ISSN: 2093-5374 / E-ISSN: 2093-5382

A Novel Credit Card Fraud Detection by Outlier Identification and Elimination

V. Suganthi[1*], and Dr.J. Jebathangam[2]

[1*]Research Scholar Department of Computer Science, Vels Institute of Science, Technology & Advanced Studies (VISTAS) Chennai, India. suganthi14.phd@vistas.ac.in, https://orcid.org/0009-0007-6045-5719

[2]Associate Professor, Department of Computer Science, Vels Institute of Science, Technology & Advanced Studies (VISTAS), Chennai, India. jthangam.scs@vistas.ac.in, https://orcid.org/0000-0001-6363-2636

Received: May 13, 2025; Revised: June 28, 2025; Accepted: August 18, 2025; Published: September 30, 2025

Abstract

Generally, credit card fraud refers to the unauthorized use of a credit card for accessing money; it may result in financial losses. However, the existing studies didn't detect credit card fraud based on the transaction pattern similarity of various states. Therefore, this paper proposes SKCMA and LHHS-ASV3N-enabled diverse transaction pattern similarity-aware credit card fraud detection. Primarily, the "Credit Card Fraud Dataset" is taken. Then, the dataset is separated for training (80%) and testing (20%). During training, the Apache Spark is employed to handle the big data. Then, the outliers in the handled big data are identified and eliminated by utilizing EVA. Next, the attributes are extracted from the outlier's eliminated data. Subsequently, the unwanted attributes are reduced by the technique named FG2DA. Thereafter, the data is grouped according to the state by employing SKCMA. Based on the ADASYN, the grouped data is balanced. Lastly, credit card fraud is detected as a fraud transaction and fraudless transaction by using LHHS-ASV3N. During testing, the outlier in the 20% of the data are identified and eliminated. Then, unwanted attributes are reduced by employing FG2DA. Based on LHHS-ASV3N, credit card fraud is detected. The experimental results proved that the proposed technique achieved a high accuracy of 97.6%, thus outperforming the prevailing methods.

Keywords: Adaptive Synthetic Sampling (ADASYN), Credit Card Fraud Detection, Outlier Identification and Elimination, Attribute Reduction, Linear Horse Herd Scaling Based Artificial Support Vector Nodes Neural Network (LHHS-ASV3N), Extreme Value Analysis (EVA), Big Data Handling, and Apache Spark.

1 Introduction

With the fast-growing dependence on Internet technology, many financial institutions have expanded their services by providing business facilities through Internet banking (Cherif et al., 2023). E-payment methods are important in today's competitive financial society. Also, financial institutions make the lives of people convenient by giving credit cards to customers without carrying cash while they go shopping or purchasing anything (Sourabh & Arora, 2022). Probably, all online and offline transactions

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), volume: 16, number: 3 (September), pp. 79-102. DOI: *10.58346/JOWUA.2025.I3.006*

*Corresponding author: Research Scholar Department of Computer Science, Vels Institute of Science, Technology & Advanced Studies (VISTAS) Chennai, India.

are made via credit cards (Patel et al., 2025). Although credit cards give many benefits to consumers, they may create several issues like security and fraud (Itoo et al., 2021). The usage of credit cards increases the number of credit card frauds. Nowadays, many banks, the automobile industry, and financial institutions face credit card fraud issues that happen when the credit card is utilized by unapproved persons to obtain money (Madhurya et al., 2022). Thus, the customers need to verify the transaction with the trader before carrying out any transaction through their credit card. Fraud may occur when a credit card has been stolen, lost, or counterfeited (Yu et al., 2024).

Globally, fraud activities that are related to credit cards are the main cause of financial losses (Kumar et al., 2022). Fraudsters illegally obtain the credit card numbers of users without their knowledge. This credit card fraud also occurs when employing certain software applications, which are controlled by the fraudsters (Asha & Kumar, 2021). Generally, online and offline, card theft, data phishing, application fraud, and telecommunication fraud are the common types of fraudulent activities that happen in credit card fraud (Tanapanichkan et al., 2024). Therefore, detecting credit card fraud is important to avoid financial losses. Recently, many Artificial Intelligence (AI) techniques have been developed to identify credit card fraud (Hemantha Kumar & Senthil Kumar, 2025). In existing studies, Machine Learning (ML) approaches, such as Artificial Neural Network (ANN), eXtreme Gradient Boosting (XGBoost), and Support Vector Machine (SVM) were employed for detecting credit card fraud (Bin Sulaiman et al., 2022; Alarfaj et al., 2022; Harish et al., 2024). Likewise, the Random Forest (RF) and DT algorithms were employed for the fraud detection of credit cards (Dileep et al., 2021).

Similarly, some existing research works utilized hybrid ML techniques with a group search firefly algorithm for detecting credit card fraud activities (Jovanovic et al., 2022). Also, for identifying credit card fraud as normal or fraudulent, Deep Learning (DL) methods, such as Convolutional Neural Network (CNN), simple Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) were employed (Mienye & Jere, 2024). Likewise, certain prevailing works used transfer learning strategies for identifying credit card activities as fraud transactions or fraudless transactions (Lebichot et al., 2021). Additionally, feature extraction techniques like Principal Component Analysis (PCA) and Convolutional AutoEncoder (CAE) methods and a data sampling method named SMOTE were used in existing works for credit card fraud detection (Salekshahrezaee et al., 2023). However, the conventional studies didn't detect credit card fraud regarding transaction pattern similarity of various states. To conquer this shortcoming, an effective model named LHHS-ASV3N and SKCMA-enabled credit card fraud detection is proposed in this article.

1.1 Problem Statement

Conventional credit card fraud detection systems have some limitations, which are listed below,

- The conventional (Karthik et al., 2022) employed a map and reduce of Hadoop Distributed File System (HDFS) for big data handling, which was not efficient for real-time data processing.
- Most of the existing works used uncleaned data for credit card fraud detection, thus increasing the false negative rate.
- Due to the numerous input features in (Esenogho et al., 2022), the predictive modeling task becomes more challenging.
- None of the existing works concentrated on credit card fraud detection based on transaction pattern similarity of various states.
- The prevailing (Ileberi et al., 2022) had imbalanced data, which increased the computation time and reduced the accuracy.

1.2 Objective

The key objectives of the proposed model are described as follows,

- Apache Spark is utilized to handle big data for efficient real-time data processing.
- EVA is used to detect and eliminate the outliers to reduce the false negative rate of credit card fraud detection.
- FG2DA is employed to reduce the unwanted features, thus making the predictive modeling task easier.
- SKCMA is established for grouping the transaction pattern similarity of various states, thus improving the accuracy of credit card fraud detection.
- ADASYN is introduced to balance the imbalanced data to reduce the computation time and improve the accuracy.

The proposed paper is structured as: Section 2 illustrates the literature survey for credit card fraud detection, Section 3 explains the proposed credit card fraud detection framework, Section 4 elucidates the result and discussion, and finally, Section 5 conveys the conclusion of the proposed work along with future scope.

2 Literature Survey

Esenogho et al., 2022 presented a framework named improved credit card fraud detection based on a neural network ensemble. In this research, the LSTM neural network was employed as a base learner in Adaptive Boosting (AdaBoost), and this combination was considered as the ensemble classifier for credit card fraud detection (Sapna Sugandha et al., 2024). Also, the Synthetic Minority Oversampling TEchnique and Edited Nearest Neighbor (SMOTE-ENN) was utilized to perform hybrid resampling. The research proficiently detected the fraud transactions with high sensitivity and specificity. However, in this work, credit card fraud modeling became challenging owing to the greater number of input features.

Ileberi et al., 2021 introduced a model for credit card fraud detection based on ML. Here, the European credit card fraud dataset was employed to assess the research. Also, the optimal features were selected by using a Genetic Algorithm (GA). Then, the fraudulent in the credit card was identified by the ML classifiers, including Decision Tree (DT), Naïve Bayes (NB), Logistic Regression (LR), RF, and ANN. The results demonstrated that the research performed well and achieved high accuracy and Area Under the Curve (AUC). Yet, it had imbalanced data, thus leading to high computation time and low accuracy.

Karthik et al., 2022 suggested behaviour pattern-based credit card fraud identification model. In this work, the boosting and bagging ensemble learning techniques were combined for credit card fraud detection. Here, the original feature space was mapped to the best feature space by using the ensemble feature engineering techniques. For classification, the tree-based learning algorithm was utilized. Hence, the model excellently identified the unseen fraudulent transactions; also, it effectively handled the data imbalance problem. Nevertheless, in this work, the map and reduce of HDFS for big data handling was not efficient in processing the real-time data.

Benchaji et al., 2021 propounded an improved credit card fraud identification model. Here, the most significant features were selected by employing a Swarm intelligence-based approach. Also, the Uniform Manifold Approximation and Projection (UMAP) was established for reducing the

dimensionality of the dataset. Likewise, the data imbalance issue was solved by SMOTE. Based on the sequential modeling of data, this work detected credit card fraud by utilizing an attention mechanism and LSTM deep recurrent neural network. The model superiorly predicted the fraudulent transaction at high accuracy. But, in this work, the missing, erroneous, or non-representative data degraded the performance of the model.

