Journal of Autonomous Intelligence (2024) Volume 7 Issue 5
doi: 10.32629/jai.v7i5.1477

ORIGINAL RESEARCH ARTICLE

A deep learning based credit card fraud detection using feature engineering: An analytical approach

Ajanthaa Lakkshmanan[1,*], Gulshan Soni[2], Anand Kumar Mishra[3], Senthil Kumar Arumugam[4], Amit Kumar Tyagi[5,*]

[1] *Department of Computing Technologies, SRM Institute of Science and Technology, Kattankaluthur 6032023, Chennai, Tamilnadu, India*

[2] *MSE&IT, MATS University, Raipur 493441, Chhattisgarh, India*

[3] *Computer Science and Engineering, NIIT University, Neemrana 301705, Rajasthan, India*

[4] *Department of Professional Studies, Christ University, Bangalore, Karnataka 560029, India*

[5] *National Institute of Fashion Technology, New Delhi, 110016, India*

*** Corresponding authors:** Ajanthaa Lakkshmanan, ajanthal@srmist.edu.in; Amit Kumar Tyagi, amitkrtyagi025@gmail.com

ABSTRACT

The new advances in online business and e-installment frameworks have started an expansion in monetary misrepresentation cases, for example, credit card extortion. As a result, it is significant to execute instruments that recognise credit card extortion. Highlights of credit card cheats assume a significant part while AI is utilized for fraud recognition, and they should be picked appropriately. With the revolutionized advancements in technology such as deep learning, comes in handy to showcase its complex way and get accurate detection results. This paper introduces a tailored deep learning approach designed to efficiently detect fraudulent activities, encompassing the crucial phases outlined below: a) Gathering data, which includes approximately 153,685 transaction records from Chinese credit cards. b) Utilization of feature engineering techniques to preprocess the data, including analysing spending patterns and time-related features. c) Extraction of essential features using autoencoders. d) Feature selection employing particle swarm optimization (PSO). e) Detecting fraud through the utilization of recurrent neural networks (RNNs). The experimental evaluation provides evidence that the suggested system outperforms current leading models across multiple performance measures, achieving a 0.97 accuracy, a 0.98 sensitivity, and a 0.98 specificity.

Keywords: artificial intelligence; autoencoder; deep learning; feature engineering; PSO; RNN

ARTICLE INFO

Received: 11 December 2023
Accepted: 4 January 2024
Available online: 5 June 2024

1. Introduction

In recent times, due to the swift progression of technology, a growing portion of the population is choosing to employ credit cards when making purchases, leading to a steady increase in related fraudulent activities. In the present-day world, credit cards have emerged as the favoured payment method for virtually all types of businesses, spanning from small enterprises to large corporations. Credit card fraud is pervasive across multiple industries, encompassing sectors like appliances, automobiles, and banking, among others. Despite efforts to address this issue using methods like data mining and machine learning algorithms, the results often fall short of the expected outcomes. Consequently, there is a pressing need to develop more effective and efficient algorithms that can yield significant improvements. In this study, we aim to prevent fraudulent

credit card transactions by employing an artificial neural network algorithm and comparing its performance with several other machine learning algorithms before transaction approval[1].

Fraud is an immoral deed perpetrated by an unauthorized person, who misleads unsuspecting victims. Credit card fraud involves the unlawful acquisition of crucial cardholder information and its improper utilization by cyber attackers[2], whether via phone calls, text messages, or by exploiting software applications under the control of the culprits (refer to **Figure 1**).

Credit card fraud detection entails a process where the user or customer provides the required information to initiate a credit card transaction. Approval for this transaction is granted only after a comprehensive examination to identify any possible fraudulent behaviour. To achieve this, the transaction details are initially sent to a verification module, which categorizes them as either fraud or non-fraud. Transactions categorized as fraud are declined, while those in the non-fraud category are approved[3].

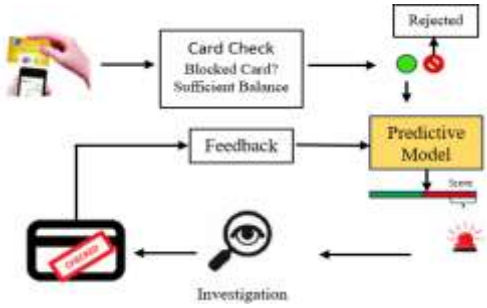

Figure 1. Overall conceptual view of fraud detection using DL.

1.1. Classifications

We need to explain few essential components of classifications as:

- Application fraud: After a malicious actor takes control of the application, steals customer credentials, and establishes a fake account, further transactions take place.
- Electronic or manual card imprints: In this deceptive plot, the offender unlawfully obtains data from the magnetic stripe of the card and subsequently employs these acquired credentials to conduct unauthorized transactions[4].
- Card not present: This relates to a specific category of credit card that does not require a physical card to complete a transaction.
- Counterfeit card fraud: The fraudulent activity involves a type of fraud where the perpetrator duplicates all the information from a magnetic strip, resulting in a counterfeit card that closely resembles an authentic one and functions in the same manner. This cloned card is employed for illicit purposes.
- Lost/stolen card: This form of deceitful behaviour takes place when the cardholder either loses their card or has it taken from them.
- Card ID theft: Credit card fraud takes place when the cardholder's identity is compromised, resulting in unauthorized and deceitful transactions.
- Mail non-received card fraud: In the credit card issuance procedure, there is a stage that entails sending an email to the recipient, and this particular step introduces a vulnerability to potential fraud, such as mail tampering or phishing attempts.
- Account takeover: In this situation, the wrongdoer seizes complete authority over the account holder's account in order to carry out deceptive actions.
- Fake fraud in the website: A cunning person may insert a malicious code that carries out their activities on the website.
- Merchant collision: In this specific type of fraudulent activity, merchants disclose cardholder information

to third parties or fraudsters without securing the cardholder's approval.

In summary, credit card fraud takes place when an unauthorized individual utilizes someone else's credit card without authorization, after obtaining essential information such as the PIN, password, and other credentials, even if they don't have the physical card in their possession. Employing a combined approach of machine learning and deep learning in fraud detection allows for the distinction between the authenticity of a prospective transaction and its fraudulent characteristics[5].

Machine learning stands out as the leading and most widely adopted technology due to its diverse applications and its efficiency in delivering highly accurate results. Machine learning is a technological paradigm that focuses on algorithms, enabling computers to learn and evolve through experience without the need for explicit programming. This technology finds utility across numerous domains, including healthcare for diagnosis and regression analysis.

