mathematics

MDPI

Article

RaKShA: A Trusted Explainable LSTM Model to Classify Fraud Patterns on Credit Card Transactions

Jay Raval [1], Pronaya Bhattacharya [2], Nilesh Kumar Jadav [1], Sudeep Tanwar [1,*], Gulshan Sharma [3], Pitshou N. Bokoro [3,*], Mitwalli Elmorsy [4], Amr Tolba [5] and Maria Simona Raboaca [6,7]

1 Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad 382481, Gujarat, India; 21mced18@nirmauni.ac.in (J.R.); 21ftphde53@nirmauni.ac.in (N.K.J.)
2 Department of Computer Science and Engineering, Amity School of Engineering and Technology, Research and Innovation Cell, Amity University, Kolkata 700157, West Bengal, India; pbhattacharya@kol.amity.edu
3 Department of Electrical Engineering Technology, University of Johannesburg, Johannesburg 2006, South Africa; gulshans@uj.ac.za
4 Private Law Department, Faculty of Law and Political Science, King Saud University, Riyadh 12584, Saudi Arabia
5 Computer Science Department, Community College, King Saud University, Riyadh 11437, Saudi Arabia; atolba@ksu.edu.sa
6 Doctoral School, University Politehnica of Bucharest, Splaiul Independentei Street No. 313, 060042 Bucharest, Romania; simona.raboaca@icsi.ro
7 National Research and Development Institute for Cryogenic and Isotopic Technologies—ICSI Rm. Vâlcea, Uzinei Street, No. 4, P.O. Box 7, Râureni, 240050 Râmnicu Vâlcea, Romania
* Correspondence: sudeep.tanwar@nirmauni.ac.in (S.T.); pitshoub@uj.ac.za (P.N.B.)

Abstract: Credit card (CC) fraud has been a persistent problem and has affected financial organizations. Traditional machine learning (ML) algorithms are ineffective owing to the increased attack space, and techniques such as long short-term memory (LSTM) have shown promising results in detecting CC fraud patterns. However, owing to the black box nature of the LSTM model, the decision-making process could be improved. Thus, in this paper, we propose a scheme, *RaKShA*, which presents explainable artificial intelligence (XAI) to help understand and interpret the behavior of black box models. XAI is formally used to interpret these black box models; however, we used XAI to extract essential features from the CC fraud dataset, consequently improving the performance of the LSTM model. The XAI was integrated with LSTM to form an explainable LSTM (X-LSTM) model. The proposed approach takes preprocessed data and feeds it to the XAI model, which computes the variable importance plot for the dataset, which simplifies the feature selection. Then, the data are presented to the LSTM model, and the output classification is stored in a smart contract (SC), ensuring no tampering with the results. The final data are stored on the blockchain (BC), which forms trusted and chronological ledger entries. We have considered two open-source CC datasets. We obtain an accuracy of 99.8% with our proposed X-LSTM model over 50 epochs compared to 85% without XAI (simple LSTM model). We present the gas fee requirements, IPFS bandwidth, and the fraud detection contract specification in blockchain metrics. The proposed results indicate the practical viability of our scheme in real-financial CC spending and lending setups.

Keywords: Explainableartificial intelligence; credit card frauds; deep learning; long short-term memory; fraud classification

MSC: 91G45

Citation: Raval, J.; Bhattacharya, P.; Jadav, N.K.; Tanwar, S.; Sharma, G.; Bokoro, P.N.; Elmorsy, M.; Tolba, A.; Raboaca, M.S. *RaKShA*: A Trusted Explainable LSTM Model to Classify Fraud Patterns on Credit Card Transactions. *Mathematics* **2023**, *11*, 1901. https://doi.org/10.3390/math11081901

Academic Editors: Snezhana Gocheva-Ilieva, Hristina Kulina and Atanas Ivanov

Received: 12 March 2023
Revised: 14 April 2023
Accepted: 15 April 2023
Published: 17 April 2023