Ileberi et al., 2021 employed ML approaches for credit card fraud detection. In this work, the datasets generated from European credit cardholders were employed to train the model. Also, the SMOTE technique was introduced to re-sample the dataset. Then, ML approaches, such as SVM, LR, RF, XGBoost, DT, and Extra Tree (ET) were used to detect credit card fraud. Next, these ML techniques were combined with AdaBoost for improving classification. Hence, the model proficiently solved the class imbalance problem. But the generated synthetic samples didn't fully capture the real fraudulent behavior.

Khalid et al., 2024 utilized an ensemble ML approach for enhanced credit card fraud detection. By employing under-sampling and the SMOTE technique, the research handled the dataset imbalance problem. Then, the ensemble model, which comprised ML models like SVM, K-Nearest Neighbor (KNN), RF, bagging, and boosting classifiers, was established for credit card fraud detection. The model achieved superior performance, and it aided as a valuable tool against fraudulent transactions. However, the research had interpretability issues; hence, it was difficult to understand how individual predictions were made (Kumar, 2025).

Habibpour et al., 2023 explored uncertainty-aware credit card fraud detection. For evaluating the degree of uncertainty associated with predictions, the three Uncertainty Quantification (UQ) techniques, such as Monte Carlo dropout, ensemble, and ensemble Monte Carlo dropout were employed. The model improved the credit card fraud classification and provided additional insights for decision-makers to improve fraud prevention. Nevertheless, the model had data imbalance problems, thus degrading the efficacy of the research.

Van Belle et al., 2023 established network-enabled credit card fraud identification framework. In this research, credit card fraud was detected by utilizing the Representation Learning (RL) technique through innovative network design, careful downstream classifier configuration, and an effective inductive pooling operator. The model excellently neglected the manual feature engineering and explicitly focused on the relational structure of transactions. However, the model didn't scale well with the increasing volume of transactions, thus hindering the performance of the model.

Mienye & Sun, 2023 developed a model based on data resampling with a DL ensemble for detecting credit card fraud. The classes in the dataset were balanced by using the hybrid SMOTE-ENN technique. Here, the credit card fraud detection model comprised LSTM and GRU as base learners and MultiLayer Perceptron (MLP) as the meta-learner. The research achieved high sensitivity and specificity. But the model might struggle to generalize to unknown patterns unless retrained frequently.

Lin & Jiang, 2021 implemented a credit card fraud identification framework. In this research, the low dimensionality features were extracted by employing the autoencoder. For classifying the data as fraudulent or normal, the technique named probabilistic random forest was established. The model was suitable for imbalanced datasets. Also, the research attained superior performance with respect to accuracy, true positive rate, and Matthew's correlation coefficient. Yet, the model had a lack of interpretability and flexibility issues, thus affecting its efficiency.

3 Proposed Lhhs-Asv3n and Eva-Enabled Credit Card Fraud Detection by Outlier Identification and Elimination Framework

Owing to the increase in internet technologies, credit card fraudulent activities have increased. Therefore, in the proposed framework, the LHHS-ASV3N is established to detect credit card fraud transactions. Also, the proposed FG2DA is introduced for reducing the unwanted attributes. Likewise, the data are grouped according to the state by employing the proposed SKCMA. The diagrammatic layout of the proposed model is depicted in Figure 1.

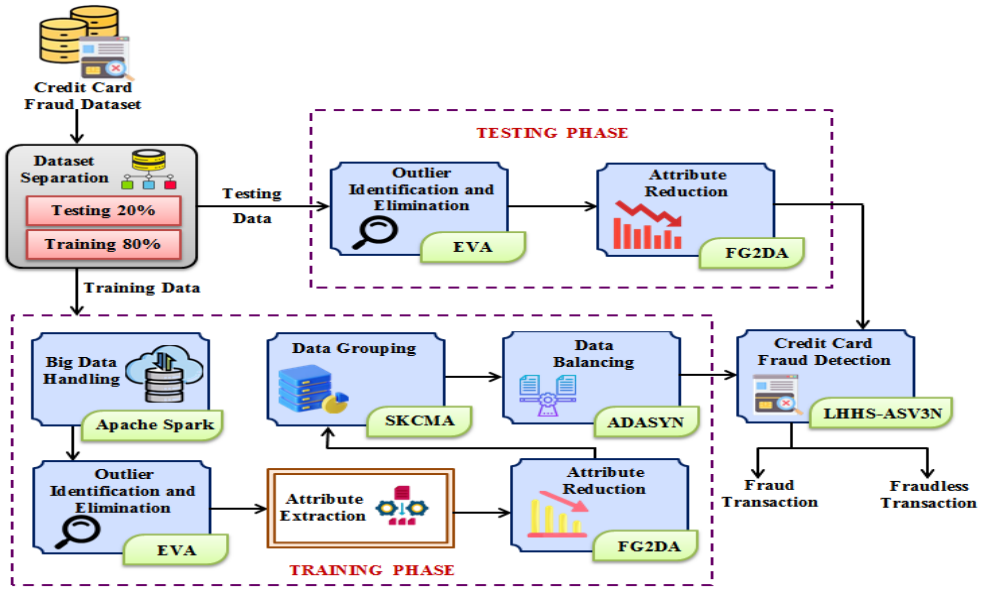


Figure 1: Diagrammatic Layout of the Proposed Credit Card Fraud Detection Framework

Here, significant processes like dataset separation, big data handling, outlier identification and elimination, attribute extraction and reduction, data grouping, data balancing, and credit card fraud detection are performed. The step-by-step process of the proposed model is explained briefly below,

3.1 Credit Card Fraud Detection Dataset

Initially, the "Credit Card Fraud Detection" dataset is taken to train and test the proposed framework. This dataset is collected from publicly available sources, and it consists of credit card transaction details. The total x number of data $\left(\chi_r^{data}\right)$ in the dataset is defined as,

$$\chi_r^{data} \rightarrow \left[\chi_1^{data} + \chi_2^{data} + \chi_3^{data} + \ldots\ldots + \chi_x^{data}\right] \tag{1}$$

Where, χ_1^{data} indicates the first data in the dataset and χ_x^{data} implies the x^{th} data in the dataset.

3.2 Dataset Separation

Next, the $\left(\chi_r^{data}\right)$ are separated into training and testing; here, 80% of the data are employed for training, and the remaining 20% of the data are utilized for testing purposes. Thus, the separated data for training is denoted as T_κ. Also, the separated data for testing is expressed as λ_ν.

In the training phase, the T_κ is processed for detecting credit card fraud, which is described as follows,

3.3 Big Data Handling

Firstly, big data handling is performed due to the large amount of data (T_κ) that creates computational complexities and takes more time for processing. Here, the Apache Spark analytics engine is utilized to handle the big data. In general, Apache Spark efficiently processes the big data quickly with in-memory computation and optimized task execution. It is suitable for large-scale analytics due to its scalability and fault tolerance. The process of Apache Spark is described below,

❄ Apache Spark operates on distributed datasets named Resilient Distributed Datasets (RDDs), which indicates the distributed collection of data. In the first step, the (T_κ) is partitioned. Therefore, RDD $(\Re)$ is equated as,

$$\Re(T_\kappa) \Rightarrow (M_1, M_2, M_3, \ldots, M_\upsilon) \tag{2}$$

Where, M_υ denotes the υ^{th} partition of (T_κ).

❄ Here, RDD offers two significant operations, such as transformation and action. In transformation, a new RDD $(\Re^*)$ is generated by applying a function ∂ to each element of $(\Re)$. It is expressed as,

$$\Re^* = map(\Re, \partial) = \{\partial(M_1), \partial(M_2), \ldots\ldots, \partial(M_\upsilon)\} \tag{3}$$

Here, map specifies the mapping process. Then, the elements that satisfy a predicate $(a(M))$ are selected as follows,

$$\tilde{\Re}^* = filter(\Re, a) = \{M \in \Re \mid a(M) \; is \; true\} \tag{4}$$

Where, $\tilde{\Re}^*$ denotes the filter transformation, M indicates the partitions of (T_κ), and $filter$ represents the filter process. Afterward, the ReduceByKeys operation $(\ddot{\Re}_{reduce})$ applies the reduce function (γ), which groups the elements by a key.

$$\ddot{\Re}_{reduce} \xrightarrow{\tilde{\Re}^*} \{(key, \gamma(Vl_1, Vl_2, \ldots))\} \tag{5}$$

Here, key represents the key, and Vl defines the values for key.

❄ Next, the computations are triggered by the action operation, and the results are returned. Here, the number of elements in the $(\Re)$ is counted (C) as,

$$C = \sum count(\Re) \tag{6}$$

Here, $count$ defines the count process. Lastly, the elements of $(\Re)$ are gathered into the driver. Here, the reduced data are specified as Ψ_φ.