Machine learning uses a blend of algorithms and statistical models, empowering computers to execute tasks without relying on rigid, predetermined programming. It operates by creating a model through training data and subsequently evaluating it on the trained model[6–10].

1.2. Key highlights

The main focus of this research project is to develop an exceptionally efficient deep learning system designed specifically for detecting credit card fraud. In summary, the objectives of this endeavor can be characterized as:

- Creating a credit card transaction fraud detection system through the utilization of recurrent neural networks (RNN).
- Improving the classifier's effectiveness requires the integration of advanced methods and the inclusion of supplementary stages, rather than maintaining it as an independent system.
- With good DL parameters, the model is been trained and tested with real-time transaction images.
- The usage of autoencoder and PSO gives additional angles for the low-complex classifier.
- The usage of the feature engineering method for doing the preprocessing of transaction records gives another dimension in terms of accuracy.
- By using advanced models and larger datasets, we are able to eradicate small datasets and overfitting issues.
- According to the results of the experiments, the proposed system surpasses the performance of the most advanced available models.

Organization of the paper: In section 1, we have previously discussed about the connection between deep learning and its application within the field of credit card fraud detection. Further, section 2 explain about literature review. Section 3 outlines the methodology employed, and section 4 provides an in-depth examination of the performance. The paper culminates with the conclusion in section 5.

2. Literature review

Habipour et al.[11] introduced three approaches to measure uncertainty: Monte Carlo dropout, ensemble, and ensemble Monte Carlo dropout. Additionally, the researchers evaluate the accuracy of their predictive uncertainty evaluations using a UQ confusion matrix and various performance metrics. Their results reveal that the ensemble approach is particularly adept at accurately capturing the uncertainty associated with the generated predictions. Additionally, they showcase how these novel UQ techniques provide valuable information beyond point predictions, thereby enhancing the effectiveness of fraud prevention measures.

Fanai and Abbasimehir[12] introduced a two-step framework designed for the detection of fraudulent

transactions. This framework integrates a deep autoencoder for representation learning and uses supervised deep learning methods. The outcomes of their experiments illustrate that this suggested method improves the performance of deep learning classifiers. Notably, the deep learning classifiers, when trained on data modified by the deep Autoencoder, surpass their original counterparts, which were trained on unaltered data, in all performance metrics. Moreover, models developed with the deep Autoencoder outshine those created using a dataset generated through principal component analysis (PCA) and surpass existing models in terms of performance.

Mienye and Sun[13] introduced a resilient deep-learning methodology that employs a stacking ensemble structure. This method utilizes long short-term memory (LSTM) and gated recurrent unit (GRU) neural networks as its core models, alongside a multilayer perceptron (MLP) functioning as the meta-learner. To maintain an equitable distribution of classes in the dataset, a hybrid approach combining the synthetic minority oversampling technique and the edited nearest neighbor method (SMOTE-ENN) was implemented. The experimental results indicate that the incorporation of this deep learning ensemble with the SMOTE-ENN technique yields outstanding outcomes, surpassing the performance of other frequently employed machine learning classifiers and techniques outlined in prior research. This led to sensitivity and specificity values of 1.000 and 0.997, respectively.

Gupta et al.[14] showcased the application of multiple classifiers and data balancing methods to address a notably imbalanced dataset. Through a thorough evaluation of the imbalanced dataset, they observed that the XGBoost classifier delivered exceptional results, achieving a precision score of 0.91 and an accuracy score of 0.99. To enhance key evaluation metrics such as precision, recall, F1-score, and accuracy, the researchers introduced various sampling techniques. Among these methods, Random oversampling emerged as the most effective strategy for addressing data imbalance issues. When implemented with the high-performing XGBoost model, it yielded a precision score of 0.99 and an accuracy score of 0.99.

Noviandy et al.[15] explored the utilization of machine learning, particularly the XGBoost (eXtreme Gradient Boosting) algorithm, in conjunction with data augmentation techniques for improving the detection of credit card fraud. Their research showcases the efficacy of these strategies in effectively tackling the issues related to imbalanced datasets, leading to enhanced accuracy in the identification of fraudulent transactions. By using historical transaction data and implementing methods like synthetic minority over-sampling technique-edited nearest neighbors (SMOTE-ENN), this research embraces a balanced strategy to enhance both precision and recall within the realm of fraud detection. The implications of these findings for modern financial management are profound, as they provide the potential to reinforce financial security, allocate resources more efficiently, and bolster customer trust in the midst of evolving fraudulent practices.

Btoush et al.[16] have conducted an in-depth analysis and synthesis of previously published research pertaining to the detection of cyber fraud in credit card transactions. Their focus was particularly on the use of machine learning and deep learning techniques in this endeavour. This thorough investigation encompassed 181 research articles that were published between 2019 and 2021, providing valuable insights for scholars in this field. It serves as a valuable reference for assessing the potential applicability of machine learning and deep learning approaches in the realm of identifying cyber fraud in credit card transactions. Furthermore, the review delves into significant challenges, identifies areas where research gaps exist, and acknowledges the inherent limitations in the field of credit card cyber fraud detection. Additionally, it provides guidance on potential directions for future research. By providing such a comprehensive assessment, their review contributes significantly to facilitating innovation projects in the realm of cyber fraud detection, benefiting both researchers and the banking industry.

3. Proposed methodology

Figure 2 provides a summary of the structural layout of the envisioned system, with a comprehensive examination of each stage provided in the subsequent sections.