1. Introduction

Modern credit-card (CC)-based applications are web/mobile-driven, and the customer base has shifted toward electronic payment modes. The online repayment modes for CC bring users flexibility and quality of service (QoS). Still, on the downside, it also opens the

doors for malicious intruders to intercept the web channels. Thus, recent statistics have suggested a surge in security attacks in CC ecosystems and payment gateway services [1,2]. These attacks mainly include banking frauds, attacks on credit and debit payments of CCs due to unsecured authentication, expired certificates, web injection loopholes, attacks on payment gateways (third-party gateway services), and many others [3]. A recent report by the Federal Trade Commission (FTC) has suggested that financial fraud on a global scale has exponentially risen from 2018 to 2023. Consumers reported losing more than $5.8 billion to fraud in 2021, which was up more than 70% from the year before, according to a newly released FTC report [4]. Thus, it becomes highly imperative to study the nature of these CC frauds conducted by malicious attackers.

The surge in CC frauds has pushed researchers globally to look at possible solutions that can thwart the attack vectors and secure the boundaries of the CC financial system (network, software, and hardware) [5]. The fraud incidents have forced innovative solutions to secure the network perimeters and present privacy and integrity [6,7], authorization and identity-based [8] and non-repudiation-based solutions [9]. Owing to the complex nature of attacks and zero-day possibilities, it is difficult to build an end-to-end CC fraud detection scheme that addresses financial ecosystems' security, privacy, and accuracy requirements. Traditional crypto-primitives (secured 3D passwords and multi-layer gateway encryption) require overhead due to proxy channels and the requirement of identity control and multi-attribute signatures. It significantly hampers the Quality-of-Service (QoS) for end CC applications [10]. For CC fraud detection, security schemes are proposed of specific nature, and thus, such schemes are not generic and are custom-built to support end applications. Thus, it is crucial to analyze and study the CC attack patterns, the effect, and the disclosure strategy to conceptualize a generic security scheme that can cater to large attack sets.

Lately, artificial intelligence (AI)-based models have been used as a potential tool in CC financial fraud (FF) patterns [11,12]. The CC-FF detection algorithm works on URL detection, phishing detection, behavior-based authentication, and others. Machine learning (ML) and deep learning (DL) models are proposed to increase the attack detection accuracy in financial payment ecosystems [13]. Thus, the AI scope in CC-FF detection has solved challenges of security vulnerabilities of Android/IoS mobile OS, permission attacks, and web-URL attacks [14]. However, owing to the massive amount of available data, and real-time analysis, ML models are not generally considered effective for CC-FF detection. Thus, DL techniques are mostly employed to improve the accuracy and precision of CC-FF detection [15]. A fraudulent transaction closely resembles a genuine transaction, which can be detected by minute-level (fine-grained) pattern analysis. In such cases, the behavioral pattern technique aids in determining the transaction's flow and order. So, the anomaly is quickly identified based on the behavioral trend observed from previous attacks dictionaries.

For small datasets, standard ML techniques employ decision trees, random forests, and support vector machines. For large data, mostly recurrent neural networks (RNNs) and long short-term memory (LSTM) models are considered as they can process data sequences, which is a common feature in financial transactions. These models maintain a memory of past events (unusual patterns in CC transaction histories, spending behavior, unusual withdrawals, deposits, and small transactions to multiple accounts) [16]. Such events are considered anomalous events. Other suitable models include deep belief networks, autoencoders, and gated recurrent unit models. These models have shown promising models, but the performance varies significantly owing to the application requirements and dataset characteristics. In most average cases, RNNs and LSTM perform well [14]. Thus, in the proposed scheme, we have worked on the CC-FF detection based on the decoded–encoded input using the LSTM model.

With LSTMs, the accuracy of the prediction model improves, but it is equally important to understand the factors the model uses to make its predictions. Thus, including explainable AI (XAI) with LSTM is a preferred choice which would help the users understand significant data features the LSTM model uses to predict fraudulent CC transactions [17].