3.4 Outlier Identification and Elimination

Subsequently, the outliers in Ψ_{φ} are identified and eliminated by employing EVA. Outliers are the values that significantly deviate from the majority of values in the data. This outlier changes the normal data into attacked data. EVA ensures robust detection of rare and significant deviations. The process of EVA is derived as follows,

❄ Firstly, the Generalized Extreme Value (GEV) distribution is chosen for modeling the extreme value that consists of parameters like location (l), shape (sh), and scale (sc). It is given as,

$$Pf\left(\Psi_{\varphi};l,sh,sc\right)=\exp\left(-\left(1+sh\frac{\Psi_{\varphi}-l}{sc}\right)^{-\frac{1}{sh}}\right) \quad (7)$$

Here, $\exp$ signifies the exponential function and Pf is the cumulative distribution function, which is employed to fit the extreme values.

❄ Regarding the percentile of the fitted distribution, the threshold (Th) for identifying the outliers is estimated as,

$$Th=l+sc\cdot\left(-\log\left(1-per\right)\right)^{1/sh} \quad (8)$$

Where, per indicates the percentile value and $\log$ defines the logarithmic function. Values that exceed the threshold (Th) are the outliers.

$$OD=\begin{cases} if\ \Psi_{\varphi}>Th & outlier \\ otherwise & no-outlier \end{cases} \quad (9)$$

Where, OD indicates the outlier identification outcomes. Then, the identified outliers are removed from the Ψ_{φ}. Thus, the outliers removed data (O_{rd}) are formulated as,

$$O_{rd}\rightarrow\left[O_1+O_2+O_3+\ldots\ldots+O_m\right] \quad where \quad rd=(1\,to\,m) \quad (10)$$

Where, $rd=(1\,to\,m)$ denotes the number of outliers removed data and O_m signifies the m^{th} outliers removed data.

3.5 Attribute Extraction

From (O_{rd}), the attributes, including cardholder name, fraud Risk customer ID, gender, state, transaction date and place, numIntlTrans, balance, numTrans, and Credit Line are extracted. The extracted attributes (ζ_{ε}) are defined as,

$$\zeta_{\varepsilon}\left(O_{rd}\right)=\left[\zeta_1,\zeta_2,\zeta_3,\ldots\ldots,\zeta_z\right] \quad (11)$$

Here, ζ_z implies the z^{th} extracted attribute, and ζ_1 signifies the first extracted attribute.

3.6 Attribute Reduction

Thereafter, the unwanted attributes are reduced from (ζ_ε) by employing Flexible Gaussian Distribution Discriminant Analysis (FG2DA). Flexible Discriminant Analysis (FDA) superiorly reduces the attributes by capturing non-linear relationships in the data. Also, it aids in retaining the most discriminative features. If the projection matrix is constructed with excessive flexibility, then overfitting issues may happen. To address this problem, the Gaussian distribution is employed in FDA. The mathematical expression of FG2DA is given below,

❄ In the first step, they (ζ_ε) are transformed into numerical scores as follows,

$$OS \xrightarrow{\zeta_\varepsilon} \min_{R,K} \left\| I - RK^{Tp} \right\|_{Fr}^2 \quad (12)$$

Where, OS implies optimal scoring outcomes, I signifies the indicator matrix of class memberships, R specifies the optimal score matrix, K represents the regression coefficient matrix, $\|.\|_{Fr}$ indicates the Frobenius norm, $\min_{R,K}$ elucidates the minimization with respect to R and K, and Tp is the transpose matrix.

❄ For capturing the complex and non-linear relationships, non-linear regression is done. Here, from the (ζ_ε), the optimal scores (R^*) are predicted. It is expressed as,

$$R^* = W(\zeta_\varepsilon) + E \quad (13)$$

Where, W signifies the non-linear regression function, and E implies the error term.

❄ Afterward, Linear Discriminant Analysis (LDA) is applied for the fitted values obtained from non-linear regression. Here, the within-class scatter matrix (Sa_{with}) and between-class scatter matrix (Sa^*_{bet}) are computed. It is written as follows,

$$Sa_{with} = \sum_{\varepsilon=1}^{z} \sum_{R^* \in \zeta_\varepsilon} (R^* - \phi_\varepsilon)(R^* - \phi_\varepsilon)^{Tp} \quad (14)$$

$$Sa^*_{bet} = \sum_{\varepsilon=1}^{z} z(\phi_\varepsilon - \phi)(\phi_\varepsilon - \phi)^{Tp} \quad (15)$$

Where, ϕ_ε indicates the mean vector of R^*, and ϕ denotes the overall mean vector of R^*.

❄ Then, the projection matrix (P) is determined by employing Gaussian distribution in order to solve the overfitting issues. It is equated as,

$$P = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(- \frac{\left((Sa_{with})(Sa^*_{bet}) - N\right)^2}{2\sigma^2} \right) \quad (16)$$

Where, N defines the mean of the distribution, σ^2 is the variance, and $\exp$ demonstrates the exponential function.

❄ Afterward, the attributes are reduced as,

$$A_\tau = \zeta_\varepsilon \cdot P \tag{17}$$

Eventually, the selected attributes are defined as A_τ.

3.7 Data Grouping

Next, the data $(D_\propto)$ from the selected attributes (A_τ) are grouped by using the Spearman K-Correlation Means Algorithm (SKCMA). Here, the $(D_\propto)$ are grouped according to the state. Generally, the K-Means Algorithm (KMA) is highly scalable and can efficiently handle large datasets, and it is computationally efficient. Also, KMA is easy to understand and implement. However, KMA has issues in clustering nodes, where clusters are of varying density. To address this problem, Spearman Correlation is used instead of Euclidean distance, which improves the grouping accuracy. The working of SKCMA is provided as,

Step 1: Primarily, the centroids (i.e., center points of the clusters) are initialized randomly, and it is mathematically expressed as,

$$\delta_c \xrightarrow{D_\propto} (\delta_1, \delta_2, \delta_3, \ldots\ldots, \delta_n) \tag{18}$$

Where, δ_c specifies the initialized centroids, and n denotes the number of initialized centroids.

Step 2: In the next step, the distance between the δ_c and $(D_\propto)$ is calculated based on Spearman Correlation, which is employed to improve the grouping accuracy. It is formulated as,

$$\alpha(\delta_c, D_\propto) = 1 - \frac{6\sum X^2}{n(n^2-1)} \tag{19}$$

Where, α demonstrates the Spearman Correlation, and X implies the difference betwixt the centroid (δ_c) and data $(D_\propto)$.

Step 3: Subsequently, for all data that belong to each cluster, the average is found. Thereafter, the new centroid (Ct) is calculated for each cluster. It is equated as,

$$Ct = \frac{\sum_{c=1,\propto=1}^{n,\Xi} \forall_{c\Xi} D_\propto}{\sum_{c=1,\propto=1}^{n,\Xi} D_\propto} \tag{20}$$

Where, Ξ signifies the number of $(D_\propto)$, and $\forall_{c\Xi}$ depicts the average of each data. Lastly, each node is reassigned to the new closest centroid of each cluster. The above-specified steps are followed until the best clusters are identified. The grouped data (G_h) are indicated as,

$$G_h \xrightarrow{grouped\ data} (G_1, G_2, \ldots\ldots, G_{ab}) \quad Here \quad h = (1,2,\ldots,ab) \tag{21}$$

Where, G_{ab} defines the ab^{th} grouped data. The pseudocode for SKCMA is depicted as follows,

Pseudocode for SKCMA

Input: Data $(D_{\propto})$ from the selected attributes (A_{τ})

Output: Grouped data (G_h)

Begin

Initialize $(D_{\propto})$

For each $(D_{\propto})$

Initialize centroids (δ_c)

Estimate Spearman Correlation

$$\alpha(\delta_c, D_{\propto}) = 1 - \frac{6\sum X^2}{n(n^2-1)}$$

Compute average for all data $(\forall_{c\Xi})$

Discover new centroid

$$Ct = \frac{\sum_{c=1,\propto=1}^{n,\Xi} \forall_{c\Xi} D_{\propto}}{\sum_{c=1,\propto=1}^{n,\Xi} D_{\propto}}$$

Reassign each node to the new closest centroid

End For

Obtain Grouped data (G_h)

End

After grouping the data, the grouped data are balanced to improve the detection accuracy, which is explained in the upcoming section.

3.8 Data Balancing

Subsequently, data balancing is performed on the grouped data (G_h) for balancing the imbalanced data (i.e., the target class has an uneven distribution of observations) for improving the classification accuracy. Here, the ADASYN algorithm is employed for data balancing. ADASYN balances the data by creating synthetic samples for the minority class, thus enhancing the class separability and improving the model's performance. The working procedure of ADASYN is given below,

Step 1: Primarily, the ratio $(Ratio)$ of minority and majority class samples is computed as,

$$Ratio \xrightarrow{G_h} \frac{mi_j}{mj_d} \tag{22}$$

Where, mi_j indicates the number of minority class samples, and mj_d specifies the number of majority class samples.