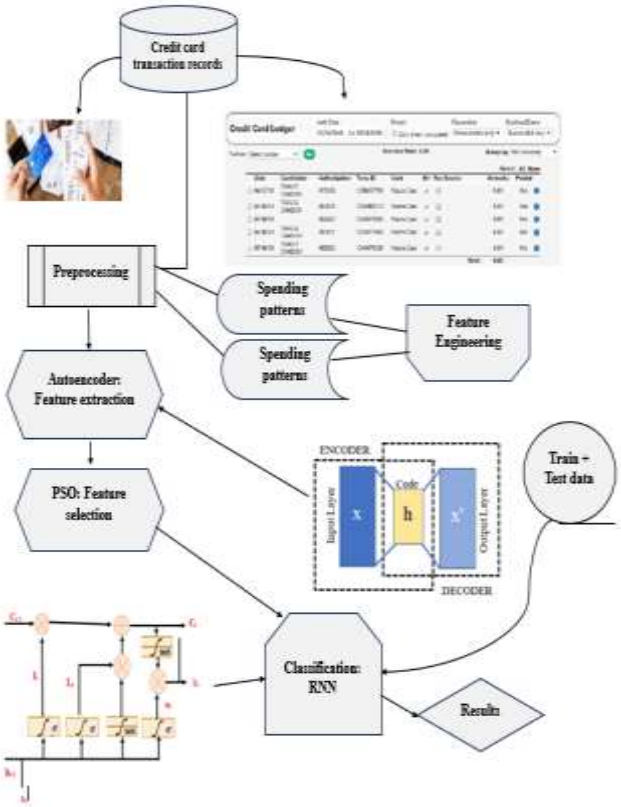

Figure 2. Proposed architecture overall outlook.

3.1. Data collection

In this research, we used an authentic dataset sourced from one of China's major commercial banks. To identify instances of fraudulent transactions, we used the information recorded by the bank's fraud investigation division. The dataset comprises a total of 153,685 transaction records, and it's worth noting that there is a significant imbalance, with only 2046 (less than 1.5%) of these transactions being categorized as fraudulent. To get our data ready for analysis, we conducted thorough data preprocessing. This involved the careful removal of duplicates, outliers, and transactions that were not customer-initiated, such as reversal transactions. This diligent preprocessing yielded a clean dataset comprising 114,779 transactions, forming the basis for our experimental investigation. To support our research, we partitioned this dataset into two distinct segments: one for training and validation, which consisted of 61,735 transactions, and a separate test dataset comprising 53,044 transactions. It's worth noting that the training set contained 866 instances of fraudulent transactions, while the testing set included 1176 such transactions[17].

3.2. Preprocessing

When creating an algorithm to detect credit card fraud, the initial feature set comprises data related to individual transactions. Previous research indicates a remarkable consistency in the types of data collected, which can be attributed to the necessity of adhering to international financial reporting standards during credit card transactions.

3.2.1. Spending patterns

Many studies rely solely on the raw features for their analysis. Nevertheless, a solitary transaction's details

are inadequate for identifying fraudulent transactions[18]. The reason for this is that relying solely on raw features fails to consider essential data, like consumer spending behaviours, which are commonly utilized in commercial fraud detection systems.

In addressing this issue, a fresh set of attributes was introduced in several works with the aim of incorporating information from the customer's most recent credit card transaction into the prediction process. The primary goal is to identify highly divergent continuous transactions within a customer's purchase history. These newly introduced attributes encompass the following: the duration since the last transaction, the transaction amount from the previous occurrence, and the country where the transaction took place. Nevertheless, these attributes do not encompass consumer behaviour beyond the client's latest transaction, resulting in an incomplete customer profile.

A more concise approach to understanding a customer's spending behaviour involves generating specific attributes through a transaction aggregation method, as originally introduced in the study of Lucas et al.[19]. This procedure entails categorizing transactions conducted during a defined period. Initially, they are grouped based on card or account numbers, and then further sorted by transaction type, merchant category, location (country), or other pertinent criteria. Afterwards, it calculates either the transaction count or the total expenditure linked to these transactions. Numerous research works have adopted this approach for cyber fraud detection in early stages[20].

When consolidating customer transactions, a critical question arises regarding the extent of aggregation. This relates to the idea that the additional benefit of acquiring new information may dwindle over time. As discussed in a previous source[21], it is stressed that combining 101 transactions is unlikely to provide substantially more insights than combining 100 transactions. The reason for this is that over time, the significance of information decreases since customer spending patterns are anticipated to evolve with the passage of years. To address this challenge, Whitrow and associates propose the notion of a set period that can be customized to 24, 60, or 168 h. The process of merging attributes involves selecting transactions that occurred within the past t_p hours for each transaction i within the dataset S.

$$S_{agg} \equiv TRX_{agg}(S, i, t_p) = \left\{X_l^{amt} | \left(X_l^{id} = X_i^{id}\right) \wedge \left(\text{hours}(X_i^{\text{time}}, X_l^{\text{time}}\right) < t_p\right\}_{l=1}^{N} \quad (1)$$

In the context that follows, let's focus on the function TRX_{agg}, which is responsible for producing a subset of the set S linked to transaction i that occurred during the time frame t_p. We use N to denote the size of set S, indicated as $|S|$. In this particular scenario, $X^{time}i$ denotes the timestamp of transaction i, $x^{amt}i$ corresponds to the amount linked with transaction i, and $x^{id}i$ signifies the customer identification number for transaction i. Additionally, we use the function hours($t1$, $t2$) to determine the time difference in hours between two timestamps, $t1$ and $t2$. Following this, our next step is to compute the following metrics: the count of transactions and the total transaction value within the last t_p hours.

$$X_i^{a1} = \left|S_{agg}\right|,$$

and

$$X_i^{a2} = \sum_{x^{amt} \in S_{agg}} X^{amt} \quad (2)$$

We observe that this form of aggregation is insufficient because it fails to consider the amalgamation of various attributes. As an illustration, it's crucial to assess not just the total number of transactions but also to categorize them based on particular parameters. These criteria may include transactions conducted within the past "t_p" hours, transactions originating from the same country, and transactions falling under the same transaction category. To calculate these characteristics, we initiate the expansion of Equation (6) in the following manner.

$$S_{agg2} \equiv TRX_{agg}(S, i, t_p, cond_1, cond_2)s$$
$$= \{X_l^{amt} | (X_l^{id} = X_i^{id}) \wedge (hours(X_i^{time}, X_l^{time}) < t_p) \wedge (X_l^{cond_1} X_i^{cond_1}) (X_l^{cond_2} X_i^{cond_2})\}_{l=1}^{N} \quad (3)$$

where $cond_1$ and $cond_2$ can each represent any of the transaction features listed in **Table 1**, the calculation of these features is as follows:

$$x_i^{a3} = |S_{agg2}|$$

and

$$x_i^{a4} = \sum_{x^{amt} \in S_{agg2}} x^{amt} \quad (4)$$

3.2.2. Time features

Even when utilizing aggregated features, undisclosed information, especially regarding the timing of transactions, may remain concealed. This curiosity arises from the expectation that customers tend to carry out transactions during similar hours. The challenge emerges when delving into transaction timing, particularly when assessing a metric like the mean transaction time, as the conventional arithmetic mean is found to be inadequate. As depicted in **Figure 3**, the arithmetic mean overlooks the cyclic or repetitive nature of the time variable. For example, when calculating the average transaction time for four transactions occurring at 2:00, 3:00, 22:00, and 23:00, the result is 12:30, without considering the recurring pattern. This outcome appears counterintuitive because there were no transactions conducted around that time.

