threshold Values

Figure 1 illustrates the variation of sensitivity across different threshold values for the three models. ESFOA-AdaBoost consistently achieved higher sensitivity compared with AdaBoost and SFOA-AdaBoost, indicating improved fraud detection capability with fewer missed fraudulent transactions.

Specificity measures the ability of the model to correctly identify legitimate transactions, while precision evaluates the proportion of predicted fraud cases that are actually fraudulent. The specificity values increased from 91.33% for AdaBoost to 93.33% for SFOA–AdaBoost and eventually to 97.78% for ESFOA–AdaBoost. Similarly, precision improved from 96.28% to 97.14% and finally to 99.05%. These improvements indicate that the optimization process effectively reduced false alarms and improved the reliability of fraud predictions. The superior precision achieved by ESFOA–AdaBoost demonstrates that the framework minimizes unnecessary fraud alerts while maintaining strong detection capability.

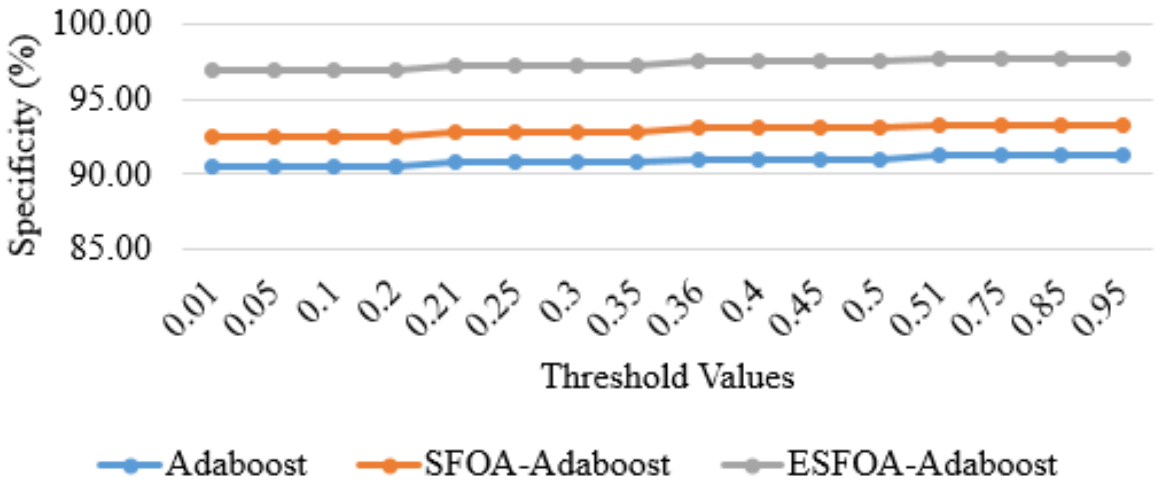

Figure 2: Specificity against Threshold Values

Figure 2 shows that ESFOA-AdaBoost maintained superior specificity across threshold levels, demonstrating improved discrimination of legitimate transactions and reduced false alarms.

The F1-score provides a balanced evaluation of precision and sensitivity, making it particularly suitable for imbalanced fraud detection tasks. The F1-score improved from 95.48% in the conventional AdaBoost model to 96.52% in SFOA–AdaBoost and reached 98.81% in ESFOA–AdaBoost. Similarly, overall classification accuracy increased from 94.70% to 95.90% and eventually to 98.57%. These findings demonstrate that the ESFOA–AdaBoost framework achieved the most balanced and reliable classification performance among the evaluated models. The combination of elite-guided optimization and adaptive boosting improved both fraud detection capability and overall classification consistency. Figure 3 shows that ESFOA-AdaBoost achieved the highest F1-score throughout the threshold range, demonstrating a stronger balance between precision and recall in highly imbalanced fraud detection tasks.

