Journal of Autonomous Intelligence (2024) Volume 7 Issue 5
doi: 10.32629/jai.v7i5.1568

ORIGINAL RESEARCH ARTICLE

Hybrid resampling technique with HWSO based temporal convolution network for credit card fraud detection

S. Abijah Roseline[1], Rakesh Chandrashekar[2], Jothi Prabha Appadurai[3], D Sudha[4], D Anu Disney[4], Balasubramanian Prabhu kavin[5,*], Gan Hong Seng[6]

[1] *Department of Computational Intelligence, SRM Institute of Science and Technology, Kattankulathur 603203, India*

[2] *Department of Mechanical Engineering, New Horizon College of Engineering, Bangalore 560103, India*

[3] *Department of Computer Science and Engineering (Networks), Kakatiya Institute of Technology & Science, Warangal 506015, India*

[4] *School of Computing, Sathyabama Institute of Science and Technology, Chennai 600119, India*

[5] *Department of Data Science and Business System, SRM Institute of Science and Technology, Kattankulathur Campus, Tamil Nadu 603203, India*

[6] *School of AI and Advanced Computing, XJTLU Entrepreneur College (Taicang), Xi'an Jiaotong-Liverpool University, Suzhou 215400, China*

*** Corresponding author:** Balasubramanian Prabhu kavin, ceaserkavin@gmail.com

ABSTRACT

In the wake of recent progresses in electronic trade and communiqué links, credit card use has skyrocketed for both online and in-person purchases. Maximum credit card datasets are very skewed, making it difficult to design efficient fraud detection algorithms that can help mitigate these losses. Traditional approaches are inefficient for credit card fraud detection because their architecture requires a vector to the output vector. As a result, they are unable to billet the ever-changing holders. In instruction to well recognize credit card fraud, the authors of this research suggest a hybrid classifier and data resampling strategy. The hybrid resampling is accomplished by combining the synthetic marginal oversampling procedure (SMOTE) with the edited nearest neighbour (ENN) approach. Temporal convolutional networks (TCN) are combined with a Bidirectional Gated Recurrent Unit (BiGRU) and a Dual Attention network (DATT) to perform the categorization in the suggested model. Second, in order to quickly get the deep semantic features of the credit card data, we employed TCN and BiGRU networks to extract characteristics were then spliced and merged, and a dual attention method was implemented to assign global weight to the most crucial information. In the end, classification was performed using a Softmax classifier. The accuracy of the categorization is further enhanced by the use of the Hybrid White shark Optimisation model (HWSO) model for selecting the model's weights. Using publicly accessible, credit card dealings, the recommended strategy is proved. The trial results show that the models adjusted using the proposed approach outperformed those using hybrids of competing metaheuristics.

Keywords: synthetic minority oversampling technique; temporal convolutional networks; bidirectional gated recurrent unit; hybrid white shark optimization model; credit card fraud detection

1. Introduction

The trend towards a cashless society means people will have to rely more and more on digital payment systems. In today's digital age, fraudsters rarely need to be present at the scenes of their crimes. They may hide their identities while carrying out their evil deeds in the safety of their own houses. Using a virtual private network (VPN), redirecting the victim's traffic through the Tor network, or anything similar makes it difficult to track their true identity. The significance of monetary losses incurred online cannot be overstated. In India, card data of some 70 million individuals[1], criminals can either use the stolen card details themselves or sell them to other people. In the United Kingdom, GBP 17 million were lost due to one of the largest fraud

ARTICLE INFO

Received: 30 January 2024
Accepted: 29 February 2024
Available online: 19 April 2024

occurrences in recent history. More than 32,000 credit card numbers were stolen by an international ring of thieves in the middle of the 2000s[2]. This is widely regarded as the largest card scam ever. Credit card fraud costs businesses and consumers alike year[3]. There is reassurance for both cardholders making purchases with their cards and card issuers handling those purchases. Fraudsters' goal is to have victims and financial institutions think their fraudulent transactions are real when they are not.

There are also ongoing fraudulent transactions that take place for financial advantage, unbeknownst to card issuers and cardholders alike. The darkest aspect of credit card transactions is that sometimes neither authorised institutions nor cardholders are aware that they have fraudulent transactions. When the sum of fraudulent dealings is small associated to the number of legitimate ones, it becomes extremely difficult to identify the former[4]. Predictive analytics methods and anomaly detection[5], are only two of the numerous fraud detection strategies that assist to avoid fraud in the financial industry. Machine learning procedures, either supervised or unsupervised, can be useful in classifying credit card fraud[6], but without them, none of these methods would be possible. However, such machine learning systems face a plethora of obstacles when attempting to detect all forms of fraud[7]. Since it is exceedingly unusual to have a balanced algorithm often diminishes the minority class in the dataset[8]. Actually, the minority group is the most significant factor in the categorization procedure, notably in identifying credit card fraud. The suggested method draws attention to the resampling approaches to compensate for the uneven distribution of classes in the dataset.

The class inequity in the datasets makes credit card fraud detection difficult from a learning standpoint[9]. Credit card fraud detection has been hampered by more than just the class divide, but it is by far the most significant issue. Due to classes in datasets, the class imbalance applications[10]. As an illustration, the dominant class's samples tend to be larger than the minority classes. The genuine dealings often outweigh the fraudulent ones[11], creating an imbalance in most credit card transaction databases. When trained on evenly distributed data, most standard ML techniques perform admirably. Conventional ML methods have biassed performance towards the majority class due to the skewed class distribution since the algorithms are designed to address the error rate rather than the class distribution. As a result, misclassification occurs more frequently among minority group samples than among majority group samples[12].

A data-level classification strategies for unbalanced data may be broken down into three broad classes approaches will under sample the majority class in order to oversample the minority class, or vice

versa[13]. Algorithmic approaches attempt to address the subject of class imbalance by training the classifier to place a greater emphasis on instances from the underrepresented classes. Ensemble learning and cost-sensitive learning approaches[14] are two examples of techniques that may be applied at the algorithm level. On the other hand, hybrid approaches integrate data-level and algorithm-level methodologies. The typical assessment measures should all point to perfect scores for the ideal account[15,16].

This study uses the SMOTE-ENN methodology to resample the unbalanced data in order to construct an efficient feature engineering strategy. TCN-BiGRU-DATT is used to classify data, with HWSO optimising the weight given to each approach. Finally, two datasets are used to evaluate the various indicators. The remaining contents of the paper are as shadows: The research contribution is broken down into five sections: Section 2 shows the linked papers, Section 3 explains the suggested perfect, Section 4 describes the findings, and Section 5 suggests directions for further study.