The Explainable Artificial Intelligence (XAI) refers to techniques and approaches in machine learning that enable humans to understand and interpret the reasoning behind the decisions made by AI models. XAI aims to improve AI systems' transparency, accountability, and trustworthiness. It [18] is also used in other domains such as healthcare, education, marketing, and agriculture. For instance, the authors of [19] utilize XAI in an autonomous vehicle where they efficiently interpret the black box AI models to enhance the accuracy scores and make autonomous driving safe and reliable. Furthermore, in [20], the authors use the essential properties of XAI for fall detection using wearable devices. They applied the Local Interpretable Model-Agnostic Explanations (LIME) model to obtain important features from the fall detection dataset and provide better interpretability of the applied AI models. The integrated model of XAI and LSTM is termed an explainable LSTM (X-LSTM) model. It reduces the bias in the data and model, which is essential for validating the obtained results. This approach applies XAI before the LSTM in the X-LSTM model. X-LSTM helps to improve the accuracy of simple LSTM models via the identification of gaps sequences in the data or model that need to be addressed. It can handle regulatory requirements, which improves the visibility and transparency of CC financial transactions.

Once the prediction results are obtained from the X-LSTM model, there is a requirement for transaction traceability and verification. Thus, the integration of blockchain (BC) and smart contracts (SC) makes the CC-FF detection ecosystem more transparent, auditable, and visible for interpretation to all financial stakeholders (banks, users, CC application, and gateway servers) [21–23]. The obtained model results are stored with action sets in SC, and such contracts are published over decentralized offline ledgers, such as interplanetary file systems (IPFS) or swarm networks. The use of IPFS-assisted SC would improve the scalability factor of public BC networks, as only the metadata of the transactions are stored on the public BC ledger. The actual data can be fetched through the reference hash from the BC ledger and mapped with the IPFS content key to obtain the executed SC. This makes the CC-FF scheme distributed among networked users and adds a high degree of trust, auditability, and compliance in the ecosystem.

After going through several studies in the literature mentioned in Section 2, we analyzed that recent approaches in CC-FF detection mainly include rule-based systems, statistical models, and sequence-based models (RNNs, LSTMs), which often need more interpretability. Most approaches do not cater to the requirements of new FF patterns and are tightly coupled to the end application only. Thus, from the novelty perspective, we integrate XAI with the LSTM model (our proposed X-LSTM approach). *RaKShA* addresses the issue of transparency and interpretability of CC-FF detection and further strengthens the power and capability of LSTM models. Secondly, our scheme is innovative as we propose the storage of X-LSTM output in SC, which provides financial compliance and addresses auditability concerns in financial systems. Via BC, the proposed scheme ensures that all results are verifiable, traceable, and tamper-proof, which is crucial in the financial industry. Finally, to address the scalability concerns of the public BC networks, we have introduced the storage of SC and associated data in IPFS and the content hash to be stored as metadata (transaction) in public BC. This significantly reduces a transaction's size, allowing more transactions to be packaged in a single block. This makes our scheme resilient, adaptable to real-time financial systems, and generic in CC-FF detection scenarios. Furthermore, the research contributions of the article are as follows.

- A system flow model of the proposed scheme is presented for the CC-FF datasets considering the X-LSTM model and the storage of prediction results finalized via SC on the BC network.
- Based on the system flow, a problem formulation is presented, and we present a layered overview of our proposed scheme and its associated layers.
- In the scheme, at the AI layer, we design a boosted XAI function on the CC dataset post the preprocessing phase, and the output is fed to the LSTM model, which improves the accuracy. The LSTM output is fed to SC to detect fraudulent CC transactions.

- The performance analysis is completed on testing and validation accuracy, RMSProp optimizer, and XAI variable importance plot. The transaction costs, IPFS bandwidth, and SC contract are evaluated for BC simulation.

The rest of the paper is structured as follows. Section 2 discusses the existing state-of-the-art (SOTA) approaches. Section 3 presents the proposed scheme's system model and problem formulation. Section 4 presents the proposed scheme and details the data preprocessing, the sliding window formulation, the X-LSTM model, and the proposed SC design. Section 5 presents the performance evaluation of the proposed scheme. Finally, Section 6 concludes the article with the future scope of the work.