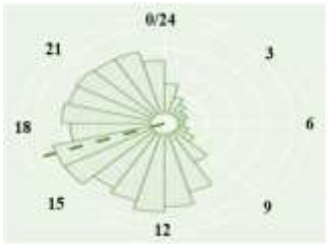

Figure 3. Time features and their arithmetic schema.

In order to address this constraint, our suggestion is to control the timing of the transaction by considering it as a cyclic variable and making use of the von Mises distribution[22]. The von Mises distribution, often referred to as the periodic normal distribution, characterizes the circular distribution of variables that exhibit a normal distribution pattern. In the context of a dataset $D = \{t_1, t_2, \cdots, t_N\}$, the von Mises distribution is defined as follows:

$$D \sim vonmises\left(\mu_{vM}, \frac{1}{\sigma_{vM}}\right), \quad (5)$$

Substitute "where μ_{vM} and σ_{vM}" with "μ_{vM}, which signifies the periodic mean, and σ_{vM}, which stands for the periodic standard deviation."

Our main objective is to create a confidence interval (CI) for the transaction time. To achieve this aim, we begin by choosing a group of transactions executed by a single client within the preceding t_p hours.

$$S_{per} \equiv TRX_{vM}(S, i, t_p) = \{X_l^{time} | (X_l^{id} = X_i^{id}) \wedge (hours(X_i^{time}, X_l^{time}) < t_p\}_{l=1}^{N} \quad (6)$$

Following this, the probability distribution function for the timing of the transaction set is determined as follows:

$$X_i^{time} \sim vonmises\left(\mu_{vM}(S_{per}), \frac{1}{\sigma_{vM}(S_{per})}\right) \quad (7)$$

3.3. Feature extraction: AE

Autoencoders (AEs) are a type of unsupervised representation learning method designed to capture the underlying data distribution by discovering a collection of hidden representations, often referred to as embeddings. These embeddings, represented by the model's hidden-layer units or low-dimensional features, effectively encapsulate the most significant structural aspects of the original data distribution[22,23]. The encoder part of the autoencoder captures a diverse set of nonlinear characteristics from the sensory information, whereas the decoder's role is to utilize these extracted features to recreate the original signal. This approach is motivated by the ability of autoencoders to identify anomalies, primarily because anomalies are infrequent and exhibit substantial deviations from the typical patterns observed in healthy data. The primary objective when training this model is to teach it to recognize and reproduce the normal behavior of a given system, without explicitly identifying or flagging anomalies.

The network comprises two essential elements: the encoder and the decoder, both employing long short-term memory (LSTM) units. LSTMs are part of the recurrent neural network (RNN) family and possess the ability to incorporate temporal data into the network while maintaining a hidden state vector, which functions as a storage for prior information.

a) Encoder: We examine the input data sequence denoted as $X = (x(1), x(2), \ldots, x(N))$, where $x(N) = (x(N)1, x(N)2, \ldots, x(N)T)$ belongs to the space $RTxd$. This implies that for every index N, there exists a time-series sequence comprising T time steps, with each step having a d-dimensional component. To handle the variable input sequence and capture the sequential information embedded in the time-series data, we employ an RNN. This explanation is applicable to every individual sequence as follows:

$$C_t' = \tanh(W_C x_t + R_C h_{t-1} + b_C) \tag{8}$$

In the previous step, we had the memory state ($Ct0$), input vector (xt), and output vector ($ht - 1$), while the input weights are represented by W_C, recurrent weights by R_C, and the bias as b_C. A hyperbolic tangent ($\tanh$), which is a non-linear activation function, is used to produce output values that are limited to a range between −1 and 1.

The input sequences are subjected to encoding in the encoder component of the LSTM network. In this process, LSTM units are utilized to encode either a single input sequence or a group of sequences, with their hidden states being continuously updated. The result of this encoding procedure is subsequently presented in the following manner:

$$h_t = \sigma_{\emptyset}^{e}(x_t, h_{t-1}) \tag{9}$$

The i-th LSTM encoder's output, labelled as h_t, is determined by the encoder's parameter set φ. To address the issues related to vanishing and exploding gradients, the ReLU activation function denoted as σ is chosen for both the encoder and decoder.

b) Decoder: The encoder initially transforms the vibration signal into lower-dimensional embeddings, which are subsequently employed by the decoder to recreate the initial signal. Ultimately, the encoder's output serves as the input for the LSTM-decoder network.

$$h_t' = \sigma_{\varphi}^{d}(h_t, h_{t-1}'); x_t' = \sigma(h_t') \tag{10}$$

In this context, the decoder's parameter set is denoted as ϕ, and x_0 represents the reconstructed input used for calculating the reconstruction error (RE), also referred to as the mean squared error (MSE). This error is important for updating both the encoder and decoder parameters within the network and, subsequently, for determining anomaly scores. The autoencoder (AE) model should be capable of faithfully reproducing the original signal while being resilient to the effects of noise and the specifics of the training data. The model's flexibility enables it to acquire a versatile understanding of the data. During the training process, the model's main goal is to understand the regular behaviour of the system, deliberately leaving out any unusual data points

from the training set. Consequently, when there is a departure from the standard behaviour in the prediction phase, it leads to an increased reconstruction error (RE). By monitoring this increase in the reconstruction error, it becomes feasible not only to identify anomalies but also to forecast potential faults by recognizing the point at which the RE begins to escalate, referred to as the degradation point.