Step 2: Next, the number of minority and majority classes that need to be generated (U) is determined, and it is written as,

$$U = (\varpi - 1) \times mj_d \tag{23}$$

Where, ϖ signifies the desired ratio of minority to majority class samples.

Step 3: Afterward, based on Euclidean distance, the y nearest neighbors are discovered for each minority class sample. Then, the number of majority class samples among its y nearest neighbors is counted, and the ratio (J) is computed. It is formulated as,

$$J = \frac{\Gamma}{y} \tag{24}$$

Where, Γ indicates the number of majority class neighbors. Subsequently, the ratio is normalized and is given as,

$$n\tilde{r} = \frac{J}{\sum J} \tag{25}$$

Here, $n\tilde{r}$ defines the normalized ratio.

Step 4: After that, the number of synthetic samples per minority class is identified. It is mathematically expressed as,

$$f = n\tilde{r} \times U \tag{26}$$

Where, f defines the number of synthetic samples to generate.

Step 5: Lastly, the f synthetic samples are generated as follows,

$$B_q = G_h + \hbar \times \partial f \tag{27}$$

Here, $\hbar$ indicates the random scalar within the range of [0, 1], and ∂f implies the difference vector (i.e., difference between randomly selected y nearest neighbors and minority class sample). The balanced data (B_q) are represented as,

$$B_q \Rightarrow \langle B_1, B_2, B_3, \ldots\ldots, B_{\kappa'} \rangle \tag{28}$$

Where, $B_{\kappa'}$ signifies the number of balanced data.

3.9 Credit Card Fraud Detection

Lastly, by using a Linear Horse Herd Scaling-based Artificial Support Vector Nodes Neural Network (LHHS-ASV3N), credit card fraud is detected from (B_q). In general, Artificial Neural Network (ANN) has the ability to process multiple tasks simultaneously, thus enhancing computational efficiency. Nevertheless, ANN has overfitting issues, thus leading to poor generalization to new, unseen data. Also, ANN can be complex to develop. To address the overfitting issue, the Horse Herd Optimization (HHO) algorithm is employed. HHO effectively balances exploration and exploitation, thus improving the overall performance. However, HHO may stuck in local optima, thus leading to slower convergence. To improve the velocity updation of the horse, the Linear Scaling technique is included in HHO. Likewise, the complexity of ANN is solved by including Support Vector Nodes that improve the classification accuracy. The structural layout of LHHS-ASV3N is displayed in Figure 2.

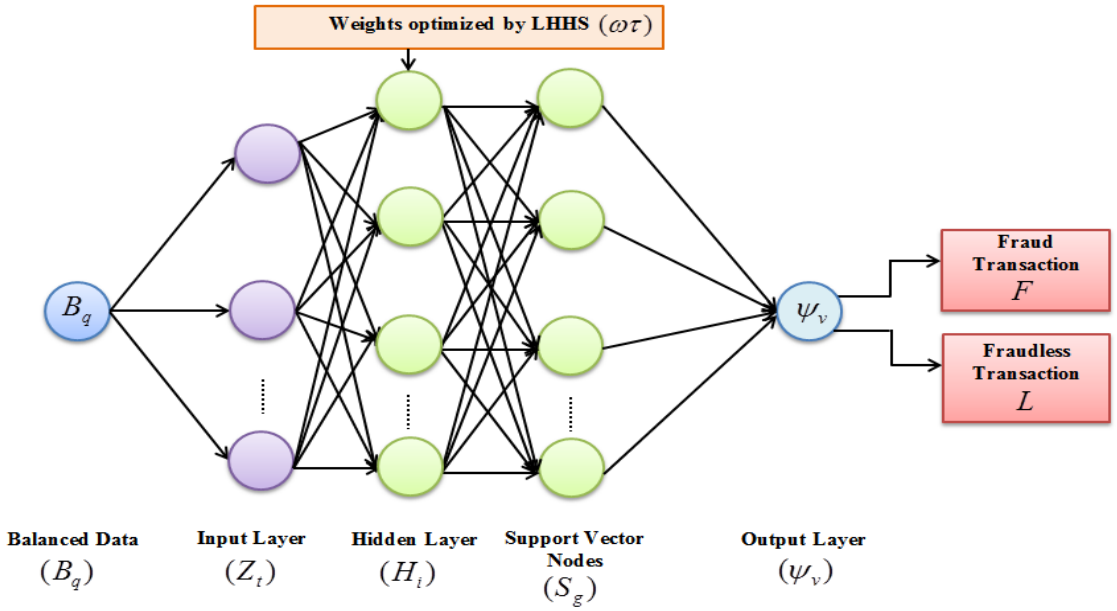


Figure 2: Structural Layout of LHHS-ASV3N

In general, LHHS-ASV3N consists of an input layer, a hidden layer, support vector nodes, and an output layer. The step-by-step process of LHHS-ASV3N is explained as follows,

✓ **Input Layer**

Initially, the balanced data (B_q) are passed through the input layer, which further passes the data to the subsequent layers for processing. The total u number of input layer data (Z_t) is expressed as,

$$Z_t(B_q) \xrightarrow{input\,layer} [Z_1, Z_2, Z_3, \ldots\ldots, Z_u] \quad where \quad t = (1\,to\,u) \qquad (29)$$

Where, Z_1 indicates the first input layer data and Z_u defines the u^{th} input layer data.

✓ **Hidden Layer**

After that, the hidden layer processes Z_t with weight and bias and passes the results through the activation function. Here, the weights are optimized by employing Linear Horse Herd Scaling (LHHS), thus avoiding the overfitting issue. The hidden layer operation (H_i) is equated as,

$$H_i = \xi\left(\sum(\omega\tau \cdot Z_t + b)\right) \qquad (30)$$

Where, ξ indicates a sigmoid activation function, $\omega\tau$ implies weights optimized by LHHS, and b specifies the bias term.

Weight Optimization Process

Here, weight optimization is done by employing LHHS, which effectively balances exploration and exploitation. However, HHO may stuck in local optima, thus leading to slower convergence. Thus, in order to improve the velocity updation of the horse, the Linear Scaling technique is included in HHO.

LHHS mimics the behavioral pattern of horses. Here, the balanced data (B_q) are considered as the initialized population. The most common behavioural patterns of the horse are grazing, hierarchy, sociability, imitation, defence mechanism, and roaming behavior. In each step, the horse's position $(Q_k^{it,ag})$ is equated as,

$$Q_k^{it,ag} = V_k^{it,ag} + Q_k^{(it-1),ag} \qquad ag = \ddot{\alpha}, \ddot{\beta}, \ddot{\gamma}, \ddot{\delta} \qquad (31)$$

Where, $V_k^{it,ag}$ implies the velocity of the vector of that horse, $Q_k^{(it-1),ag}$ specifies the position of the horse in the previous iteration $(it-1)$, it defines the current number of iterations, and $ag = \ddot{\alpha}, \ddot{\beta}, \ddot{\gamma}, \ddot{\delta}$ indicates the range of each horse. Here, the training time (TT) is considered as the fitness function $(\Im it)$. It is equated as,

$$\Im it = TT * \Im it\left(Q_k^{it,ag}\right) \qquad (32)$$

Next, the behavioural patterns of the horse are explained below,

Grazing

Horses graze between 16 and 20 hours per day, with less hours of rest. All the horses do graze in some locations. The grazing behavior of horse is expressed as,

$$Gz_k^{it,ag} = \lambda_k^{it,ag}\left(lb + e * up\right)\left(Q_k^{(it-1)}\right) \qquad (33)$$

$$\lambda_k^{it,ag} = p_\lambda \times \lambda_k^{(it-1),ag} \qquad (34)$$

Where, $Gz_k^{it,ag}$ implies the grazing ability of the horse, lb and up specifies lower and upper bound, respectively, e denotes the arbitrary value, $\lambda_k^{it,ag}$ implies the grazing co-efficient of the horse at the current iteration, $\lambda_k^{(it-1),ag}$ denotes the grazing co-efficient of the horse at the previous iteration, and p_λ implies the reducing factor.

Hierarchy

In general, horses live their lives by following a leader, which is regarded as the coefficient hr_k. The hierarchy of horses $\left(Hch_k^{it,ag}\right)$ is indicated as,

$$Hch_k^{it,ag} = hr_k^{it,ag}\left(Q_{\Im it}^{(it-1)} - Q_k^{(it-1)}\right) \qquad (35)$$

$$hr_k^{it,ag} = hr_k^{(-1+it),ag} \times p_{hr} \qquad (36)$$

Where, $Q_{\Im it}^{(it-1)}$ demonstrates the position of the best horse, $hr_k^{it,ag}$ specifies the hierarchy co-efficient of the horse at the current iteration, $hr_k^{(-1+it),ag}$ indicates the hierarchy co-efficient of the horse at the previous iteration, and p_{hr} defines the reducing factor.