2. Related works

Using a deep Autoencoder as a procedure, Fanai and Abbasimehr[17] offer a scheme for sleuthing fraudulent transactions. Experiment results show that using the suggested method boosts the efficiency of the used classifiers. In particular, classifiers employing the altered data set generated by the deep Autoencoder show considerable improvements classifiers employing the original data across the board. Furthermore.

Using a four ensemble classifiers Salekshahrezaee et al.[18] explore these two preprocessing strategies. Both Convolutional Autoencoder (CAE) are compared and contrasted with one another in the context of feature extraction. Random Undersampling (RUS), SMOTE, and SMOTE Tomek are compared for their efficacy in data sampling. Measures of categorization performance RUS and CAE techniques for credit card fraud finding yields the greatest results.

A semi-supervised graph neural network for fraud finding has been proposed by Xiang et al.[19]. Through the use of transaction records, we are able to build a temporal deal graph that contains temporal transactions as nodes and interactions between them as edges. Then, a Gated Temporal Attention Network (GTAN) is used to learn by passing messages among the nodes. We further simulate the fraud trends by simulating the spread of risk from one transaction to another. Using a real-world transaction dataset and two open-source fraud detection datasets, comprehensive tests are undertaken. Our results on three different fraud detection datasets demonstrate that our suggested technique, GTAN, is superior to previous state-of-the-art baselines. Semi-supervised studies show that our model achieves state-of-the-art results in fraud detection while using just a negligible amount of labelled data.

The XGBoost (eXtreme Gradient Boosting) algorithm and data augmentation approaches are investigated by Noviandy et al.[20] to see if they may improve credit card fraud finding. This study provides empirical evidence that these methods are useful for correcting for unbalanced datasets and enhancing the precision with which fraud may be detected. By drawing on past transaction data and using methods like Over-sampling Technique-Edited Nearest Neighbours (SMOTE-ENN), the study demonstrates a method that strikes a good compromise between accuracy and the likelihood of catching fraud. These results have far-reaching implications for modern financial management, as they may help improve financial integrity, resource allocation, and consumer trust in the face of changing fraud strategies.

The OCSODL-CCFD method, as presented by Prabhakaran and Nedunchelian[21], combines a deep learning model for CCFD with a unique optimization-based feature selection methodology. The primary goal of the OCSODL-CCFD method is to identify and categorise credit card fraud. To pick the best set of features, the OCSODL-CCFD method develops a novel OCSO-based feature selection algorithm. Additionally, the chaotic krill herd procedure (CKHA) is used to tune the hyperparameters of the model, which is then used to the frauds. Numerous simulation evaluations were run to show how well the OCSODL-CCFD model

performed. The comprehensive comparison study revealed that the OCSODL-CCFD model outperformed the others in key assessment categories.

To assurance that only the most significant characteristics are used, Mienye and Sun[22] present a hybrid feature-selection approach that syndicates filter and selection processes. A employing the extreme algorithm is used in the suggested method, after the features have been rated using the information gain (IG) methodology. While traditional accuracy is not used in the proposed GA wrapper, the geometric mean (G-mean) is used as the fitness function. The suggested method outperformed previous baseline procedures and methods in the current literature, with a sensitivity of 0.997 and a specificity of 0.994, correspondingly.

Using Long Short-Term Memory (LSTM) DNNs for sequential data modelling, Fakiha[23] proposes a forensic fraud. This research investigates if an LSTM-attention algorithm can prioritise fraudulent transaction predictions based on an input sequence. Selecting the best enhances the LSTM-attention model's efficacy. The findings demonstrate the efficacy of using LSTM-attention algorithms for forensic is interesting since it effectively employs an LSTM-attention procedure to demonstrates the model's utility in reducing fraudulent dealings at financial institutions.

To combat credit card fraud, Berhane et al.[24] created a hybrid CNN-SVM perfect. Using publicly available credit card transaction data, we evaluated the efficacy of our proposed hybrid CNN-SVM model for identifying credit card fraud. To create our hybrid CNN-SVM model, we swapped out the CNN model's output layer with an SVM classifier. End-to-end training is used to create the first classifier, a fully softmax, while a support vector machine is layered on top by omitting the layer with softmax. Experiment results show that our hybrid CNN-SVM model achieved classification presentations of 91.08% accuracy, 90.50% precision, 90.34% recall, 90.41 F1-score, and 91.05% AUC.

3. Proposed system

3.1. Dataset description

3.1.1. First dataset

The widely-used credit card dataset[25] is the basis of this study. Machine Learning Group at ULB focused on massive data mining and fraud detection to compile the dataset[26]. The data collection includes September 2013 credit card transactions from European customers who made their purchases within a span of two days. Only 492 out of 284 807 transactions in the sample are fraudulent, which indicates an imbalance. However, the dataset was transformed, so all the characteristics except "Time" and "Amount" are numeric, and they are coded as V_1,V_2,...,V_28 to protect the privacy of the information. The "Amount" field represents the total amount spent, whereas the "Time" parameter is the amount of time from the dataset's inception that has passed since the initial transaction. Finally, the dependent variable is the attribute "Class," which takes the worth 1 for fake dealings and the value 0 for valid ones.

3.1.2. Second dataset collection

Researchers from IEEE Computational Intelligence Society[27] made use of data given by Vesta and made it available to the Kaggle community. About 500,000 credit card purchases were used to compile this dataset, which includes a goal feature and 432 characteristics (both numerical and categorical) per purchase. There are around 569 K valid transactions and about 20 K invalid ones, making for a very unbalanced data set (imbalance rate = 0.035).

Due to the large sum of transactions in the original dataset, we had to produce a smaller random dataset in order to show our proof of concept (POC). It's important to note that the POC dataset still has original dataset. Due to the sheer volume of features, only the first and last names of features sharing a first name have been included in the diagram. For instance, there are often six different cards in a deck: cards 1, 2, 3, 4, and 6.

3.2. Data preparation

This stage is essential since it determines the ultimate outcome of any predictive analytics project. It is common knowledge that real-world datasets are inherently disordered due to the presence of numerous outliers, missing values, atypical cardinality, etc. If these situations aren't managed properly, the research might fail. Our methods in this study for dealing with missing data, altering category features, scaling features, selecting features, and resampling.

3.2.1. Missing values

For several characteristics, the percentage of missing values was as high as 99 percent. In all, missing values account for almost 45 percent of the dataset, and estimating such a big proportion introduces bias into the model and leads to inaccurate forecasts. Therefore, as a general rule, if the missing values get beyond 60%, the features will be deleted since the feature's data is insufficient to contribute to the prediction model[28]. Less than half of the characteristics had missing values, hence the median and mode were used to infer the missing values for numerical features. The data are extremely skewed, making the median a more appropriate metric to use than the mean.