3.4. Feature selection: PSO

PSO employs a collective of particles that continually update their positions as they search for the best solution. In the quest for the best solution, every particle refines its position by taking into account both its individually recognized best position (*pbest*) and the overall best-known position within the entire group (*gbest*)[24–26].

$$pbest(i,t) = \arg\min\left[f\left(P_i(k)\right)\right], \begin{matrix} i \in \{1,2,\dots,N_p\} \\ k = 1,\dots,t \end{matrix}$$
$$pbest(i,t) = \arg\min\left[f\left(P_i(k)\right)\right], \begin{matrix} i = 1,\dots,N_p \\ k = 1,\dots,t \end{matrix} \quad (11)$$

In this specific situation, it's crucial to elucidate that "i" is used to represent the particle's index, "N_p" signifies the total number of particles, "t" denotes the current iteration count, "f" stands for the fitness function, and "V" and "p" correspond to the velocity and position of the particle, accordingly. The equations for updating the particles' velocity and position are as follows:

$$V_i(t+1) = \omega V_i(t) + c_1 r_1\left(pbest(i,t) - P_i(t)\right) + c_2 r_2\left(gbest(t) - P_i(t)\right),$$
$$P_i(t+1) = P_i(t) + V_i(t+1) \quad (12)$$

In the context, V represents velocity, w serves as the inertia weight responsible for balancing global exploration and local exploitation, while r_1 and r_2 are random variables uniformly distributed within a specified range. Additionally, c_1 and c_2 are constant positive parameters known as "acceleration coefficients."

Establishing an upper limit on the velocity parameter, known as "velocity clamping," is a widely practised technique to prevent particles from exceeding the boundaries of the search space. An alternative approach, referred to as the "constriction coefficient" method, was introduced by Clerc and Kennedy based on a theoretical analysis of swarm dynamics, which also imposes constraints on particle velocities. In Equation (2), the initial segment, designated as "inertia," assimilates the particles' prior velocities, imparting the essential momentum for their exploration of the search space. The subsequent segment, referred to as the "cognitive" element, mirrors the individual decision-making of each particle, prompting them to approach their individually optimal positions that have been uncovered. The final component, the "cooperation" element, signifies the collective effort of the particles to discover the globally optimal solution[19].

3.5. Classifier RNN

Long short-term memory (LSTM) is a unique and crucial architectural variation in the field of deep learning, explicitly tailored for the modelling of time series data, as depicted in **Figure 4**. Unlike conventional feedforward neural networks, LSTM makes use of recurrent connections among its hidden units, where each connection is associated with particular time steps. This characteristic empowers the network to capture extended dependencies within sequential data, making it proficient in predicting transaction labels by considering past transaction sequences. LSTMs were introduced as a solution to address the challenging issue of vanishing and exploding gradients often encountered during the training of conventional RNNs, as discussed in reference[20].

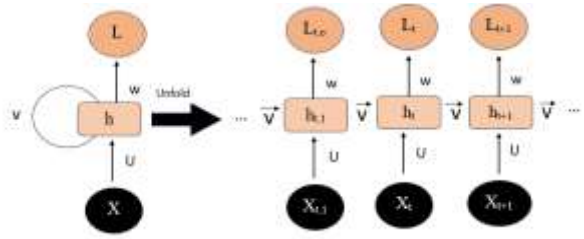

Figure 4. LSTM structure.

An LSTM unit comprises a memory cell responsible for retaining information, and this cell's state is modified by three distinct gates: the input gate, the forget gate, and the output gate. The memory cell can retain data for varying lengths of time, while the trio of gates regulates the ingress and egress of information. The schematic representation of the LSTM unit can be observed in **Figure 5**.

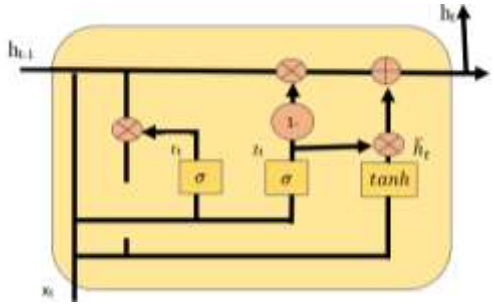

Figure 5. Units of LSTM.

At the time '*t*', 'x_t' represents the input data supplied to the LSTM cell, 'h_{t-1}' corresponds to the prior time step's LSTM cell output, 'C_t' stands for the memory cell's contents, and 'h_t' signifies the present output from the LSTM cell. The LSTM unit's functioning can be broken down into the subsequent phases:

The initial process, as outlined in Equation (3), involves computing the memory cell value, denoted as *ct.* This computation depends on the weight matrix W_c and the bias coefficient b_c.

$$\tilde{c} = \tanh(W_c[h_{t-1}, x_t] + b_c) \tag{13}$$

Determine the input gate value denoted as i_t, which plays a pivotal role in regulating the transformation of the current input data into the state value of the memory cell. The input gate is computed by employing the sigmoid function Sigma, where the weight matrix W_i and bias b_i are integral elements of the input gate equation.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{14}$$

Find the weight matrix (W_f) and bias (B_f) for calculating the forget gate value. The forget gate is crucial for controlling how previous information is integrated into the current state of the memory cell, and its mathematical representation is given by:

$$f_t = \sigma\big(W_f[h_{t-1}, x_t] + b_f\big) \tag{15}$$

Find the current memory cell value, labelled as C_t, using the equation provided, where c_{t-1} represents the state value of the LSTM unit that came before it.

$$c_t = f_t * c_f + i_t * \tilde{c}_t \tag{16}$$

In this particular scenario, we represent the dot product with "*", and the memory cell's update depends on the prior cell's condition, the candidate cell, and is controlled by the input gate and the forget gate.

Determine the output gate value denoted as "o_t," which is in charge of regulating the output state of the memory cell, using the weight matrix W_o and bias term B_o. The formula for calculating the output gate is as follows:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{17}$$

Ultimately, compute the LSTM unit's output in accordance with the given equation:

$$h_t = o_t * \tanh(c_t) \tag{18}$$

Attention mechanisms have gained prominence in contemporary deep learning research, particularly in fields like computer vision and language translation[21,22]. They have proven to be a powerful tool for delivering impressive outcomes by identifying and prioritizing crucial information. This mechanism is designed to concentrate on the most pertinent data, bypassing the need to process the entirety of it, thereby enhancing neural processing capabilities[23].