Sociability

Horses require social interaction and may coexist with other animals. This sociability is indicated by elements as a movement toward the average location of other horses. Horses interested in being herd is defined as,

$$Slo_k^{it,ag} = slo_k^{it,ag}\left[\left(\frac{1}{\beta}\sum_{s=1}^{\beta} Q_s^{(-1+it)}\right) - Q_k^{(-1+it)}\right] \quad (37)$$

$$slo_k^{it,ag} = slo_k^{(-1+it),ag} \times p_{slo} \quad (38)$$

Where, $Slo_k^{it,ag}$ implies the social motion vector of the horse, $slo_k^{it,ag}$ specifies the orientation of that horse, $slo_k^{(-1+it),ag}$ demonstrates the orientation of the horse at the previous iteration, p_{slo} is the reducing factor, $Q_s^{(-1+it)}$ signifies the horses at the previous iteration, and β depicts the number of horses.

Imitation

Likewise, horses copy the other horses' positive and negative habits like locating the best grazing spot. The imitation behavior of horses $\left(Ita_k^{it,ag}\right)$ is expressed as,

$$Ita_k^{it,ag} = ita_k^{it,ag}\left[\left(\frac{1}{w\beta}\sum_{s=1}^{w\beta} Q_s^{(-1+it)}\right) - Q^{(-1+it)}\right] \quad (39)$$

$$ita_k^{it,ag} = ita_k^{(-1+it),ag} \times p_{ita} \quad (40)$$

Where, $w\beta$ demonstrates the number of horses in the best location, $ita_k^{it,ag}$ indicates the imitation co-efficient of the horse at the current iteration, $ita_k^{(-1+it),ag}$ indicates the imitation co-efficient of the horse at the previous iteration, and p_{ita} denotes the reduction factor.

Defence Mechanism

Horses employ fight-or-flight reactions to defend themselves. It works by fleeing away from the horses who reveal improper behaviors. The defence mechanism keeps the horse away from dangerous situations. It is given as,

$$DM_k^{it,ag} = dm_k^{it,ag}\left[\left(\frac{1}{q\beta}\sum_{s=1}^{q\beta} Q_s^{(-1+it)}\right) - Q^{(-1+it)}\right] \quad (41)$$

$$dm_k^{it,ag} = dm_k^{(-1+it),ag} \times p_{dm} \quad (42)$$

Where, $DM_k^{it,ag}$ represents the escape vector of the horse, $dm_k^{it,ag}$ defines the defense co-efficient of the horse at the current iteration, $dm_k^{(-1+it),ag}$ illustrates the defense co-efficient of the horse at the previous iteration, $q\beta$ denotes the number of horses in the worst position, and p_{dm} is the reduction factor.

Roaming Behavior

Horses roam from one pasture to another and get to know their surroundings. The random movement of a horse in the herd is known as the roaming behavior. It is determined as,

$$Rot_k^{it,ag} = rot_k^{it,ag} * Q^{(-1+it)} \quad (43)$$

$$rot_k^{it,ag} = rot_k^{(-1+it),age} \times p_{rot} \quad (44)$$

Where, $Rot_k^{it,ag}$ indicates the roaming behavior vector of the horse, $rot_k^{it,ag}$ represents the roaming co-efficient of the horse at the current iteration, $rot_k^{(-1+it),age}$ denotes the roaming co-efficient of the horse at the previous iteration, and p_{rot} is the reduction factor. Here, the reduction factor p is computed by the linear scaling formula. It is given as,

$$p = \frac{Q_k^{it,ag} - \Im it_{\max}\left(Q_k^{it,ag}\right)}{\Im it_{\max}\left(Q_k^{it,ag}\right) - \Im it_{\min}\left(Q_k^{it,ag}\right)} \quad (45)$$

Here, $\Im it_{\max}$ and $\Im it_{\min}$ indicate the maximum and minimum fitness function, respectively. Based on the above p, the reduction factors in 6 behavioral patterns of horses are updated. Then, by applying these behavioral patterns of horses in velocity, the updated velocity $V_k^{it,ag}$ of horses is obtained. The optimized weights are indicated as $\omega\tau$.

- ✓ **Support Vector Nodes**

Thereafter, the outcomes of (H_i) are subjected to the support vector nodes; they are used to reduce the complexity of the model, thus improving the classification accuracy. It is denoted as (S_g).

- ✓ **Output layer**

Then, the output layer (ψ_v) delivers the detection outcomes. Lastly, the detection outcomes of credit card fraud (Φ) are defined as,

$$\Phi \rightarrow (F, L) \quad (46)$$

Here, F indicates a fraud transaction, and L signifies a fraudless transaction. The pseudocode for LHHS-ASV3N is given below,

Pseudocode for LHHS-ASV3N

Input: Balanced Data (B_q)

Output: Credit Card Fraud Detection Outcomes (Φ)

Begin

Initialize (B_q)

For each (B_q)

Find input layer (Z_t)

Estimate hidden layer

$$H_i = \xi\left(\sum\left(\omega\tau^* \cdot Z_t + b\right)\right)$$

Update weights by LHHS

$\omega\tau$

Perform Support Vector Nodes (S_g)

Compute output layer (ψ_v)

End For

Obtain $\Phi \rightarrow (F, L)$

End

In the testing phase, the λ_v is given for credit card fraud detection. Here, firstly, the outliers in λ_v are identified and eliminated by employing the EVA technique. The process of EVA is explained in Section 3.4. After that, the attributes are reduced from the outliers eliminated data $(Ou_{\varepsilon\ell})$ based on FG2DA. The working of FG2DA is described in Section 3.6. Lastly, based on the reduced attributes $(\Re\varepsilon_{att})$, credit card fraud is detected as fraud or fraudless transactions by using LHHS-ASV3N, which is explained above. Thus, the proposed framework excellently detected credit card fraud.

4 Result and Discussion

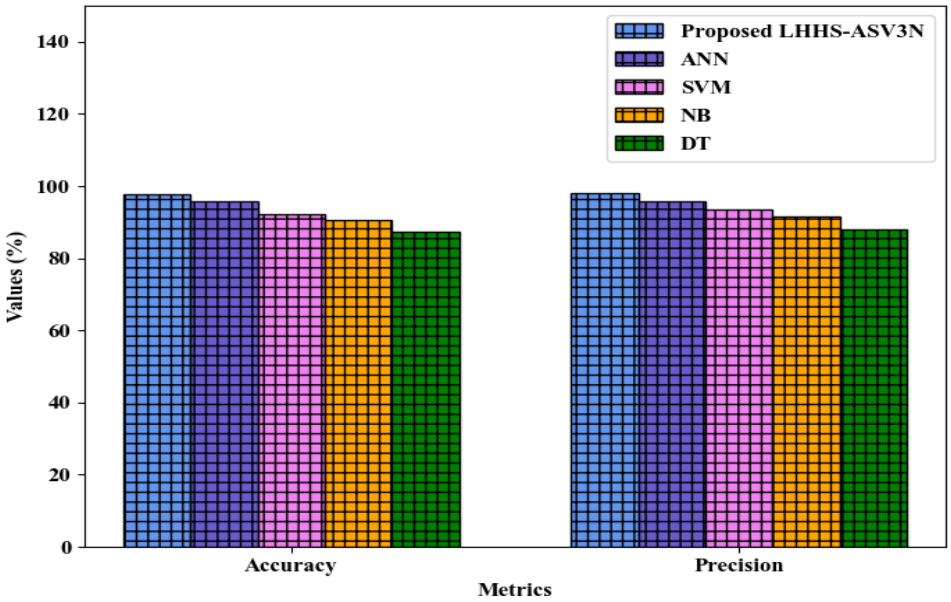
In the result and discussion section, the performance assessment and comparative analysis of the proposed and prevailing techniques are carried out to demonstrate the proposed model's reliability. Also, the proposed model is implemented in the working platform of PYTHON.

4.1 Dataset Description

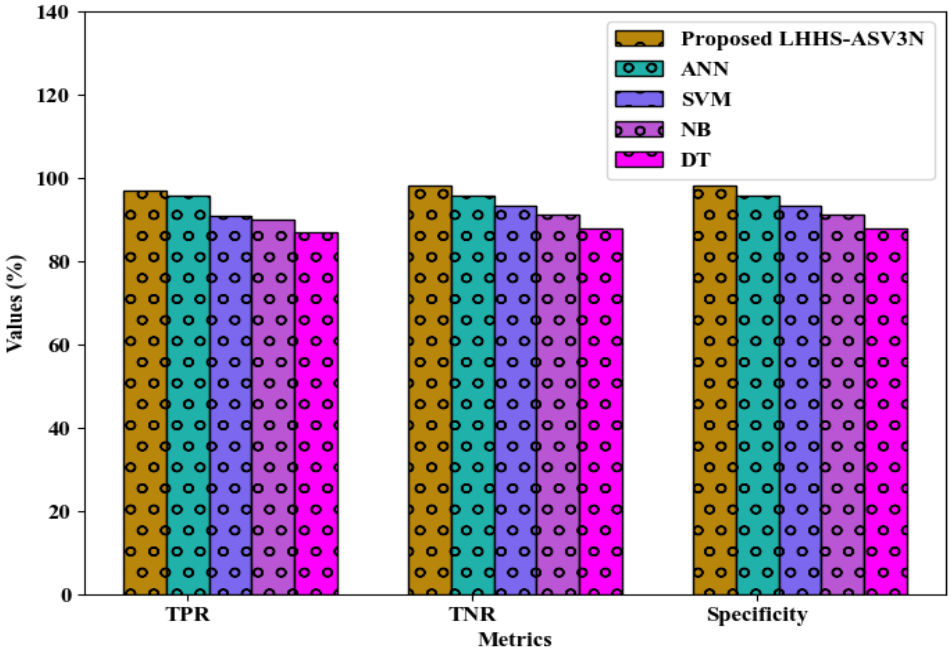
Here, the Credit Card Fraud (ccFraud) dataset from Revolution Analytics is employed to assess the proposed framework. This dataset is collected from publicly available sources, and the dataset link is provided under the reference section. Also, this dataset consists of 1048575 credit card holders' details. From the whole data, 80% of the data (i.e., 838860) is employed for training, and the remaining 20% of the data (i.e., 209715) is used for testing purposes.