3.2.2. Encoding categorical topographies

Maximum machine learning procedures expect numerical data formats for both input and output attributes. This necessitates the transformation of categorical data into numeric form before a prediction model can be created. After filtering down characteristics with significant missing data, we were left with 15 category features. Ten of them had just two possible values, therefore we substituted 0 and 1 for them (for example, attributes with false and true). One-hot encoding, a simple and commonly used encoding method, was employed for the conversion of the remaining category characteristics with more than two levels. A new categorical column is created for each category, and a binary value of 1 or 0 is assigned to each value. Here, we have a 5-dimensional feature vector representation of the category property Product CD, which has the values encoded as [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 1, 0], and [0, 0, 0, 1]. The get dummies function from the Pandas library is used to implement this method.

3.2.3. Feature scaling

The range, unit, and magnitude of most characteristics in a real-world dataset will be different. The issue comes when the size of one aspect is significantly greater than that of the others, as that trait will inevitably predominate. As a result, classification algorithms require raw data to be scaled to remove the effect of different quantitative units. This study rescaled the characteristics between 0 and 1 using the MinMaxScaler method. Since it makes use of statistical methods that do not change the variance of the data, this method is resistant to outliers (Equation (1)).

$$x' = (x - \frac{\min(x)}{\max(x)} - \min(x)) \quad (1)$$

In the aforementioned Equation (1), x is the unscaled value, x' is the scaled value, max is the maximum feature value, and min is the least feature value. By maintaining the sparsity of the input data, MinMaxScaler's scaling is efficient even for data with many zero entries.

3.2.4. Data resampling

Unfortunately, most machine learning algorithms assume minorities and majorities are distributed equally, leading to inaccurate results and subpar predictive modelling performance, as is the case with the dataset used in this research. Learning with too few minority class examples appears to contribute to the imbalance issue, especially when additional variables, such as class overlaps, are present[29]. This research makes use of SMOTE-ENN to get over the issue of unbalanced datasets and overlapping classes because of this advantage. **Table 1** shows the difference in the number of observations between the pre-SMOTE-ENN and post-SMOTE-ENN classes. Thus, 37, 894 observations are obtained in this study, including both fraudulent and non-

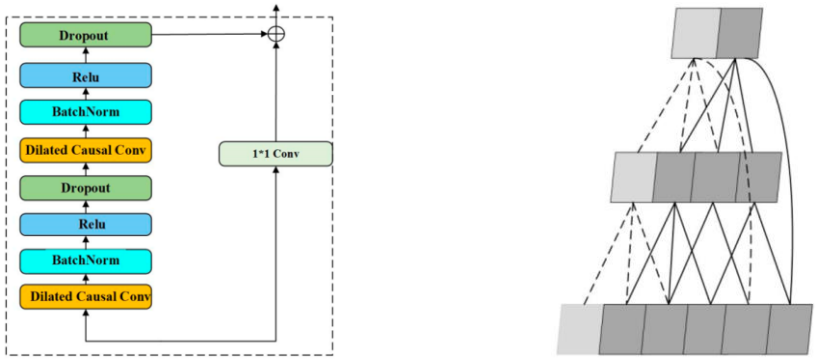
fraudulent cases.

Table 1. The sum of comments for each class beforehand and afterward SMOTE-ENN.

	Sum of comments for fraud cases	Sum of comments for non-fraud cases
Before SMOTE-ENN	1157	32,060
After SMOTE-ENN	28,689	27,155

3.3. Classification for CCFD

Text sentiment analysis benefits from the feature extraction on time scales that temporal convolutional networks (TCN) perform. The TCN residual unit, composed of two levels of dilation via a residual link, is the fundamental building block of the TCN model. As the TCN field expands, the number of its layers decreases; the residual network is employed to counteract this trend. Benefits include constant gradient, reduced memory use, and parallel processing. Two basic units, including weight normalisation, a ReLU layers, make up the residual module shown in **Figure 1a**. In order to regularise the network, we use weight normalisation and dropout, and we replace the simple connections between the TCN layers with the residual structure. **Figure 1b** is an example of a residual connection in a TCN.



(a) **(b)**

Figure 1. TCN residual unit. **(a)** residual block covering two basic components; **(b)** instance of residual connectivity in TCN.

Convolution calculation is the primary tool used by the TCN model to process the time series text, with causal convolution expansion and classification and normalisation of the parameters following to round out the nonlinear calculation. Time characteristics are extracted well, and both high- and low-dimensional hidden elements of context sequences are captured with ease. The following are some of TCN's primary benefits: TCN uses fewer network layers thanks to the inclusion of dilated convolution, which in turn makes it possible for TCN to analyse a time series without missing any important information at pivotal periods in the past. It can also expand its receptive field to take in additional data in real time. By decreasing the number of layers, we can cut down on the parameters, the amount of memory needed, and the number of computations required. When the sum of network layers is significant, gradient vanishing can occur during backpropagation, an issue that is addressed by the TCN's introduction of a residual module. However, helpful in getting a complete picture of the text. The BiGRU model was implemented in light of this issue.

3.3.1. TCN-BiGRU-DATT perfect

This research offers the TCN-BiGRU-DATT uses credit card data from the period of the outbreak as input and returns a sentiment category using a combination of the TCN and BiGRU mechanism. **Figure 2** depicts the construction of the device.

The last component of the model is a word vector layer: TCN-ATT features, is part of ALBERT's text vector representation. BiGRU-ATT feature extraction, which helps us better relations between words, is part of ALBERT's second channel feature extraction layer.

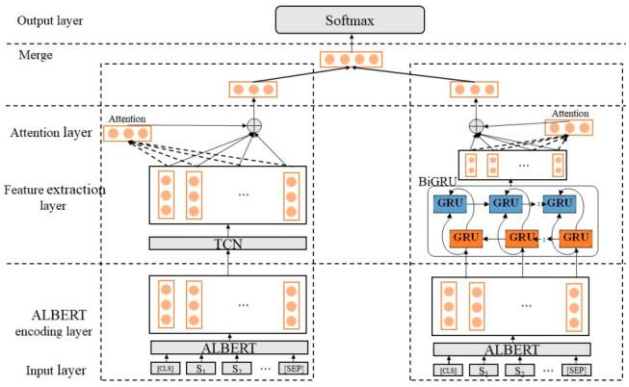


Figure 2. TCN-BiGRU-DATT model construction.

3.3.2. Input layer

The paper's word vector was created by tweaking the ALBERT pretraining model. The phrase is represented as where n is the number of tokens in ALBERT's input from the input layer. $S = s_1, s_2, \ldots, s_n$.