To illustrate the attention mechanism, a closer look at an RNN encoder-decoder framework is given below. In this setup, an encoder processes an input sequence of vectors, denoted as $x = (x1, x2, ..., xn)$, and transforms them into a resulting vector c. This concept is frequently elucidated in the context of an RNN structure and can be formulated as follows:

$$S_t = f(x_t, S_{t-1}, c_t) \tag{19}$$

and:

$$c = q(S_1, \ldots, S_n) \tag{20}$$

In the era of recurrent neural networks (RNNs), where "s_t" denotes the hidden state and "c" represents the vector derived from these hidden states, the attention model establishes a strong connection between the context vector "C_t" and a set of annotations ($h1, h2, ..., hn$) generated by an encoder based on the input sequence. These annotations contain information about the entire input sequence, with a specific emphasis on segments close to the "t-th" word in the input sequence, referred to as "h_t". Further details on this topic are provided in subsequent explanations. **Figure 6** visually illustrates the attention mechanism within the neural network, showing how the context vector "c_t" is computed as a weighted sum of these annotations "h_t".

$$c_t = \sum_{j=1}^{n} a_{tj} h_j \tag{21}$$

where the weight a_{tj} of each annotation h_t is given by:

$$a_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{n} \exp(e_{tk})} \tag{22}$$

In which:

$$e_{tj} = a(S_{t-1}, h_j) \tag{23}$$

The alignment model, denoted as $a(S_{t-1}, h_j)$, characterizes how well the elements around position j in the inputs align with the elements at position t in the outputs. The alignment score is determined through the fusion of the RNN hidden state, S_{t-1}, with the j-th annotation, h_j, extracted from the input sentence. The attention mechanism provides a neural network with the ability to focus on particular elements within its input, consistently choosing the most relevant ones. As depicted in **Figure 6**, the attention mechanism is specifically crafted to emphasize the key inputs from the input sequences $x1, x2, ..., xn$, using the weighting factor alpha t_j.

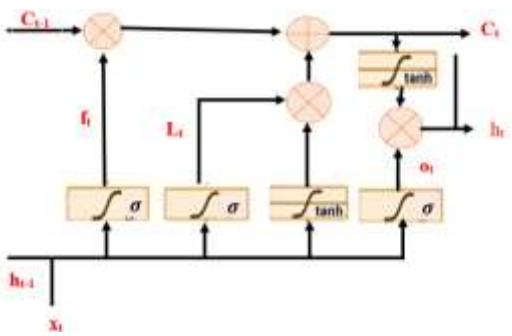

Figure 6. Attention mechanism.

4. Performance analysis

The proposed approach has been applied using hardware components, including a GTX NV graphics card, a 1 TB hard drive, and the Windows 10 operating system. In terms of software, the method utilizes Python along with open-source libraries for machine learning framework development. It utilizes Google Colab, an open-source platform for developing machine learning models. As we already explained the working equations in above section, the parameters of the RNN include the weights $W_{xh} \in \mathbb{R}^{d*h}, W_{hh} \in \mathbb{R}^{h*h}$ and the bias $b_h \in \mathbb{R}^{1*n}$ of the hidden layer, together with the weights $W_{hq} \in \mathbb{R}^{h*q}$ and the bias $b_q \in \mathbb{R}^{1*q}$ of the output layer. It is worth mentioning that even at different time steps, RNNs always use these model parameters. Therefore, the parametrization cost of an RNN does not grow as the number of time steps increases[24–32]. Data selection held in a way that labelled data partitioning chosen to mirror GBT models as closely as possible. The features include such as Current Transaction details; some aggregations of historical transactions related to merchants still used (e.g., count, mean, etc.), customer details. We also create a 'time elapsed from previous transaction' feature to account for irregular time intervals between transactions. Dealing with imbalance data shows sampling and training on sub-sequences matching the desired end-label allows for precise data balancing. We create training batches by sampling sub-sequences ending with fraud/non-fraud with a 50/50 split. The sub-sequences starting point is fixed. We also used Adam optimizer for training the datasets with a learning rate of exponential decay.

The experimental investigation covers a variety of performance measures, including accuracy, sensitivity, specificity, precision, recall, F1-score, detection rate, true positive rate (TPR), and false positive rate (FPR). Furthermore, we've performed testing with different numbers of transaction inputs to prevent overfitting and tackle challenges associated with imbalanced small datasets. In **Table 1**, we can find a comprehensive comparison between our proposed system and other cutting-edge models referred to as "L". In that, our proposed system (RNNAEPSO) has a greater performance due to training and testing of the classifier with the boostage in with other stages (**Figure 7**).

Table 1. Comparison analysis of accuracy, sensitivity, specificity.

Models	Accuracy	Sensitivity	Specificity
L1	76	85	88
L2	84	87	89
L3	86	81	86
L4	83	79	80
L5	88	92	89
RNNAEPSO	97	98	98

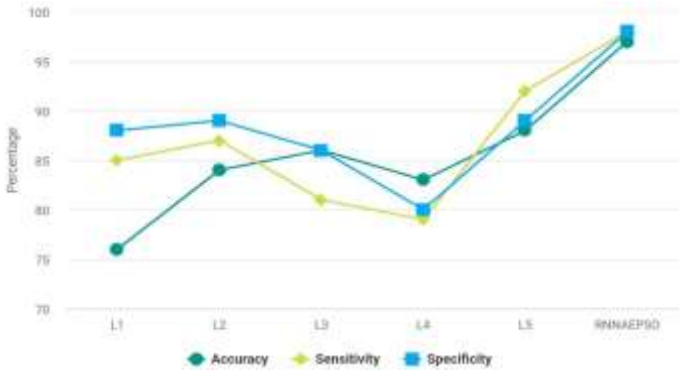

Figure 7. Models vs. accuracy, sensitivity, specificity.

Further, **Table 2** presents a contrast in precision, recall, and F1-score across different models, emphasizing the exceptional performance of the suggested system.

Table 2. Comparison analysis of precision, recall, F1-score.

Models	Precision	Recall	F1-score
L1	72	77	83
L2	76	82	82
L3	83	84	85
L4	85	84	86
L5	81	86	89
RNNAEPSO	86	90	92

This advantage stems from the unique approach of not training the model in isolation, but rather enhancing its knowledge through a complex classifier i.e., RNN[33–35]. By using of autoencoder with its convolutional approach and improving selection ability with PSO, the efficacy of the classifier increased than expected (**Figure 8**).