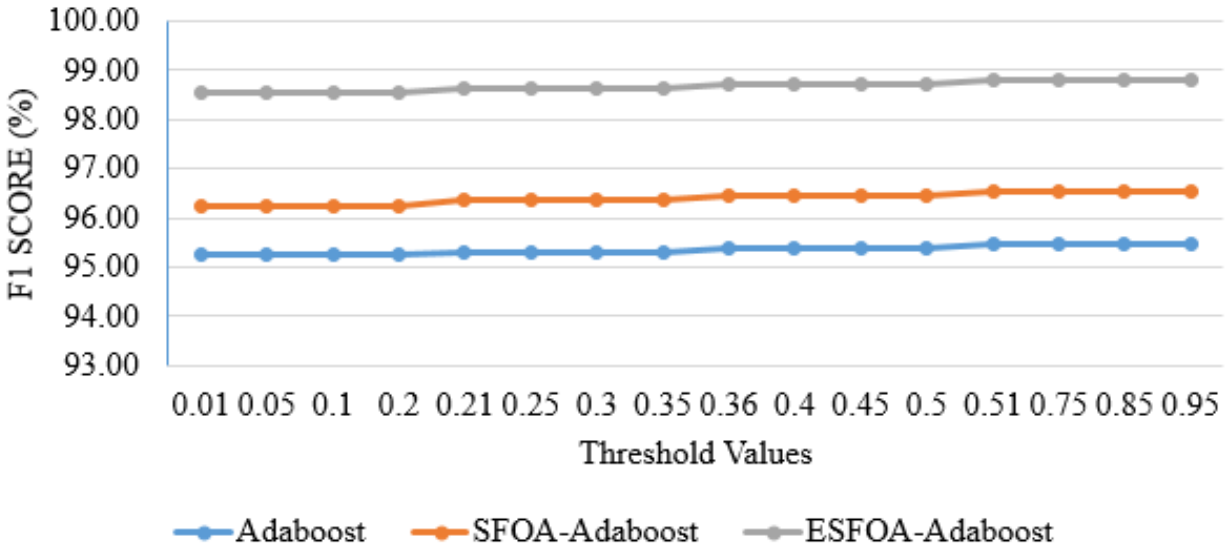

Figure 3: F1-Score against Threshold Values

Detection time is an important consideration in real-time fraud detection systems where rapid decision-making is required to prevent financial loss. The conventional AdaBoost model required 31.32 seconds for detection, while SFOA–AdaBoost reduced the detection time to 24.50 seconds. The ESFOA–AdaBoost framework achieved the shortest detection time of 16.53 seconds. The reduced computational time achieved by ESFOA is primarily due to the elite preservation mechanism, which accelerates convergence by retaining high-fitness candidate solutions and reducing redundant exploration of poor regions within the search space. Figure 4 presents the computational efficiency of the models. ESFOA-AdaBoost required the least detection time, showing that elite-guided optimization accelerated convergence and reduced computational overhead. This finding demonstrates that the framework not only improves predictive performance but also enhances computational efficiency, making it more suitable for real-time fraud detection environments.

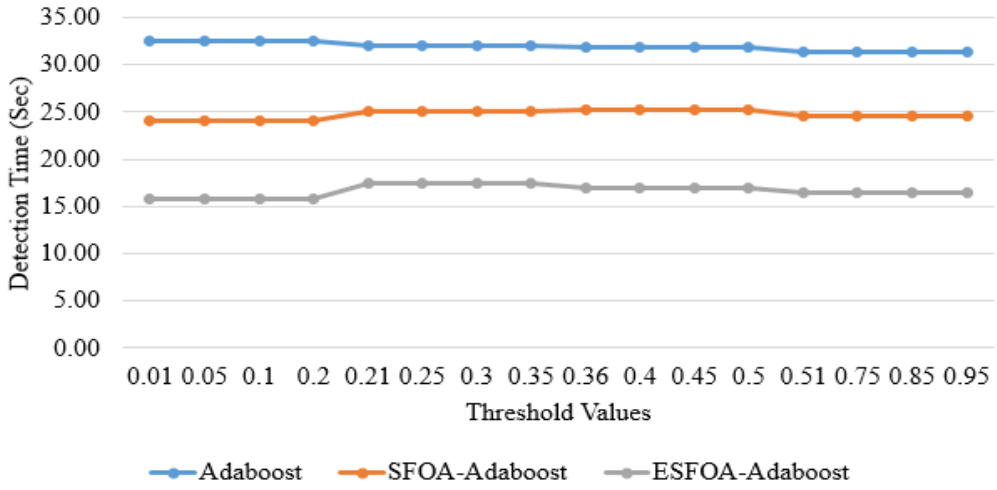

Figure 4: Detection Time against Threshold Values

The convergence characteristic analysis further demonstrates the effectiveness of the ESFOA optimization strategy. Compared with the standard Starfish Optimization Algorithm, ESFOA converged more rapidly toward optimal fitness values and maintained greater stability throughout the optimization process. The Elite Preservation Strategy contributed significantly to this improvement by ensuring that the best candidate solutions were retained across iterations. This prevented premature loss of high-quality solutions and improved the balance between exploration and exploitation.

The convergence behavior confirms that ESFOA provides a more stable and reliable optimization mechanism than the conventional SFOA approach. Figure 5 compares the convergence behavior of SFOA and ESFOA. The ESFOA algorithm converged more rapidly toward the optimal fitness value and maintained greater stability throughout the optimization process. The inclusion of elite preservation improved convergence reliability by retaining high-quality candidate solutions during successive iterations.