Token Embeddings of each word as model, respectively. The defining trait is $(e_1^t, e_2^t, \ldots, e_n^t)$, the sentence feature is $(e_1^s, e_2^s, \ldots, e_n^s)$, the location feature is $(e_1^p, e_2^p, \ldots, e_n^p)$, and the calculation formula of the input layer is $C_i = (e_i^t + e_2^s + e_i^p)$. Input last word embedded in $X_w = (x_1, x_2, \ldots, x_i) \in R^{L \times d}$, where L d vector.

3.3.3. Feature extraction layer

TCN-ATT feature extraction path:

The TCN and attention mechanism components are located in Section 1. In order to extract more thorough and deep text feature info, the ALBERT model is extended with the TCN model by sampling and calculating feature info produced by ALBERT. Input the TCN model with the vector h_t that was generated by the ALBERT model's last hidden layer. Here is the exact Equation:

$$S_i = Conv(M_i + K_j + b_i) \tag{2}$$

$$\{S_0, S_1, \ldots., S_n\} = LayerNorm(\{S_0, S_1, \ldots, S_n\}) \tag{3}$$

$$\{C_0, C_1, \ldots, C_n\} = ReLU\{S_0, S_1, \ldots, S_n\} \tag{4}$$

where S_i stands for the rank value obtained M_i. K_j represents core. b_i characterizes the bias vector. $\{S_0, S_1, \ldots, S_n\}$ that was derived via non-linear analysis. The output q is acquired by non-linear transformation after the feature vector H has been obtained via processing the TCN model. This is the correct Equation:

$$H = h_t + \{C_0, C_1, \ldots, C_n\} \tag{5}$$

$$q = HW^{n \times m} \tag{6}$$

Equation (6), $W^{n \times m}$ indicates the linear transformation's parameter matrix, n stands for the semantic vector's dimensions before the transformation, and m stands for its dimensions thereafter. The output matrix q from process is injected into the attention mechanism to filter out more salient emotional feature info and enhance classification precision. This is the Equation for calculating:

$$u_i = tanh(W_s q_i + b_s) \tag{7}$$

$$a_i = \frac{exp(u_i)}{\sum_{s=1}^{n} \exp(u_s)} \tag{8}$$

$$F = \sum_{t=1}^{n} a_i q_i \tag{9}$$

To clarify, q_i is the TCN model's learned feature representation vector, u_i is the hidden layer representation of q_i derived from calculation, W_s is the weight matrix, b_s is the bias matrix, and a_i is the normalisation derived from the Softmax value. After the weighted procedure, the produced feature vector comprises essential feature information, and the normalised weight value reflects the value of the result.

BiGRU-ATT feature extraction path:

In this work, the BiGRU network located on route 2. When processing text sequences in both directions, we employ the BiGRU network, with its gate structure controlling the transmission state to provide a memory function. Left-to-right and right-to-left semantic information is obtained by feeding the feature vector generated from the ALBERT perfect into BiGRU. More data on the characteristics of lengthy text sequences may be stored using the ALBERT model. As can be seen in the diagram below, the BiGRU network takes as its input the vector h_t representing final hidden layer in the ALBERT model.

$$\vec{l}_t = \overrightarrow{GRU}(\overrightarrow{l_{t-1}}, h_t) \tag{10}$$

$$\overleftarrow{l_t} = \overleftarrow{GRU}(\overleftarrow{l_{t+1}}, h_t) \tag{11}$$

$$l_t = \vec{l}_t \oplus \overleftarrow{l_t} \tag{12}$$

BiGRU's forward output result at time t is denoted by l_t, its reverse output consequence at time t by (l_t), and its output result at time t by l_t in the preceding formula. By feeding the BiGRU model's output into an attention apparatus, the latter can assign a higher weight to the vector information that consequence and thus extract additional crucial info from the review text. The math goes like this:

$$z_t = tanh(W_s l_t + b_s) \tag{13}$$

$$\beta_t = \frac{exp(z_t)}{\sum_{s=1}^{n} \exp\,(u_s)} \tag{14}$$

$$V = \sum_{t=1}^{n} a_i l_t \tag{15}$$

The above formula, z_t characterizes transformation, β_t characterizes individually.

3.3.4. Output layer

First, a new feature vector is built by fusing the feature vectors collected from the two channels. Equation (16) shows that the feature vectors F and V are spliced together using the vector splicing approach to get the final feature vector representing y. This reduces the model's complexity.

$$y = F \oplus V \tag{16}$$

After the features have been fused, they are sent into a Softmax classifier to be utilised as input in calculating the model's final projected classification probability value, which is described as follows:

$$O = Softmax(W \cdot y + b) \tag{17}$$

The polarity of the credit card data is determined by the value of yield fault data belongs, which is represented by Equation (17), W is the weight matrix ideally selected by WSO, and b is the bias vector.

3.3.5. Weight Selection using WSO

Braik et al.[30] presented the white of the newest meta-heuristic methods. The authors were inspired to create this algorithm by the complex behaviours of great white sharks, which include the use of highly developed senses of hearing and smell. White sharks are gorgeous and highly adapted hunters; they use their powerful jaws and fins to catch dolphins, small whales, crabs, seabirds, and seals. When hunting, great white sharks use surprise techniques to get the jump on their prey, and then follow up with large, lethal blows.

Following the hesitations in the waves caused by the prey's movement, randomly searching for depths, and discovering nearby prey are the three behaviours necessary for devouring the prey. The great white sharks are aided in updating their locations and finding optimal solutions by following these procedures. A population matrix of starting solutions can be used to represent the WSO. Where N is the population size and d is the issue dimension, the size of the initial population matrix is $N\,d$[30]:

$$w = \begin{bmatrix} w_1^1 & w_2^1 & \cdots & w_d^1 \\ w_1^2 & w_2^2 & \cdots & w_d^2 \\ \vdots & \vdots & \vdots & \vdots \\ w_1^n & w_2^n & \cdots & w_d^n \end{bmatrix} \tag{18}$$

where w_j^i characterizes the ith white shark site dimension. It can be intended based on the lower (lb_j) (ub_j) dimension as[30]:

$$w_j^i = lb_j + rand \times (ub_j - lb_j) \tag{19}$$

where rand is an arbitrary integer between zero and one. The fitness values for the initial solutions are determined using Equation (20), and an updating procedure is set into motion if the new location is superior to the old one. After determining the position of its prey by wave hesitation, the great white shark will swim in undulating motions at a rate of:

$$v_{k+1}^i = \mu \left(v_k^i + \rho_1 [w_{gbest_k} - w_k^i] \times c_1 + \rho_2 \left[w_{best}^{v_k^i} - w_k^i \right] \times c_2 \right) \tag{20}$$