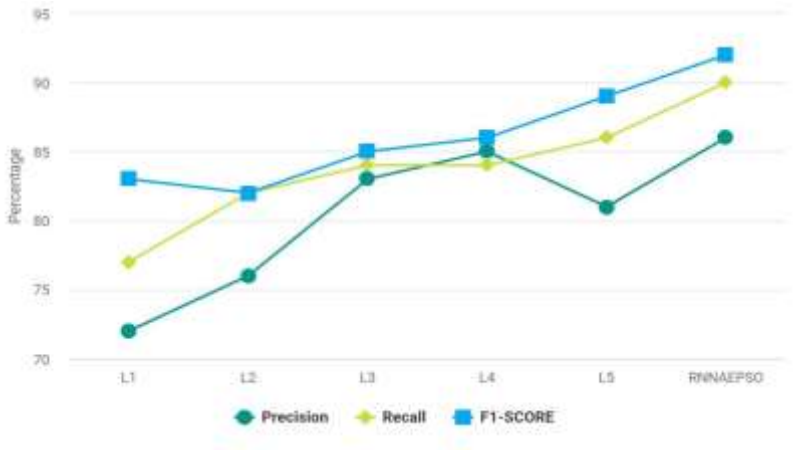

Figure 8. Models vs. precision, recall, F1-score.

Table 3 shows the comparison analysis of the proposed system over detection rate, TPR, and FPR. By overcoming the issue of overfitting and small data imbalance, the system was able to train under greater parameters and also with quality processed inputs with the help of feature engineering (**Figure 9**).

Table 3. Comparison analysis: Detection rate, TPR, FPR.

Models	Detection rate	TPR	FPR
L1	75	77	23
L2	72	75	25
L3	82	86	14
L4	85	82	18
L5	88	87	13
RNNAEPSO	94	90	10

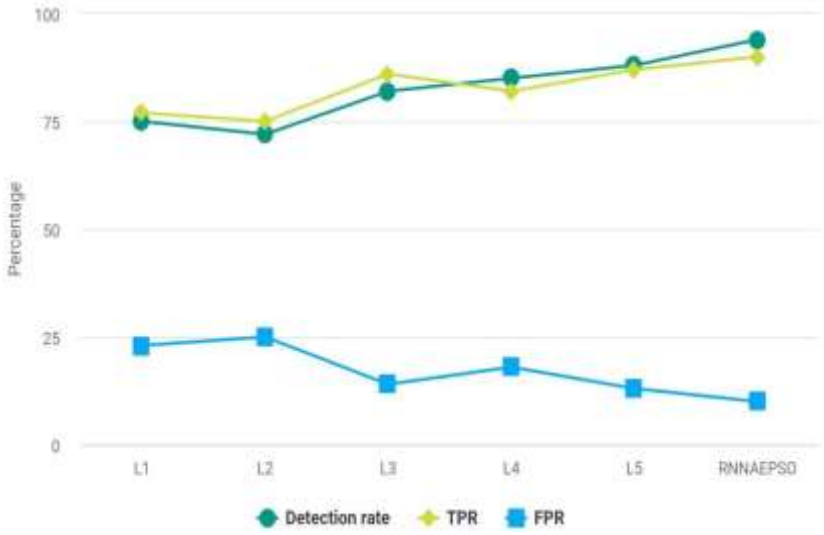

Figure 9. Models vs. detection rate, TPR, FPR.

Discussion

Our study focused on detection of fraud occurrence in credit card usage with help of advance deep learning strategies and techniques our proposed model to foresee the mark of transaction in the wake of having seen a few exchanges that go before it. Each dataset is separated into three sets. The principal 70% subset of information is the training set utilized for preparing the models, the second 15% subset of information is the approval set utilized for approving the classifiers to keep away from overfitting and work on model execution and the last 15% test subset of information is utilized to test the network speculation.

The exactness and review plots for the two models applied for instance over our dataset were introduced in **Figures 7** and **8** from which we see that our model accomplished the more prevalent precision and sensitivity rates. This huge improvement is on the grounds that by utilizing consideration component, more significant examples can be extricated from grouping exchanges which permit the succession classifier to naturally zero in on the information things that are the most important to the order task by an information driven weighted normal of every exchange contained in the arrangement which brings about a superior recognition execution.

Besides, to feature the classification execution of our proposed model, as far as responsiveness, we present a representation of the disarray framework performed on each model applied for instance over our dataset from which we show that our proposed model has a decent capacity to limit the quantity of false classified delegated typical and get the uncommon fake transaction, which is important, all things considered, for monetary specialist organizations. Too, to evaluate the examination of our exploratory outcomes, we contrasted our work and cutting edge extortion models recorded in above tables. The significant justification behind choosing these models is that they display promising exhibitions and they utilize the little dataset and not yet live depicted in this work, subsequently making the correlation more reasonable and dependable. From the above tables, it shows the exhibition upsides of each pre-owned model, in term of exactness, accuracy and responsiveness. The last measurement is of high significance in extortion discovery space, since monetary establishments are intrigued more with regards to identifying misrepresentation cases that might happen, to safeguard customers' inclinations and lessen the weighty yearly monetary misfortunes brought about by extortion.

5. Conclusion

In conclusion, this research paper has discussed a comprehensive approach to identifying credit card fraud, incorporating a combination of advanced methods including recurrent neural networks (RNN), Autoencoders, feature engineering, and particle swarm optimization (PSO). Through the careful integration of

these techniques, the study has achieved an impressive 97% accuracy in detecting fraudulent transactions. This achievement underscores the efficacy of combining deep learning models, data preprocessing, and optimization algorithms to enhance the security and reliability of financial transactions. The results highlight the potential for real-world applications, illustrating the significance of interdisciplinary efforts in the ongoing battle against credit card fraud.