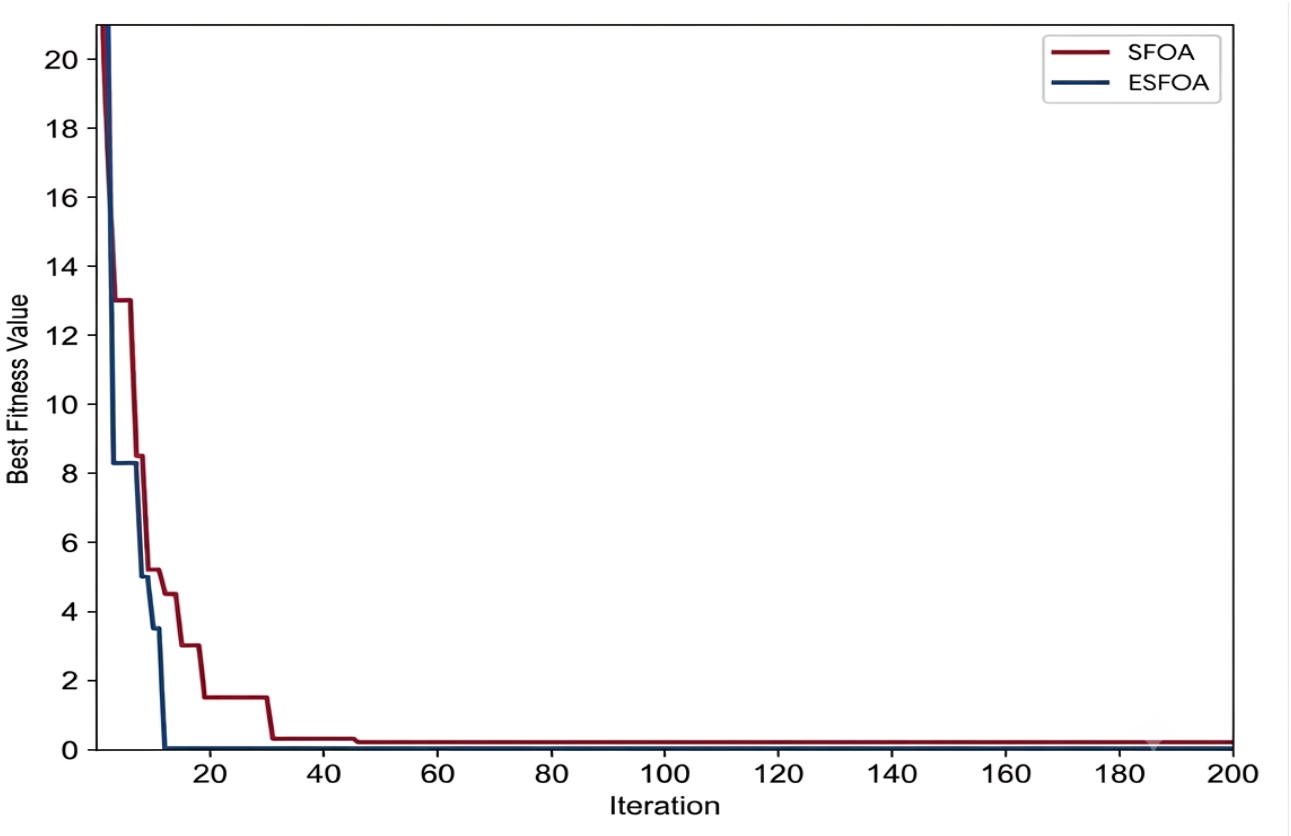

Figure 5: Convergence Curve of SFOA and ESFOA

The ESFOA - AdaBoost framework was also compared with recent state-of-the-art fraud detection approaches reported in the literature. Table 2 compares the ESFOA-AdaBoost model with recent fraud detection approaches reported in the literature. The developed model achieved higher accuracy, lower false positive rate, and faster detection time than existing optimized ensemble and deep learning approaches. These findings demonstrate the effectiveness of elite-guided starfish optimization for improving ensemble learning performance in financial fraud detection systems.

Table 2: Comparison with Existing Fraud Detection Models

Technique / Model	Optimization Strategy	Accuracy (%)	FPR (%)	F1-Score (%)	Detection Time (sec)	Reference
CNN–LSTM Hybrid	Manual tuning	96.20	5.10	96.00	28.40	Abdel-Basset *et al.* (2022)
PSO-Optimized Random Forest	Particle Swarm Optimization	97.10	4.30	97.00	22.60	Li *et al.* (2022)
GA–XGBoost	Genetic Algorithm	97.85	3.90	97.60	21.10	Kumar *et al.* (2023)
WOA–AdaBoost	Whale Optimization Algorithm	98.05	3.20	98.10	18.90	Elaziz *et al.* (2023)
ESFOA–AdaBoost (Developed)	**Elite Starfish Optimization**	**98.57**	**2.22**	**98.81**	**16.53**	**This study**