where v_{k+1}^i and v_k^i are the efficient and present velocities repetitions $k + 1$ and k, correspondingly; wgbestk represents the global finest site during the kth repetition; w_k^i is the situation repetition k; c_1 and c_2 are random statistics within a range $[0, 1]$; $w_{best}^{v_k^i}$ characterizes during repetition k; and v_k^i is the index vector sum i for the procurement the best site, and it can be well-defined as shadows:

$$v = [n \times rand(1, n)] + 1 \tag{21}$$

The parameters ρ_1 and ρ_2 are the great control the w_{gbest_k} and $w_{best}^{v_k^i}$ best effects on w_k^i; they can be computed as shadows[30]:

$$\rho_1 = \rho_{max} + (\rho_{max} - \rho_{min}) \times e^{-\left(\frac{4k}{K}\right)^2} \tag{22}$$

$$\rho_2 = \rho_{min} + (\rho_{max} - \rho_{min}) \times e^{-\left(\frac{4k}{K}\right)^2} \tag{23}$$

where ρ_{min} and ρ_{max} are the initial to get white sharks, ρ_{min} = 0.5 and ρ_{max} = 1.5, and K signifies the extreme repetition. The term m in is issue; it is used to analyser the junction rate of the WSO via the subsequent expression[30]:

$$\mu = \frac{2}{\left| 2 - t - \sqrt{t^2 - 4t} \right|} \tag{24}$$

where t is some sort of algorithmic acceleration factor.

As was previously said, spend the vast majority of their time hunting for food. Because of this, their locations shift as they approach their prey, which they do by either listening to the waves caused by the prey's movements or detecting the prey's odours. Great white sharks in this scenario wander to seemingly random locations as they hunt for food; this behaviour might be depicted as follows[30]:

$$w_{k+1}^i = \begin{cases} w_k^i \blacksquare \neg \oplus w_0 + ub \blacksquare a + lb \blacksquare b & if\ rand < mv \\ w_k^i + \dfrac{v_k^i}{f} & if\ rand \geq mv \end{cases} \tag{25}$$

where ¬ is the two vectors distinct by Equations (26) and (27), w_0 characterizes logical vector subtracted via Equation (28), f is intended based on Equation (29):

$$a = sgn(w_k^i - ub) > 0 \tag{26}$$

$$b = sgn(w_k^i - lb) < 0 \tag{27}$$

$$w_0 = \oplus(a, b) \tag{28}$$

$$f = f_{min} + \frac{f_{max} - f_{min}}{f_{max} + f_{min}} \tag{29}$$

where f_{max} and f_{min} are the upper and lower bounds on the undulating frequencies at which the great white shark oscillates. The great white shark's driving power, represented by the parameter mv, is raised in an iterative fashion as follows:

$$mv = \frac{1}{a_0 + e^{\left(\frac{0.5K - 5}{a_1}\right)}} \tag{30}$$

where a_0 and a_1 are two controls used to regulate exploratory and exploitative actions. The use of mv

increases the WSO's search efficiency and fortifies its exploratory and exploitative tendencies. For this reason, the author chose to implement such an algorithm to address the addressed issue. You may model the great white shark's approach to its prey in this way:

$$\acute{w}_{k+1}^{i} = w_{gbest_k} + r_1 \vec{D}_w \times sgn(r_2 - 0.5) \quad if \quad r_3 < S_s \tag{31}$$

where $\acute{w}_{k+1}^{i}$ indicates the ith great white shark's current site in relation to its prey; sgn($r_2 - 0.5$) determines the returning are consistently distributed random values in the intermission [0, 1]; and $\vec{D}_w$ characterizes the distance between the white shark and its prey as follows:

$$\vec{D}_w = \left| rand \times \left(w_{gbest_k} - w_k^i \right) \right| \tag{32}$$

The great white shark's keen sense of sight and smell is characterised by the parameter S_s in Equation (33), which may be determined as shadows:

$$S_s = \left| 1 - e^{\frac{-a_2 k}{K}} \right| \tag{33}$$

where a_2 is a parameter used to control the performances.

The proposed hybrid WSO-Based practice

The great white sharks in the simplest implementation of the WSO all move in the same direction as they approach their meal, which might lead to the algorithm missing out on potentially better options in the area. Therefore, in this study, the WSO has been combined with a different method based in order to improve the exploitation behaviour of the original WSO. According to the whale optimisation algorithm (WOA), the application of the spiral-shaped route was motivated by whale behaviour when naiant to prey spots. The following correlation represents the great white shark's spiralling journey to its victim:

$$W_{t+1}^{i} = \vec{D}.e^{hl}.\cos(2\pi l) + W_t^* \tag{34}$$

$$\vec{D} = \left| \overrightarrow{W^*} - \vec{W} \right| \tag{35}$$

where $\vec{D}$ is the great white shark's hunting range, h is a continuous used to describe the logarithmic between −1 and 1. The great white shark may be made to go towards the direction of its prey by modifying the spiral equation as follows:

$$w_{t+1}^{i} = \begin{cases} w_{gbest_k} + r_1 \vec{D}_w \times sgn(r_2 - 0.5) & if \;\; r_3 < S_s \\ \vec{D}.e^{bl}.\cos(2\pi l) + w_t^* & if \quad r_3 < S_s \end{cases} \tag{36}$$

The primary framework of the proposed HWSO is summarised in pseudo code (see Algorithm 1) below when applied to the parameter estimation and optimisation issue of the battery model. Assigning initial random values for the model parameters' lower and upper bounds is the first step towards producing that random collection of solutions. Then, the relevant values for the goal function's starting point are determined using Equation (5). The fundamental structure of the HWSO is used to refine the original solution set over the course of several iterations. After the endpoint requirements were fulfilled, the optimal settings were shown.

4. Results and discussion

The Jupyter Notebook situation on the Anaconda platform was used to run the tests on a Windows 10 machine equipped with an Intel Core i7—16 GB RAM.

4.1. Performance metrics

Several statistical metrics, including the deviation, and the computing time consumed during the selection of features, were considered to assess the quality of the built model. The categorization process then led to the calculation of statistical measures. These parameters are expressed mathematically as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{37}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{38}$$

$$Specificity = \frac{TN}{TN + FP} \tag{39}$$

$$F_{Score} = 2 \times \frac{Specificity \times Sensitivity}{Specificity + Sensitivity} \tag{40}$$

where "TP" is the abbreviation of $true\ positives$. "TN" stands for the true negative tasters. "FP" is the abbreviation of false positives, while "FN" is the abbreviation of $false\ negatives$.

4.2. Performance analysis of proposed model

Experiment results contrast the projected perfect to state-of-the-art approaches on two datasets are exposed in **Table 2**.