4.2 Performance Assessment

In this section, the performance validation of the proposed and existing techniques is done to demonstrate the proposed model's effectiveness.



(a)



(b)

Figure 3: Graphical Representation Regarding (a) Accuracy, Precision (b) TNR, TPR, Specificity

Figures 3 (a) and (b) depict the graphical representation of the proposed LHHS-ASV3N and existing techniques in terms of accuracy, precision, True Negative Rate (TNR), True Positive Rate (TPR), and specificity. Here, the proposed LHHS-ASV3N achieved a high accuracy, precision, TNR, TPR, and specificity of 97.6%, 98.21%, 98.16%, 97.05%, and 98.16, respectively. However, the existing techniques like ANN, SVM, Naïve Bayes (NB), and Decision Tree (DT) obtained a low average accuracy, precision, TNR, TPR, and specificity of 91.48%, 92.19%, 92.02%, 90.96%, and 92.02%, respectively. Here, the Support Vector Hidden Nodes and Linear Horse Herd Scaling techniques were modified with ANN for improving the accuracy of credit card fraud detection. Thus, the effectiveness of the proposed model was proved.

Table 1: Performance Validation Based on Performance Metrics

Techniques	Recall (%)	F1-score (%)	Training time (ms)
Proposed LHHS-ASV3N	97.05	97.63	13000.66
ANN	95.80	95.84	17000.32
SVM	90.98	92.15	24000.07
NB	90	90.71	28006.39
DT	87.05	87.57	31999.67

Table 1 depicts the performance validation of the proposed LHHS-ASV3N and prevailing methods based on performance metrics. Here, the proposed LHHS-ASV3N employs Linear Horse Herd Scaling for solving the overfitting issue, thus improving the model's performance. The proposed LHHS-ASV3N achieved a high recall and F1-score of 97.05% and 97.63%, correspondingly. It also obtained a low training time of 13000.66 milliseconds (ms). Likewise, the prevailing SVM attained a low precision of 90.98%, a low recall of 92.15%, and a high training time of 24000.07ms. Also, the existing DT attained a low precision of 87.05%, a low recall of 87.57%, and a high training time of 31999.67ms. Similarly, the prevailing techniques like ANN and NB attained poor performance. Thus, the efficacy and trustworthiness of the proposed model were proved.

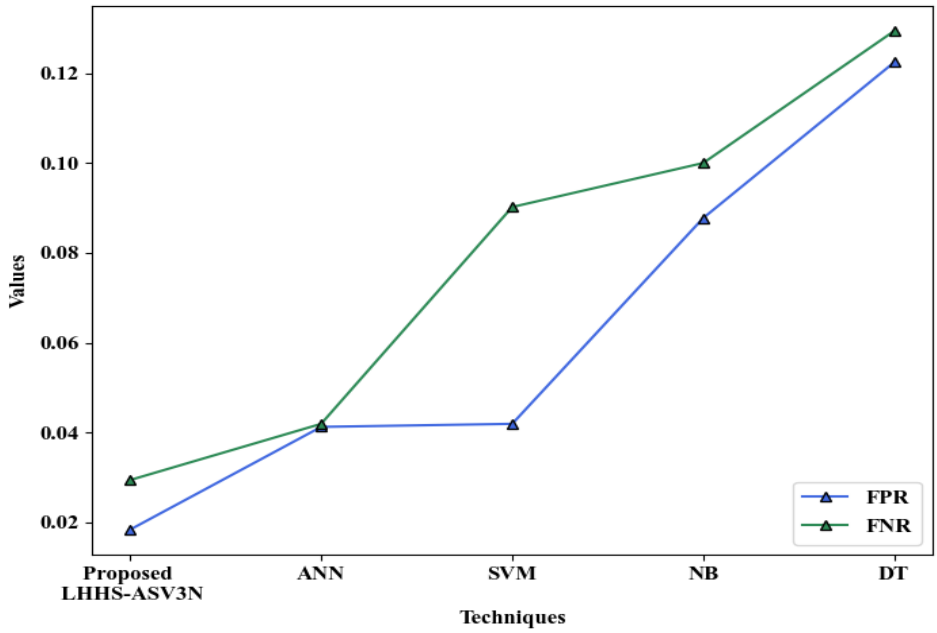


Figure 4: FNR and FPR Evaluation

FNR and FPR evaluation of the proposed LHHS-ASV3N and prevailing techniques are shown in Figure 4. Here, the proposed LHHS-ASV3N attained a low False Negative Rate (FNR) of 0.02941 and a low False Positive Rate (FPR) of 0.01836. But, the existing techniques, including ANN, SVM, NB, and DT obtained a high FNR of 0.04192, 0.09019, 0.1, and 0.12941, respectively. Also, the existing methods obtained high FPR values. Here, the proposed technique had less error than the existing techniques due to the usage of Support Vector Hidden Nodes and Linear Horse Herd Scaling.

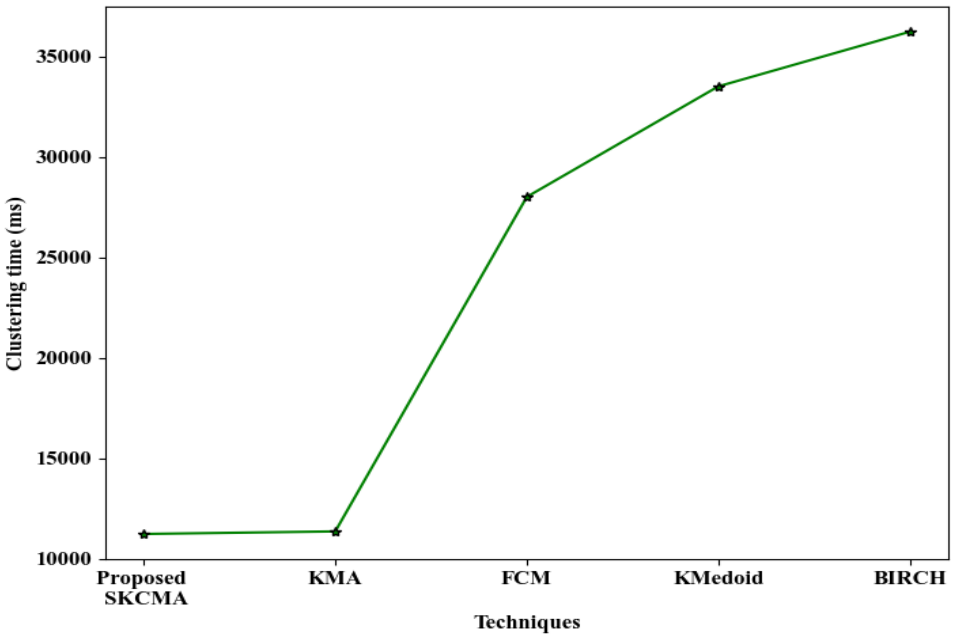


Figure 5: Clustering Time Analysis

The Spearman Correlation technique is employed in KMA instead of Euclidean distance for clustering varying density of clusters, thus improving the grouping accuracy. The clustering time analysis of the proposed SKCMA and prevailing techniques is displayed in Figure 5. Here, the proposed SKCMA took less time of 11230.101ms for clustering the data. Also, the prevailing techniques like KMA, Fuzzy C-Means (FCM), KMedoid, and Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) took a maximum time of 11357.217ms, 28000.469ms, 33499.903ms, and 36220.267ms for clustering. Thus, the results proved that the proposed SKCMA reduced the time complexity than the prevailing techniques.