2. State-of-the-Art

The section discusses the potential findings by researchers for FF detection via AI models and BC as a trusted component to design auditable financial systems. Table 1 presents a comparative analysis of our scheme against SOTA approaches. For example, Ketepalli et al. [24] proposed the LSTM autoencoder, vanilla autoencoder, and random forest autoencoder techniques for CC datasets. The results show high accuracy for LSTM and random forest autoencoders over vanilla autoencoders. The authors in [25] explored the potential of DL models and presented a convolutional LSTM model for CC-FF detection. An accuracy of 94.56% is reported in their work. In some works, probability and statistical inferences are presented. For example, Tingfei et al. [26] proposed an oversampling-based method for CC-FF detection using the LSTM approach. Cao et al. [7] described a unique method for identifying frauds that combines two learning modules with DL attention mechanisms. Fang et al. suggested deep neural networks (DNN) mechanisms for Internet and web frauds. The scheme utilized the synthetic minority oversampling approach to deal with data imbalances [27]. Chen et al. [28] proposed using a deep CNN network for fraud classification.

Similarly, trust and provenance-based solutions are proposed via BC integration in financial systems. Balagolla et al. [29] proposed a BC-based CC storage scheme to make the financial stakeholders operate autonomously. Additionally, the authors proposed an AI model with scaling mechanisms to improve the scalability issues of the BC. Musbaudeen and Lisa [30] proposed a BC-based accounting scheme to automate daily accounting tasks and simplify audit features for a banking system. The authors in [31] researched the imbalanced classification problem. Additionally, the authors presented limitations of CC datasets (labeled data points), which makes it difficult to summarize model findings. Thus, low-cost models are preferred. Tingfei et al. [26] proposed an oversampling strategy based on variational automated coding (VAE) and DL. This technique was only effective in controlled environments. The study results showed that the unbalanced classification problem could be solved successfully using the VAE-based oversampling method. To deal with unbalanced data, Fang et al. [27] suggested synthetic minority oversampling methods.

Zheng et al. [32] presented boosting mechanisms in CC systems. The authors used AdaBoost ML during the training process. The model incorrectly classified many different symbols. Thus, improved TrAdaBoost is presented that updates the weights of incorrectly classified data. Cao et al. [7] presented a two-level attention model of data representation for fraud detection. The sample-level attention learns in a central manner where the significant information of the misclassified samples goes through a feature-level attention phase, which improves the data representation. The dependency between model fairness and scalability is not discussed.

Table 1. Comparative analysis of proposed scheme with SOTA schemes.

Authors	Year	Objective	Algorithms	Dataset	Outcomes	Disadvantages
Proposed work	2023	Integrated XAI with LSTM to improve model interpretability	LSTM	CC fraud dataset	Improved accuracy with XAI	-
Belle et al. [33]	2023	Network-based representation learning	Representation Learning	Real-life CC dataset	Conventional performance and network measures are discussed	Security aided principles are not discussed
Ni et al. [34]	2023	Fraud detection models based on feature boosting mechanism with spiral balancing technique	Feature boosting mechanism	Two CC real-world datasets	Multifactor synchronous embedding mechanism is presented with a spiral method to improve the feature metric	Oversampling of data is not considered
Labanca et al. [35]	2022	An active learning framework and a feedback system-based model to identify fraud involving money laundering	ML with anomaly patterns	A synthetic capital market dataset	Isolation forest is best algorithm for anti-money laundering fraud detection	lack of an intuitive explanation for the anomaly score
Xiuguo et al. [14]	2022	The authors developed a model to detect fraud using textual and numerical data from the annual report	LSTM, and gated recurrent units	5130 Chinese A-share listed companies' annual reports from 2016–2020,	Accuracy could be increased by using textual data	Missing feature selection details
Esenogho et al. [36]	2022	Proposed efficient approach for fraud detection using synthetic minority oversampling technique (SMOTE) with edited nearest neighbor	SMOTE-ENN	CC dataset	90% accuracy of ensemble model	Feature selection is not properly explained
Chen et al. [37]	2022	Hierarchical multi-task learning approach	federated learning and ML	Auto loan dataset of Chinese automobile	Based on its high accuracy and F1-score, machine transfer learning approach performs better than other algorithms for predicting fraud	Feature selection is not presented