The ESFOA - AdaBoost framework achieved the highest accuracy and F1-score while also recording the lowest false positive rate and detection time among the compared methods. These findings demonstrate that integrating elite-guided swarm optimization with ensemble learning provides a highly effective framework for financial fraud detection. The framework successfully balances predictive accuracy, convergence stability, and computational efficiency, making it suitable for real-world fraud detection applications. The obtained results demonstrate that optimization-driven ensemble learning significantly improves fraud detection performance compared with the conventional AdaBoost model. Furthermore, incorporating elite preservation into the optimization process enhances convergence stability, improves classification reliability, and reduces computational overhead.

5.0 CONCLUSION

The developed ESFOA-AdaBoost ensemble model demonstrated superior performance in financial fraud detection compared with both the traditional AdaBoost and SFOA-AdaBoost models. By integrating an Elite Preservation Strategy into the Starfish Optimization Algorithm, the model effectively optimized key AdaBoost hyperparameters, resulting in improved sensitivity, specificity, precision, F1-score, and overall accuracy. Furthermore, ESFOA-AdaBoost achieved the lowest false positive rate, indicating a greater ability to distinguish fraudulent transactions from legitimate ones while minimizing false alarms.

Comparative results showed that ESFOA-AdaBoost also outperformed SFOA-AdaBoost in terms of convergence speed, generalization capability, and computational efficiency. The elite preservation mechanism enhanced the optimization process by retaining high-quality solutions throughout the search, thereby preventing premature loss of optimal parameter settings. Consequently, the developed ESFOA-AdaBoost model provides a robust, efficient, and scalable framework for real-time financial fraud detection applications.

REFERENCES

1. Abdallah, A., Maarof, M. A., and Zainal, A. (2016). *Fraud detection system: A survey*. Journal of Network and Computer Applications, 68, 90–113.
2. Ahmed, M., and Farouk, A. (2019). *Credit card fraud detection with support vector data description optimized by flower pollination algorithm*. Journal of Financial Crime, 26(3), 789–803.
3. Bose, S., Roy, P., Gupta, R., and Das, S. (2021). *Hybrid fraud detection system for e-commerce transactions using cat swarm optimization*. Expert Systems with Applications, 183, 115324.
4. Cui, Z., Zhang, J., Wang, Y., Cai, X., and Chen, W. (2023). *Starfish optimization algorithm: A novel metaheuristic algorithm for global optimization*. Expert Systems with Applications, 213, 118834.
5. Elaziz, M. A., Oliva, D., Ewees, A. A., and Lu, S. (2023). *Boosting metaheuristic algorithms with elite strategies for global optimization problems*. Applied Soft Computing, 134, 110017.
6. Freund, Y., and Schapire, R. E. (1997). *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of Computer and System Sciences, 55(1), 119–139.
7. Ileberi, E., Sun, Y., and Wang, Z. (2021). *Performance evaluation of machine learning methods for credit card fraud detection using*

SMOTE and AdaBoost. IEEE Access, 9, 165286–165294.

8. Iretiolu Yemisi Alabi, Olufemi Olayanju Awodoye, Stephen Olatunde Olabiyisi, Elijah Olusayo Omidiora, & Zubair Kamaldeen. (2026). A financial fraud detection system using an AdaBoost ensemble optimized by the starfish algorithm. UKR Journal of Multidisciplinary Studies (UKRJMS), 2(6), 186-194.

9. Kumar, R., Singh, P., and Verma, A. (2020). *Loan application fraud detection using deep neural networks optimized by whale optimization algorithm*. Applied Soft Computing, 95, 106491.

10. Patel, R., and Zhao, Y. (2022). *Robust metaheuristic tuning of boosting algorithms*. Proceedings of the 39th International Conference on Machine Learning, 11230–11241.

11. Rahman, M., Islam, M., and Chowdhury, M. (2021). *Insurance claim fraud detection using k-NN with artificial bee colony optimization*. Applied Intelligence, 51(6), 3432–3445.

12. Randhawa, K., Singh, A., Loo, C. K., Seera, M., Lim, C. P., and Nandi, A. K. (2018). *Credit card fraud detection using AdaBoost and majority voting*. IEEE Access, 6, 14277–14284.

13. Hernandez, A., Kim, S., and Park, J. (2024). *Advanced machine learning techniques for financial fraud detection: A review*. Expert Systems with Applications, 235, 121234.

14. Yang, X.-S. (2020). *Nature-inspired optimization algorithms* (2nd ed.). Academic Press. https://doi.org/10.1016/C2019-0-01863-8