Table 2: Comparative Evaluation with Respect to Dunn Index and Silhouette Score

Methods	Dunn Index	Silhouette Score
Proposed SKCMA	1.728441	0.984384
KMA	1.32753	0.952647
FCM	1.092921	0.936348
KMedoid	0.968534	0.909709
BIRCH	0.525122	0.864054

Table 2 depicts the comparative evaluation of the proposed SKCMA and prevailing methods in terms of the Dunn Index and Silhouette Score. Here, the proposed SKCMA achieved a high Dunn Index of 1.728441 and a high Silhouette Score of 0.984384. But, the prevailing BIRCH and FCM attained a low Dunn Index of 0.525122 and 1.092921, respectively. Likewise, the existing KMA and KMedoid obtained a low Silhouette Score of 0.952647 and 0.909709, correspondingly. Here, the Spearman Correlation was used by the proposed SKCMA to effectively group the data regarding transaction pattern similarity of various states.

Table 3: Fitness vs Iteration Validation

Methods	Fitness/Iteration				
	10	20	30	40	50
Proposed LHHS	15.581	13.076	13.076	11.488	5.775
HHO	14.08	8.488	5.444	1.493	1.493
SMO	13.644	7.372	4.345	1.108	1.076
WOA	12	6.3617	2.2086	1.1085	0.5375
PSO	11.19	5.653	1.315	1.064	0.378

Fitness vs Iteration validation of the proposed LHHS and existing techniques, such as HHO, Spider Monkey Optimization (SMO), Whale Optimization Algorithm (WOA), and Particle Swarm Optimization (PSO) is shown in Table 3. Here, the proposed LHHS achieved a high fitness of 15.581 for the 10th iteration and 5.775 for the 50th iteration. Likewise, the prevailing HHO, SMO, WHO, and PSO attained a low fitness of 14.08, 13.644, 12, and 11.19 for the 10th iteration. Also, the prevailing techniques attained low fitness for all iterations. Here, the proposed LHHS excellently optimized the weights with high fitness due to the usage of the Linear Scaling technique.

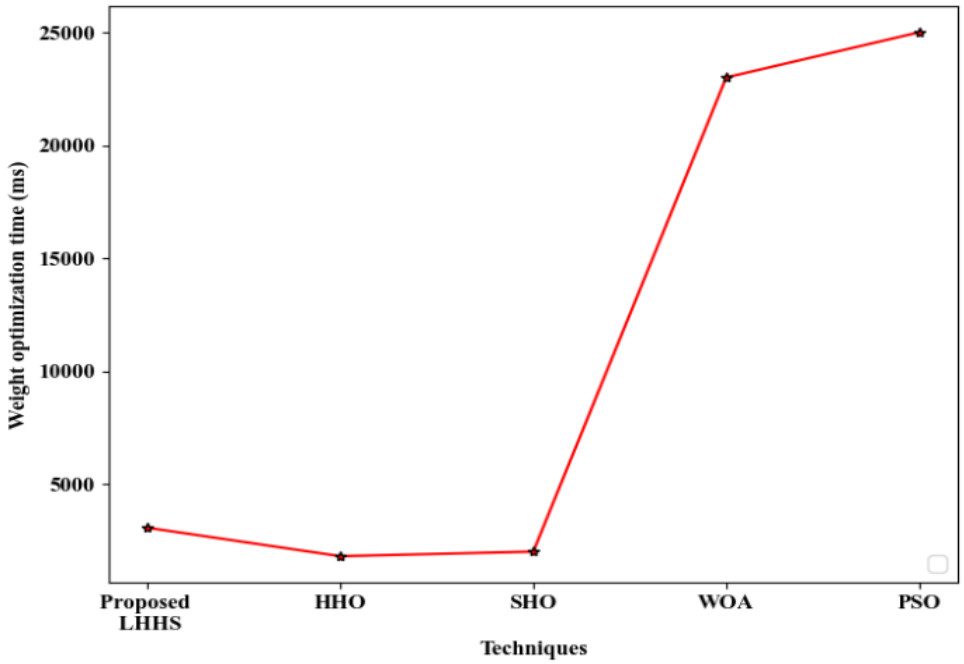


Figure 6: Graphical Analysis Regarding Weight Optimization Time

The graphical analysis of the proposed LHHS and existing techniques regarding weight optimization time is displayed in Figure 6. Here, the Linear Scaling technique is modified with HHO for faster convergence, thus improving the velocity updation of the horse. The proposed LHHS took a less time of 3065.0905ms for weight optimization, whereas the existing HHO, SMO, WOA, and PSO took a maximum time of 1819.0606ms, 2049.5371ms, 24999.9873ms, and 22999.3202ms for weight optimization. Thus, the effectiveness of the proposed model was proved.

4.3 Comparative Analysis

Here, the outcomes of the proposed work and related works are compared to demonstrate the reliability of the proposed model.

Table 4: Comparative Assessment

Authors' name	Techniques	Datasets	Recall (%)	F1-Score (%)
Proposed Model	LHHS-ASV3N	ccFraud dataset	97.05	97.63
(Prusti et al., 2021)	Supervised ML techniques like DT, RF, KNN, MLP, and SVM and unsupervised ML techniques like LOF and IF	BankSim dataset	83.12	83.41
(Alfaiz & Fati, 2022)	AllKNN-CatBoost	Credit card fraud detection dataset	95.91	87.40
(Malik et al., 2022)	Adaboost + LGBM	Credit card fraud detection dataset obtained from Vesta Corporation	64	77
(Ahmad et al., 2023)	Hybrid ML	Credit card fraud detection dataset, where real transactions collected from European cardholders	-	93
(Mniai et al., 2023)	SVDD	Credit card fraud detection dataset	97	93

Table 4 depicts the comparative assessment of the proposed and related works. Here, the proposed LHHS-ASV3N achieved a high recall and F1-score of 97.05% and 97.63%, respectively due to the usage of Support Vector Hidden Nodes and Linear Horse Herd Scaling. However, the prevailing supervised ML techniques, such as DT, RF, KNN, MLP, and SVM, and unsupervised ML methods like Local Outlier Factor (LOF) and Isolation Forest (IF) attained a low recall of 83.12% and low F1-score of 83.41%. Likewise, the prevailing AdaBoost + Light Gradient Boosting Machine (AdaBoost + LGBM) obtained a very low recall and F1-score of 64% and 77%, correspondingly. Similarly, the existing All KNN undersampling technique with Category Boosting (AllKNN-CatBoost), Hybrid ML techniques like LR, NB, KNN, and ANN, and Support Vector Data Description (SVDD) obtained poor performance owing to poor generalization and interpretability issues. Thus, the effectiveness of the proposed model was proved.

5 Conclusion

This paper presents a proficient model named diverse transaction pattern similarity aware credit card fraud detection using SKCMA and LHHS-ASV3N. Here, significant processes, such as big data handling, outlier identification and elimination, variable extraction and reduction, data grouping, data balancing, and credit card fraud detection were performed. At last, the proposed LHHS-ASV3N achieved a high accuracy, precision, and specificity of 97.6%, 98.21%, and 98.16%, respectively in credit card fraud detection, which demonstrated the efficacy of the proposed model. Likewise, the proposed SKCMA-based data grouping took a less clustering time of 11230.101ms and a high Dunn Index of 1.728441, which showed the low time complexity of the proposed model. Also, the proposed LHHS took less time of 3065.0905ms for weight optimization, which was lower than the existing techniques. Overall, the proposed model improved trustworthiness and reliability. Although the proposed model considered the customer's transaction details, it failed to analyze the behavior of customers.

Future Enhancement

In the future, enhanced approaches will be introduced to consider the transaction behavior of customers along with transaction details for credit card fraud detection to further improve accuracy.

References

[1] Ahmad, H., Kasasbeh, B., Aldabaybah, B., & Rawashdeh, E. (2023). Class balancing framework for credit card fraud detection based on clustering and similarity-based selection (SBS). *International Journal of Information Technology*, *15*(1), 325-333. https://doi.org/10.1007/s41870-022-00987-w

[2] Alfaiz, N. S., & Fati, S. M. (2022). Enhanced credit card fraud detection model using machine learning. *Electronics*, *11*(4), 662. https://doi.org/10.3390/electronics11040662

[3] Alarfaj, F. K., Malik, I., Khan, H. U., Almusallam, N., Ramzan, M., & Ahmed, M. (2022). Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms. *Ieee Access*, *10*, 39700-39715. https://doi.org/10.1109/ACCESS.2022.3166891.