Table 2. Experimental analysis of various classifiers.

Models	**Dataset1**				**Dataset2**			
Metrics	**Accuracy**	**Recall**	**Precision**	**F1-Score**	**Accuracy**	**Recall**	**Precision**	**F1-Score**
GRU	0.773	0.773	0.776	0.772	0.963	0.958	0.959	0.959
BiGRU	0.766	0.766	0.770	0.764	0.958	0.827	0.967	0.892
CNN	0.768	0.769	0.772	0.767	0.957	0.956	0.928	0.942
RNN	0.761	0.761	0.764	0.760	0.925	0.920	0.967	0.943
LSTM	0.769	0.769	0.771	0.767	0.958	0.812	0.873	0.841
BiLSTM	0.771	0.771	0.775	0.769	0.963	0.944	0.802	0.867
Proposed	0.793	0.813	0.795	0.82	0.974	0.974	0.974	0.974

In above **Table 2** characterise that the Experimental Analysis of numerous classifiers. In the dataset 1 evaluation, the CNN model accomplished the accuracy as 0.769 and recall as 0.769 and precision of 0.771 and then F1-score as 0.767 correspondingly. Then the RNN model accomplished the accuracy as 0.761 and recall as 0.761 and precision of 0.760 correspondingly. Then the LSTM model accomplished the accuracy as 0.769 and recall as 0.769 and precision of 0.771 and recall as 0.767 consistently. Then the BiLSTM model accomplished the accuracy as 0.771 and recall as 0.771 and precision of 0.769 congruently. Then the GRU model accomplished the accuracy as 0.773 and recall as 0.773 and recall as 0.776 and then F1-score as 0.772 congruently. Then the BiGRU model accomplished the accuracy as 0.766 and recall as 0.766 and precision of 0.770 and then F1-score as 0.764 congruently. Then the Projected model accomplished the accuracy as 0.793 and recall as 0.813 and precision of 0.795 and recall as 0.82 correspondingly. Then the 2nd dataset evaluation, CNN 0.957 and recall as 0.956, 0.928 and then F1-score as 0.942 correspondingly. Then the RNN model accomplished the accuracy as 0.925 and recall as 0.920 and precision of 0.967 0.943 correspondingly. Then the LSTM model accomplished the accuracy as 0.958 and recall as 0.812 and precision of 0.873 and then F1-score as 0.841 correspondingly. Then the BiLSTM model accomplished the accuracy as 0.963 and recall as 0.944 and precision of 0.802 and then F1-score as 0.867 correspondingly. Then the BiGRU model accomplished the accuracy as 0.958 and recall as 0.827 and precision of 0.967 and then F1-score as 0.892 congruently. Then the Projected model accomplished the accuracy as 0.974 and recall as 0.974 and precision of 0.974 and then F1-score as 0.974 correspondingly. **Figures 3** and **4** provides the graphical description of the proposed model with existing techniques in terms of different metrics.

In **Table 3** characterise that the Comparison of various models in terms of CPU time(s) on two datasets. In the analysis of CNN model attained the CPU period in dataset 1 as 3.481 and CPU time in dataset 2 as 26.654 RNN model attained the CPU period in dataset 1 as 3.260 and CPU time in dataset 2 as 26.572 correspondingly. Then the LSTM model attained the CPU period in dataset 1 as 4.294 and CPU time in dataset 2 as 27.009 correspondingly. Then the BiLSTM model attained the CPU period in dataset 1 as 3.602 and CPU time in dataset 2 as 31.179 correspondingly. Then the GRU model attained the CPU period in dataset 1 as

4.115 and CPU time in dataset 2 as 31.028 correspondingly. Then the BiGRU model attained the CPU period in dataset 1 as 3.197 and CPU time in dataset 2 as 24.013 correspondingly. Then the Proposed model attained the CPU period in dataset 1 as 3.123 and CPU time in dataset 2 as 25.737 correspondingly. **Figures 5** and **6** mentions the graphical description of proposed model in terms of CPU time on two datasets.

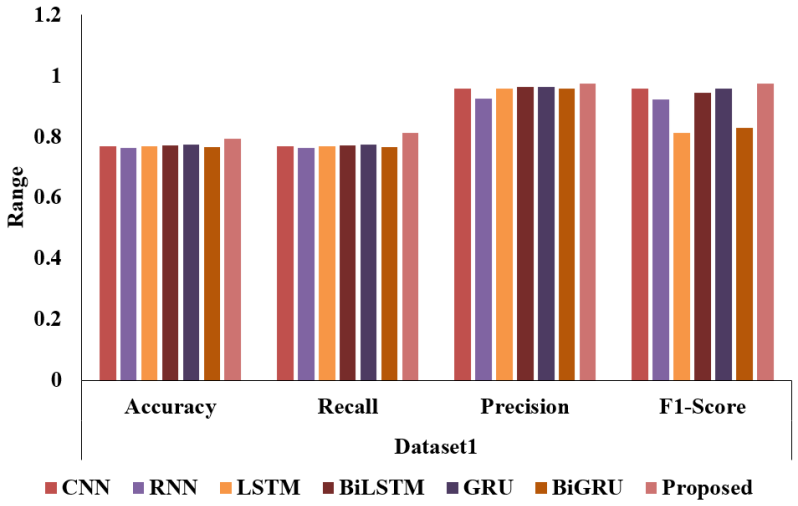


Figure 3. Graphical analysis of various classifiers.

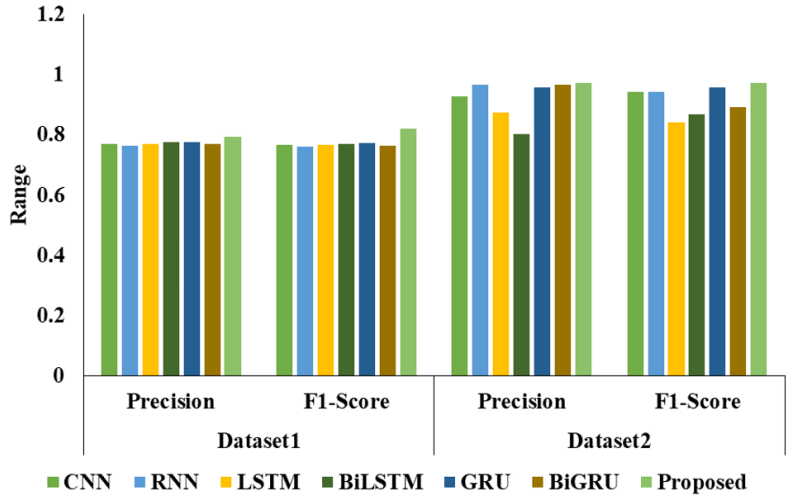


Figure 4. Analysis of different models on two datasets.