Table 1. *Cont.*

Authors	Year	Objective	Algorithms	Dataset	Outcomes	Disadvantages
Ji and Yingchao [38]	2021	Proposed the fraud detection support system using XAI	XAI over ML and DL models	Open CC datasets	DNN model scored 96.84% accuracy compared with random forest	Less emphasis on transparency and interpretability
Ileberi et al. [39]	2021	Proposed ML framework for CC frauds	logistic regression, random forest, extreme gradient boosting, and decision trees	CC fraud dataset	Extreme gradient boosting and Adaboost achieves 99% accuracy	The dataset is oversampled and overfitted
Cao et al. [7]	2021	Combines two modules of learning with DL and attention methods	DL and attention mechanisms	self-defined multiple-fraud dataset	Feature-level attention helps detection models learn more about fraud patterns	Did not discuss the interpretability of the neural network
Cui et al. [40]	2021	Created a model applying the ReMEMBeR Model to address the issue of fraud detection as pseudo-recommender	DL, anomaly detection, ensemble learning	real-world online banking transaction dataset	All applied algorithms are outperformed by the ReMEMBeR Model	Less attribute-related information
Benchaji et al. [41]	2021	Identified fraud in credit card transaction based on sequential modeling of data	LSTM, sequential learning	credit card dataset	The LSTM gives the higher at 96% accuracy compared with other models	Did not discuss the security of the non-fraud data
Balagolla et al. [29]	2021	Proposed a decentralized BC-based fraud detection model	logistic regression, SVM, and random forest	CC dataset	Random forest secured 99% accuracy with public BC	Details of BC implementations are not discussed
Chen et al. [28]	2021	Proposed a deep convolution neural network (CNN) to detect financial fraud	CNNs with mining techniques	real-time CC fraud dataset	The model offers improved detection when compared to the existing models	Feature selection is not determined
Forough et al. [42]	2020	Proposed an ensemble model based on the sequential model to detect credit card fraud.	ANN, LSTM, FFNN, GRU	European cards dataset and The Brazilian dataset	Based on precision and recall, the LSTM performs better than other models	Feature selection and data security methods are not discussed
Tingfei et al. [26]	2020	An oversampling-based VAE is suggested for the detection of credit card fraud	generative adversarial networks, principal component analysis, and VAE	CC fraud dataset	VAE reaches 0.5 times the number of positive cases in original training set	Recall rates did not have much improvement

Table 1. *Cont.*

Authors	Year	Objective	Algorithms	Dataset	Outcomes	Disadvantages
Kumar et al. [43]	2019	Random forest for CC fraud detection	random forest	CC fraud detection	Accuracy of 90% is reported	Feature selection is not discussed
Jiang et al. [6]	2018	Proposed a unique aggregation and feedback mechanism-based fraud detection technique	sliding window for a behavioral pattern of cardholder	CC self-generated dataset	An 80% accuracy is achieved	Did not consider individual time windows

Esenogho et al. [36] observed the nature of typical ML models, which entails a static mapping of the input vector to the output vector. These models are inefficient for the detection of CC frauds. To detect credit card fraud, one author proposed the neural network ensemble classifier and the SMOTE techniques to create a balanced dataset. The ensemble classifier uses the adaptive boosting (AdaBoost) algorithm and LSTM neural network as the base learner. Combining SMOTE-ENN and boosted LSTM classifier methods are efficient in detecting fraud. The research on fraud detection on a dataset of Chinese listed businesses using LSTM and GRU was presented by Xiuguo et al. [14]. A DL model with proposed encoding and decoding techniques for anomaly detection concerning time series is presented by Zhang et al. [16]. Balagolla et al. [29] proposed a methodology employing BC and machine intelligence to detect fraud before it happens. Chen et al. [37] presented research on loan fraud prediction by introducing a new method named hierarchical multi-task learning (HMTL) over a two-level fraud classification system. Chen et al. [28] proposed a deep CNN model (DCNN) for CC-FF with alert notifications, and the model presented high accuracy.