[4] Asha, R. B., & KR, S. K. (2021). Credit card fraud detection using artificial neural network. *Global Transitions Proceedings*, *2*(1), 35-41. https://doi.org/10.1016/j.gltp.2021.01.006

[5] Benchaji, I., Douzi, S., El Ouahidi, B., & Jaafari, J. (2021). Enhanced credit card fraud detection based on attention mechanism and LSTM deep model. *Journal of Big Data*, *8*(1), 151. https://doi.org/10.1186/s40537-021-00541-8

[6] Bin Sulaiman, R., Schetinin, V., & Sant, P. (2022). Review of machine learning approach on credit card fraud detection. *Human-Centric Intelligent Systems*, *2*(1), 55-68. https://doi.org/10.1007/s44230-022-00004-0

[7] Cherif, A., Badhib, A., Ammar, H., Alshehri, S., Kalkatawi, M., & Imine, A. (2023). Credit card fraud detection in the era of disruptive technologies: A systematic review. *Journal of King Saud University-Computer and Information Sciences*, *35*(1), 145-174. https://doi.org/10.1016/j.jksuci.2022.11.008

[8] Dileep, M. R., Navaneeth, A. V., & Abhishek, M. (2021, February). A novel approach for credit card fraud detection using decision tree and random forest algorithms. In *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (pp. 1025-1028). IEEE. https://doi.org/10.1109/ICICV50876.2021.9388431

[9] Esenogho, E., Mienye, I. D., Swart, T. G., Aruleba, K., & Obaido, G. (2022). A neural network ensemble with feature engineering for improved credit card fraud detection. *IEEE access*, *10*, 16400-16407. https://doi.org/10.1109/ACCESS.2022.3148298

[10] Habibpour, M., Gharoun, H., Mehdipour, M., Tajally, A., Asgharnezhad, H., Shamsi, A., ... & Nahavandi, S. (2023). Uncertainty-aware credit card fraud detection using deep learning. *Engineering Applications of Artificial Intelligence*, *123*, 106248. https://doi.org/10.1016/j.engappai.2023.106248

[11] Harish, S., Krishna, R. V. V., Satyanarayana, V., & Pattanaik, B. C. (2024). Quantum Computing and Artificial Intelligence in Materials Discovery for Batteries. In *Real-World Challenges in Quantum Electronics and Machine Computing* (pp. 211-223). IGI Global. https://doi.org/10.4018/979-8-3693-4001-1.ch015

[12] Ileberi, E., Sun, Y., & Wang, Z. (2021). Performance evaluation of machine learning methods for credit card fraud detection using SMOTE and AdaBoost. *IEEE access*, *9*, 165286-165294. https://doi.org/10.1109/ACCESS.2021.3134330

[13] Ileberi, E., Sun, Y., & Wang, Z. (2022). A machine learning based credit card fraud detection using the GA algorithm for feature selection. *Journal of Big Data*, *9*(1), 24. https://doi.org/10.1186/s40537-022-00573-8

[14] Itoo, F., Meenakshi, & Singh, S. (2021). Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection. *International Journal of Information Technology*, *13*(4), 1503-1511. https://doi.org/10.1007/s41870-020-00430-y

[15] Jovanovic, D., Antonijevic, M., Stankovic, M., Zivkovic, M., Tanaskovic, M., & Bacanin, N. (2022). Tuning machine learning models using a group search firefly algorithm for credit card fraud detection. *Mathematics*, *10*(13), 2272. https://doi.org/10.3390/math10132272

[16] Karthik, V. S. S., Mishra, A., & Reddy, U. S. (2022). Credit card fraud detection by modelling behaviour pattern using hybrid ensemble model. *Arabian Journal for Science and Engineering*, *47*(2), 1987-1997. https://doi.org/10.1007/s13369-021-06147-9

[17] Khalid, A. R., Owoh, N., Uthmani, O., Ashawa, M., Osamor, J., & Adejoh, J. (2024). Enhancing credit card fraud detection: an ensemble machine learning approach. *Big Data and Cognitive Computing*, *8*(1), 6. https://doi.org/10.1109/ICKECS61492.2024.10616809

[18] Kumar, H. (2025). AI and Quantum Network Applications in Business and Medicine Machine Learning-Based Classification and Prediction of DDoS Attacks Using Naive Bayes. In *AI and Quantum Network Applications in Business and Medicine* (pp. 371-382). IGI Global Scientific Publishing. https://doi.org/10.4018/979-8-3693-8135-9.ch022

[19] Kumar, S., Gunjan, V. K., Ansari, M. D., & Pathak, R. (2022). Credit card fraud detection using support vector machine. In V. K. Gunjan & J. M. Zurada (Eds.), *Proceedings of the 2nd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications* (Vol. 237, pp. [27-37]). Singapore: Springer. https://doi.org/10.1007/978-981-16-6407-6_3

[20] Lebichot, B., Verhelst, T., Le Borgne, Y. A., He-Guelton, L., Oble, F., & Bontempi, G. (2021). Transfer Learning Strategies for Credit Card Fraud Detection. *IEEE Access*, *9*, 114754–114766. https://doi.org/10.1109/ACCESS.2021.3104472

[21] Lin, T. H., & Jiang, J. R. (2021). Credit card fraud detection with autoencoder and probabilistic random forest. *Mathematics*, *9*(21), 2683.https://doi.org/10.3390/math9212683

[22] Madhurya, M. J., Gururaj, H. L., Soundarya, B. C., Vidyashree, K. P., & Rajendra, A. B. (2022). Exploratory analysis of credit card fraud detection using machine learning techniques. *Global Transitions Proceedings*, *3*(1), 31-37.https://doi.org/10.1016/j.gltp.2022.04.006

[23] Malik, E. F., Khaw, K. W., Belaton, B., Wong, W. P., & Chew, X. (2022). Credit card fraud detection using a new hybrid machine learning architecture. *Mathematics*, *10*(9), 1480. https://doi.org/10.3390/math10091480

[24] Mienye, I. D., & Jere, N. (2024). Deep learning for credit card fraud detection: A review of algorithms, challenges, and solutions. *IEEE Access*. *12*, 96893-96910. https://doi.org/10.1109/ACCESS.2024.3426955

[25] Mienye, I. D., & Sun, Y. (2023). A deep learning ensemble with data resampling for credit card fraud detection. *Ieee Access*, *11*, 30628-30638.https://doi.org/10.1109/ACCESS.2023.3262020

[26] Mniai, A., Tarik, M., & Jebari, K. (2023). A novel framework for credit card fraud detection. *IEEE Access*, *11*, 112776-112786. https://doi.org/10.1109/ACCESS.2023.3323842

[27] Patel, A., Patel, M. M., & Patel, P. S. (2025). Enhancing Credit Card Security Using Supervised Machine Learning Approach for Intelligent Fraud Detection. In *Advancing Cyber Security Through Quantum Cryptography* (pp. 397-412). IGI Global. https://doi.org/10.4018/979-8-3693-5961-7.ch014

[28] Prusti, D., Das, D., & Rath, S. K. (2021). Credit card fraud detection technique by applying graph database model. *Arabian Journal for Science and Engineering*, *46*(9), 1-20. https://doi.org/10.1007/s13369-021-05682-9

[29] Salekshahrezaee, Z., Leevy, J. L., & Khoshgoftaar, T. M. (2023). The effect of feature extraction and data sampling on credit card fraud detection. *Journal of Big Data*, *10*(1), 6. https://doi.org/10.1186/s40537-023-00684-w

[30] Sourabh, Arora, B. (2022). A review of credit card fraud detection techniques. In P. K. Singh, Y. Singh, M. H. Kolekar, A. K. Kar, & P. J. S. Gonçalves (Eds.), *Recent innovations in computing* (Vol. 832, pp. [1-9]). Springer. https://doi.org/10.1007/978-981-16-8248-3_40

[31] Sugandha, S., Choubey, R. R., Singh, A., & Suman, S. (2024). Adapting Talent Strategy in the Gig Economy to Incorporate the Impact of Quantum Computing on Work Evolution. In *Real-World Challenges in Quantum Electronics and Machine Computing* (pp. 1-13). IGI Global. https://doi.org/10.4018/979-8-3693-4001-1.ch001

[32] Tanapanichkan, S., Kosolsombat, S., & Luangwiriya, T. (2024). Credit card fraud detection using machine learning. In *Proceedings of the 2024 IEEE International Conference on Cybernetics and Innovations (ICCI)* (pp. 1–5). IEEE. https://doi.org/10.1109/ICCI60780.2024.10532670

[33] Van Belle, R., Baesens, B., & De Weerdt, J. (2023). CATCHM: A novel network-based credit card fraud detection method using node representation learning. *Decision Support Systems, 164*, 113866. https://doi.org/10.1016/j.dss.2022.113866

[34] Yu, C., Xu, Y., Cao, J., Zhang, Y., Jin, Y., & Zhu, M. (2024, August). Credit card fraud detection using advanced transformer model. In *2024 IEEE International Conference on Metaverse Computing, Networking, and Applications (MetaCom)* (pp. 343-350). IEEE. https://doi.org/10.1109/MetaCom62920.2024.00064.

Authors Biography

V. Suganthi received her M.C.A Degree from the Department of Faculty of Information and Communication Engineering at College of Engineering Guindy, Anna University, Chennai, India, in 2010. She is currently an assistant professor at S.S.K.V College of Arts and Science for women, Kanchipuram, Chennai, India. She is pursuing her Ph.D degree in VELS Institute of Science, Technology and Advanced Studies (VISTAS). Her current research interest includes Big Data analytic, Data Mining, and Machine Learning.

Dr.J. Jebathangam received her Ph.D degree from the Department of Computer Science at Mother Teresa University, India, in 2019. She is currently an Associate professor in the School of Computing Sciences at VELS Institute of Science, Technology and Advanced Studies (VISTAS), Chennai, India. Her research interest includes Image Processing and Machine Learning.