Author contributions

Conceptualization, AL, and AKT; methodology, AKT; software, AL; validation, AKM, and AKT; formal analysis, AKT; investigation, AKT and GS; resources, AKT; data curation, AKT and SKA; writing—original draft preparation, AKT; writing—review and editing, AKM; visualization, GS and SKA; supervision, AKT; project administration, SKA. 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. Awoyemi JO, Adetunmbi AO, Oluwadare SA. Credit card fraud detection using machine learning techniques: A comparative analysis. 2017 International Conference on Computing Networking and Informatics (ICCNI). 2017. pp. 1-9. doi: 10.1109/iccni.2017.8123782
2. Dornadula VN, Geetha S. Credit Card Fraud Detection using Machine Learning Algorithms. Procedia Computer Science. 2019; 165: 631-641. doi: 10.1016/j.procs.2020.01.057
3. Roy A, Sun J, Mahoney R, et al. Deep learning detecting fraud in credit card transactions. 2018 Systems and Information Engineering Design Symposium (SIEDS). 2018. pp. 129-134. doi: 10.1109/sieds.2018.8374722
4. Nguyen TT, Tahir H, Abdelrazek M, Babar A. Deep learning methods for credit card fraud detection. 2020. arXiv preprint arXiv:2012.03754
5. Sailusha R, Gnaneswar V, Ramesh R, et al. Credit Card Fraud Detection Using Machine Learning. 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). 2020. pp. 1264-1270. doi: 10.1109/iciccs48265.2020.9121114
6. 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
7. Pumsirirat A, Liu Y. Credit Card Fraud Detection using Deep Learning based on Auto-Encoder and Restricted Boltzmann Machine. International Journal of Advanced Computer Science and Applications. 2018; 9(1). doi: 10.14569/ijacsa.2018.090103
8. Najadat H, Altiti O, Aqouleh AA, et al. Credit Card Fraud Detection Based on Machine and Deep Learning. 2020 11th International Conference on Information and Communication Systems (ICICS). 2020. pp. 204-208. doi: 10.1109/icics49469.2020.239524
9. Azhan M, Meraj S. Credit Card Fraud Detection using Machine Learning and Deep Learning Techniques. 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS). 2020. pp. 514-518. doi: 10.1109/iciss49785.2020.9316002
10. Pillai TR, Hashem IAT, Brohi SN, et al. Credit Card Fraud Detection Using Deep Learning Technique. 2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA). 2018. pp. 1-6. doi: 10.1109/icaccaf.2018.8776797
11. Habibpour M, Gharoun H, Mehdipour M, et al. Uncertainty-aware credit card fraud detection using deep learning. Engineering Applications of Artificial Intelligence. 2023; 123: 106248. doi: 10.1016/j.engappai.2023.106248
12. 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
13. Mienye ID, Sun Y. A Deep Learning Ensemble with Data Resampling for Credit Card Fraud Detection. IEEE Access. 2023; 11: 30628-30638. doi: 10.1109/access.2023.3262020
14. Gupta P, Varshney A, Khan MR, et al. Unbalanced Credit Card Fraud Detection Data: A Machine Learning-Oriented Comparative Study of Balancing Techniques. Procedia Computer Science. 2023; 218: 2575-2584. doi: 10.1016/j.procs.2023.01.231
15. 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
16. Btoush EALM, Zhou X, Gururajan R, et al. A systematic review of literature on credit card cyber fraud detection

using machine and deep learning. PeerJ Computer Science. 2023; 9: e1278. doi: 10.7717/peerj-cs.1278
17. Varmedja D, Karanovic M, Sladojevic S, et al. Credit Card Fraud Detection - Machine Learning methods. 2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH). 2019. pp. 1-5. doi: 10.1109/infoteh.2019.8717766
18. Bahnsen AC, Aouada D, Stojanovic A, et al. Feature engineering strategies for credit card fraud detection. Expert Systems with Applications. 2016; 51: 134-142. doi: 10.1016/j.eswa.2015.12.030
19. Lucas Y, Portier PE, Laporte L, et al. Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs. Future Generation Computer Systems. 2020; 102: 393-402. doi: 10.1016/j.future.2019.08.029
20. 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
21. 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
22. Yu X, Li X, Dong Y, Zheng R. A deep neural network algorithm for detecting credit card fraud. In 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE). 2020. pp. 181-183.
23. Tyagi AK, Abraham A. (editors). Recent Trends in Blockchain for Information Systems Security and Privacy, 1st ed. CRC Press; 2021. doi: 10.1201/9781003139737
24. Zou J, Zhang J, Jiang P. Credit card fraud detection using autoencoder neural network. arXiv preprint arXiv:1908.11553
25. Li C, Ding N, Zhai Y, et al. Comparative study on credit card fraud detection based on different support vector machines. Intelligent Data Analysis. 2021; 25(1): 105-119. doi: 10.3233/ida-195011
26. Mniai A, Tarik M, Jebari K. A Novel Framework for Credit Card Fraud Detection. IEEE Access. 2023; 11: 112776-112786. doi: 10.1109/access.2023.3323842
27. Owolafe O, Ogunrinde OB, Thompson AFB. A long short-term memory model for credit card fraud detection. In Artificial Intelligence for Cyber Security: Methods, Issues and Possible Horizons or Opportunities. Cham: Springer International Publishing; 2021. pp. 369-391.
28. Benchaji I, Douzi S, Ouahidi BE. Credit Card Fraud Detection Model Based on LSTM Recurrent Neural Networks. Journal of Advances in Information Technology. 2021; 12(2): 113-118. doi: 10.12720/jait.12.2.113-118
29. Roseline JF, Naidu GBSR, Pandi VS, et al. Autonomous credit card fraud detection using machine learning approach. Computers and Electrical Engineering. 2022; 102: 108132. doi: 10.1016/j.compeleceng.2022.108132
30. Marie-Sainte SL, Alamir MB, Alsaleh D, et al. Enhancing credit card fraud detection using deep neural network. In: Intelligent Computing: Proceedings of the 2020 Computing Conference. Springer International Publishing, 2020. pp. 301-313.
31. Sarumathi R, Saraswathy G. Comparative Analysis of Classification Techniques for Credit Card Fraud Detection. History Manuscript Reference. 2022; 9: 16-25.
32. Forough J, Momtazi S. Sequential credit card fraud detection: A joint deep neural network and probabilistic graphical model approach. Expert Systems. 2021; 39(1). doi: 10.1111/exsy.12795
33. Erondu UI, Asani EO, Arowolo MO, et al. An Encryption and Decryption Model for Data Security Using Vigenere With Advanced Encryption Standard. In: Tyagi A (editor). Using Multimedia Systems, Tools, and Technologies for Smart Healthcare Services. IGI Global; 2023. pp. 141-159. doi: 10.4018/978-1-6684-5741-2.ch009
34. Tyagi AK, Abraham A (editor). Recurrent Neural Networks (1st ed.). CRC Press; 2022. doi: 10.1201/9781003307822
35. Sai GH, Tyagi AK, Sreenath N. Biometric Security in Internet of Things Based System against Identity Theft Attacks. In: 2023 International Conference on Computer Communication and Informatics (ICCCI); 2023; Coimbatore, India. pp. 1-7. doi: 10.1109/ICCCI56745.2023.10128186