From the above literature, we analyzed that many researchers proposed their solutions concerning CC fraud detection. However, their approaches utilize obsolete feature space that cannot be considered in the current timespan. None of them have used the staggering benefits of XAI that efficiently selects the best features from the given feature space. Additionally, it is also analyzed that once the data are classified using AI algorithms, the data are not overlooked for data manipulation attacks. A broad scope is available to the attackers, where they can tamper with the classified data (from AI algorithms), i.e., from fraud to non-fraud or vice versa. Hence, the amalgamation of XAI with AI algorithms and integration of blockchain is not yet explored by the aforementioned solutions. In that view, we proposed an XAI-based LSTM model (X-LSTM) that seamlessly collected the efficient feature space and then passed it to the AI algorithm for the classification task. Furthermore, the classified data are forwarded to the IPFS-based public blockchain to tackle data manipulation attacks and preserve data integrity.

3. *RaKShA*: System Flow Model and Problem Formulation

In this section, we discussed the proposed scheme, *RaKShA* through a system flow model and presented the problem formulation. The details are shown as follows.

3.1. System Flow Model

In this subsection, we present the schematics of our scheme *RaKShA*, which presents a classification model to identify fraud patterns in the financial ecosystems. Figure 1 presents the proposed system flow diagram. In the scheme, we consider the entity E_U, which denotes the user entity (whose financial data are under scrutiny). In the scheme, we assume there are n E_U, denoted as $\{U_1, U_2, \ldots, U_n\}$.

For any U_n, we consider CC details, denoted by $F(U_n) = \{U_{LB}, U_{BA}, U_{PA}, U_{RS}\}$, where U_{LB} denotes the balance limit of CC, U_{BA} denotes the pending bill amount of the monthly CC billing cycle, U_{PA} denotes the payment amount E_U is liable to make, and U_{RS} denotes the repayment status (Step 1). For the experiment, we select the credit card dataset (U_{CC}) (Step 2). The data are collected into comma-separated values (CSVs), and preprocessing techniques are applied to the collected CSV. The preprocessed data are sent through a sliding window for U_n, denoted as $W(U_n)$ (Step 3). Based on $W(U_n)$, the data are sent to the XAI for feature importance which is denoted as $X(W(U_n))$ (Steps 4, 5). The XAI output is then passed to the LSTM model to classify the fraud patterns of U_n (Step 6). Based on the X-lSTM model output, E_{U_n} executes an SC to notify the user of the genuineness and the safety of investment on U_n (Step 7). The classification details are also stored on local IPFS, where any public user can fetch U_n data based on the IPFS content key (Step 8). Finally, the transaction meta-information obtained from IPFS is stored on public BC (Step 9).

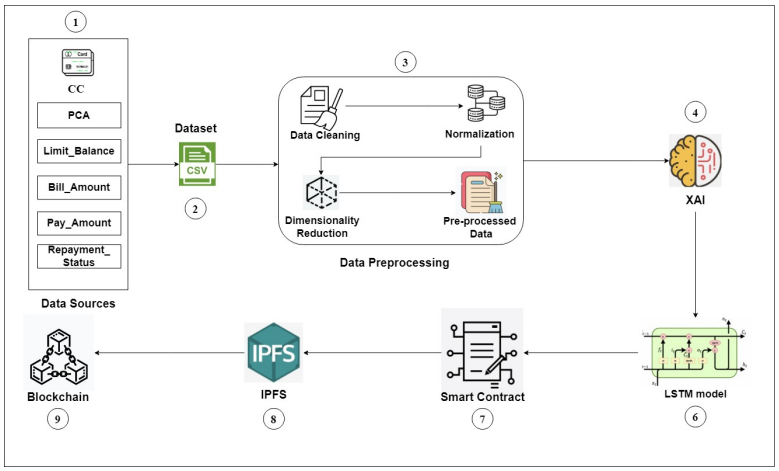


Figure 1. *RaKShA*: The proposed system flow model.