Table 3. Comparison of various models in terms of CPU time(s) on two datasets.

Dataset	**Dataset1**	**Dataset2**
Timing	**CPU Time (s)**	**CPU Time (s)**
CNN	3.481	26.654
RNN	3.260	26.572
LSTM	4.294	27.009
BiLSTM	3.602	31.179
GRU	4.115	31.028
BiGRU	3.197	24.013
Proposed	3.123	25.737

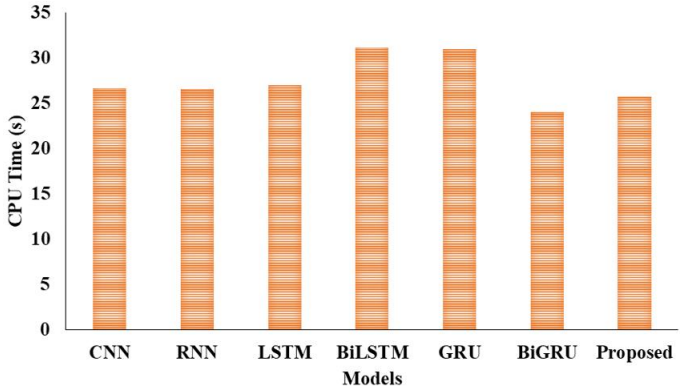


Figure 5. CPU time analysis on second dataset.

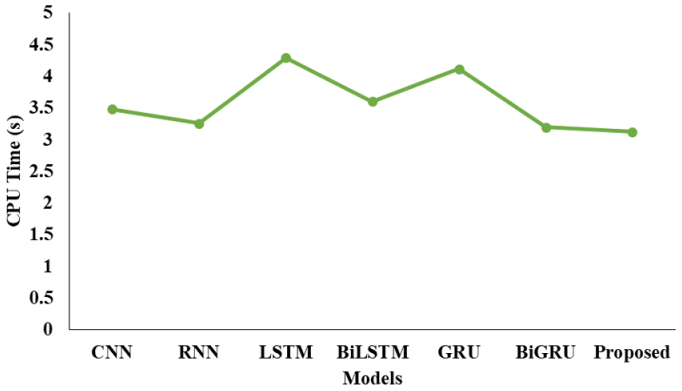


Figure 6. Analysis of time on first dataset.

Characterise the comparison with other optimisation models to obtain accuracy results in **Table 4**. In the analysis of the first dataset, the MFO with GRU achieved 0.667, BiGRU achieved 0.766, DCNN achieved 0.757, and TCN achieved 0.939, respectively. The WOA with GRU then reached 0.672 and BiGRU reached 0.751, 0.742, 0.761, and 0.937, respectively. The FFA then had a GRU of 0.670, a BiGRU of 0.784, 0.742, and a TCN of 0.769 0.960, all in the appropriate ranges. The BAT with GRU then reached 0.667, BiGRU as 0.756 0.761, and TCN as 0.771 0.935, respectively. The HGS with GRU then reached 0.672, BiGRU as 0.764, DCNN as 0.742, and TCN as 0.773, respectively. Then, the TSO with GRU reached 0.665, BiGRU reached 0.785, DCNN reached 0.725, and TCN reached 0.766 and 0.961, respectively. The second dataset was then calculated with the following values: HWSO 0.676, BiGRU 0.777, and DCNN 0.751 0.783 0.972. The MFO with GRU then reached 0.939, BiGRU reached 0.947, DCNN reached 0.938, and TCN reached 0.957, respectively. The WOA with GRU then reached 0.937, BiGRU 0.936, DCNN 0.936, and TCN 0.925, respectively. The FFA with GRU then reached 0.960, BiGRU as 0.934, and DCNN as 0.959 0.958 in that order. Then, the BAT was 0.935, the BiGRU was 0.959, and the DCNN was 0.935 to 0.963, respectively.

Table 4. Comparison with other optimization models to obtain accuracy results.

Models	**TCN**	**GRU**	**BiGRU**	**DCNN**	**GRU**	**BiGRU**	**TCN**
MFO	0.769	0.667	0.766	0.757	0.939	0.947	0.957
WOA	0.761	0.672	0.751	0.742	0.937	0.936	0.926
FFA	0.769	0.670	0.785	0.743	0.960	0.934	0.958
BAT	0.771	0.667	0.756	0.761	0.935	0.959	0.963
HGS	0.773	0.672	0.764	0.742	0.944	0.935	0.962
TSO	0.766	0.665	0.786	0.725	0.961	0.934	0.958
HWSO	0.783	0.676	0.777	0.751	0.972	0.973	0.974

5. Conclusion

Credit card fraud is a growing problem across the world, especially for banks. The goal of this work was to examine alternative reliable strategies for detecting fraudulent credit card transactions; while many methods have been employed before, there is still a need to do so. Recognising credit card fraud has become increasingly important in fresh years, with machine learning playing a pivotal role. In this study, a practical procedure for identifying credit card fraud was suggested. To begin, a symmetrical dataset was produced using the SMOTEENN method. Second, the TCN-BiGRU-DATT model was used to construct a powerful deep learning ensemble. Integrating weight was made possible by extraction. The HWSO technique is also used to choose the weights of the proposed model. It will take several trials to compare different models and determine which one is most effective. Therefore, the suggested classifier combined with the SMOTE-ENN data resampling strategy is an effective tool for identifying credit card fraud. This finding also demonstrates that errors have been reduced thanks to hybrid approaches. Different approaches of explored in the credit card arena and their effects on prediction accuracy determined in future study. Improved feature selection and additional resampling methods would be the focus of a follow-up study aimed at boosting classification accuracy. The HGS with GRU then achieved 0.944, BiGRU 0.935, DCNN 0.945, and TCN 0.963, respectively. The TSO with GRU then reached 0.961, BiGRU 0.934, DCNN 0.961, and TCN 0.958, respectively. The HWSO with GRU then reached 0.972, BiGRU as 0.973, DCNN as 0.971, and TCN as 0.974, in that order. Despite the hybrid resampling technique, imbalanced datasets remain a challenge in fraud detection. Future efforts should continue to address this issue to ensure that the model is trained on representative data from both fraudulent and non-fraudulent transactions. There's a continuous scope for enhancing the accuracy of fraud detection algorithms. Future research could focus on refining the hybrid resampling technique and optimizing the parameters of the TCN model to achieve even higher precision in identifying fraudulent transactions.

Author contributions

Conceptualization, SAR; methodology, RC; software, JPA; validation, DS, DAD and BPk; formal analysis, GHS; investigation, SAR; resources, RC; data curation, JPA; writing—original draft preparation, DS; writing—review and editing, GHS; visualization, JPA; supervision, BPk; project administration, SAR; funding acquisition, GHS. All authors have read and agreed to the published version of the manuscript.

Conflict of interest

The authors declare no conflict of interest.

References

1. Bin Sulaiman R, Schetinin V, Sant P. Review of Machine Learning Approach on Credit Card Fraud Detection. Human-Centric Intelligent Systems. 2022; 2(1-2): 55-68. doi: 10.1007/s44230-022-00004-0
2. Asha RB, KR SK. Credit card fraud detection using artificial neural network. Global Transitions Proceedings. 2021; 2(1): 35-41.
3. Carcillo F, Le Borgne YA, Caelen O, et al. Combining unsupervised and supervised learning in credit card fraud detection. Information Sciences. 2021; 557: 317-331. doi: 10.1016/j.ins.2019.05.042
4. Chen JIZ, Lai KL. Deep Convolution Neural Network Model for Credit-Card Fraud Detection and Alert. Journal of Artificial Intelligence and Capsule Networks. 2021; 3(2): 101-112. doi: 10.36548/jaicn.2021.2.003
5. Cherif A, Badhib A, Ammar H, et al. Credit card fraud detection in the era of disruptive technologies: A systematic review. Journal of King Saud University - Computer and Information Sciences. 2023; 35(1): 145-174. doi: 10.1016/j.jksuci.2022.11.008
6. Tanouz D, Subramanian RR, Eswar D, et al. Credit Card Fraud Detection Using Machine Learning. In: Proceedings of the 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS).
7. Forough J, Momtazi S. Ensemble of deep sequential models for credit card fraud detection. Applied Soft Computing. 2021; 99: 106883. doi: 10.1016/j.asoc.2020.106883
8. Alfaiz NS, Fati SM. Enhanced Credit Card Fraud Detection Model Using Machine Learning. Electronics. 2022; 11(4): 662. doi: 10.3390/electronics11040662

9. Zhang X, Han Y, Xu W, et al. HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture. Information Sciences. 2021; 557: 302-316. doi: 10.1016/j.ins.2019.05.023
10. Alharbi A, Alshammari M, Okon OD, et al. A Novel text2IMG Mechanism of Credit Card Fraud Detection: A Deep Learning Approach. Electronics. 2022; 11(5): 756. doi: 10.3390/electronics11050756
11. Alarfaj FK, Malik I, Khan HU, et al. Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms. IEEE Access. 2022; 10: 39700-39715. doi: 10.1109/access.2022.3166891
12. Xie Y, Liu G, Yan C, et al. Learning transactional behavioral representations for credit card fraud detection. IEEE Transactions on Neural Networks and Learning Systems; 2022.
13. Ileberi E, Sun Y, Wang Z. A machine learning based credit card fraud detection using the GA algorithm for feature selection. Journal of Big Data. 2022; 9(1). doi: 10.1186/s40537-022-00573-8
14. Dastidar KG, Granitzer M, Siblini W. The Importance of Future Information in Credit Card Fraud Detection. In: International Conference on Artificial Intelligence and Statistics (pp. 10067-10077). PMLR; 2022.
15. Esenogho E, Mienye ID, Swart TG, et al. A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection. IEEE Access. 2022; 10: 16400-16407. doi: 10.1109/access.2022.3148298
16. Roseline JF, Naidu GBSR, Pandi VS, et al. Autonomous credit card fraud detection using machine learning approach☆. Computers and Electrical Engineering. 2022; 102: 108132.
17. Fanai H, Abbasimehr H. A novel combined approach based on deep Autoencoder and deep classifiers for credit card fraud detection. Expert Systems with Applications. 2023; 217: 119562. doi: 10.1016/j.eswa.2023.119562
18. Salekshahrezaee Z, Leevy JL, Khoshgoftaar TM. The effect of feature extraction and data sampling on credit card fraud detection. Journal of Big Data. 2023; 10(1). doi: 10.1186/s40537-023-00684-w
19. Xiang S, Zhu M, Cheng D, et al. Semi-supervised Credit Card Fraud Detection via Attribute-Driven Graph Representation. Proceedings of the AAAI Conference on Artificial Intelligence. 2023; 37(12): 14557-14565. doi: 10.1609/aaai.v37i12.26702
20. Noviandy TR, Idroes GM, Maulana A, et al. Credit Card Fraud Detection for Contemporary Financial Management Using XGBoost-Driven Machine Learning and Data Augmentation Techniques. Indatu Journal of Management and Accounting. 2023; 1(1): 29-35. doi: 10.60084/ijma.v1i1.78
21. Prabhakaran N, Nedunchelian R. Oppositional Cat Swarm Optimization-Based Feature Selection Approach for Credit Card Fraud Detection. Computational Intelligence and Neuroscience. 2023; 2023: 1-13. doi: 10.1155/2023/2693022
22. Mienye ID, Sun Y. A Machine Learning Method with Hybrid Feature Selection for Improved Credit Card Fraud Detection. Applied Sciences. 2023; 13(12): 7254. doi: 10.3390/app13127254
23. Fakiha B. Forensic Credit Card Fraud Detection Using Deep Neural Network. Journal of Southwest Jiaotong University. 2023; 58(1).
24. Berhane T, Melese T, Walelign A, Mohammed A. A Hybrid Convolutional Neural Network and Support Vector Machine-Based Credit Card Fraud Detection Model. Mathematical Problems in Engineering; 2023.
25. Credit Card Fraud Detection. Available online: https://kaggle.com/mlg-ulb/creditcardfraud (accessed on 2 January 2024).
26. Patel H, Singh Rajput D, Thippa Reddy G, et al. A review on classification of imbalanced data for wireless sensor networks. International Journal of Distributed Sensor Networks. 2020; 16(4): 155014772091640. doi: 10.1177/1550147720916404
27. IEEE Computational Intelligence Society. IEEE-CIS Fraud Detection Can You Detect Fraud from Customer Transactions? 2019. Available online: https://www.kaggle.com/c/ieee-fraud-detection/overview (accessed on 2 January 2024).
28. Aoife D, Brian M, John DK. Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies. The MIT Press; 2015.
29. Le T, Vo MT, Vo B, et al. A Hybrid Approach Using Oversampling Technique and Cost-Sensitive Learning for Bankruptcy Prediction. Complexity. 2019; 2019: 1-12. doi: 10.1155/2019/8460934
30. Braik M, Hammouri A, Atwan J, et al. White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowledge-Based Systems. 2022; 243: 108457. doi: 10.1016/j.knosys.2022.108457