3.2. Problem Formulation

As discussed in the above Section 3.1, the AI-based *RaKShA* scheme is proposed for any n^{th} E_U. The financial resource is collected from the user entity. For simplicity, we consider that each user has a single CC for which fraud detections are classified. Thus, any user $\{U_1, U_2, \ldots, U_n \in E_U\}$ has an associated CC denoted as $\{C_1, C_2, \ldots, C_n\}$. The mapping function $M : E_U \rightarrow C$ is denoted as follows.

$$U_i \longrightarrow C_i \tag{1}$$

Similarly, the scheme can be generalized for many to one-mapping, where any user might have multiple CCs for which fraud detection is applied. In such cases, we consider a user identity U_{id} to be mapped to different CCs offered by respective banks. The mapping $M_2 : U_{id} \rightarrow C_k \rightarrow B_k$ is completed, where any n^{th} E_U is mapped to a subset $C_k \subset C$, which is further mapped to associated banking information $B_k \subset B$, where C denotes the overall CC set, and B denotes the associated banks who have presented these CCs to U_i.

$$U_i \longrightarrow C_k \tag{2}$$

In the model, we consider two types of transactions: normal (genuine) transactions and fraudulent (fraud) transactions. Any U_i uses its CC at multiple shopping venues (online or point-of-sale transactions), money transfers, cash withdrawals, and others. We consider that fake CC transactions are generated by an adversarial user U_r who can exploit the operation of the CC transaction.

$$U_r \notin \{U_1, U_2, \ldots, U_n\} \tag{3}$$

Here, a function F represents a U_r attack on the normal transaction system, which produces a fake transaction T_f in the CC network.

$$F = \left(U_r \xrightarrow{\text{attack}} \left(U_i \xrightarrow[\text{with}]{\text{Transaction}} C_i\right)\right) \tag{4}$$

The goal of the proposed scheme is to detect this malicious T_f from normal transaction sets $T_r = \{T_1, T_2, \ldots, T_l\}$, where $T_l \subset T$, which is proposed by n genuine users. The main goal is satisfied when every transaction is in normal behavior similarly to T_r. In

Equation (5), we present the sum of the maximum count of the normal behavior of the CC transaction.

$$\mathbb{O} = \left(\sum_{i=0}^{l} \text{secure}(T_r) \right) \tag{5}$$

The models work on the detection of T_f and differentiate its anomalous behavior from T_r. The detection mechanism is presented in Equation (6) as follows.

$$\mathbb{Q} = \left(\sum_{i=0}^{l} \text{detect}(T_f) \right) \tag{6}$$

We design an XAI model for the CC dataset, which finds the important features for the classification of T_f. The important features $Imp(F_s)$ are passed as inputs to the LSTM model, which generates the classification output. The goal is to maximize accuracy $A(O)$, which is fed to SC to be stored at the BC layer. In general, the problem formulation P_f aims at maximizing the A_O, $Imp(F_s)$. Secondly, the LSTM model should minimize the training loss T_loss and maximize the validation accuracy $A(Val)$. Mathematically, the conditions for P_f are summarized as follows.

1. C_0*: Maximize* $Imp(F_s)$: To improve the accuracy of the LSTM model, we consider that our XAI approach would maximize $Imp(F_s)$, which would aid in the maximization of $A(O)$.
2. C_1*: Maximize* $A(O)$: The LSTM model post-XAI (X-LSTM) would focus on maximizing $A(O)$, such that T_{loss} is minimized.
3. C_2*: Minimize* T_{loss}: This would help in improving the validation accuracy $A(Val)$.
4. C_3*: Maximize* $A(Val)$: The final condition is to reduce false positives, which would improve $A(Val)$.

The entire problem P_f is then represented as a maximization problem.

$$P_f = max(C_0, C_1, -C_2, C_3) \tag{7}$$

subject to operational constraints as follows.

$$\begin{aligned} &OC_1 : T \leq T_{max} \\ &OC_2 : T_f \leq T_r \\ &OC_3 : C(T_f) = \{0, 1\} \\ &OC_4 : T_{loss} \leq T_{thresh} \\ &OC_5 : G(C_e) \leq G(Acc) \\ &OC_6 : E(C) \leq \Delta(maxT) \end{aligned} \tag{8}$$

OC_1 denotes that the LSTM model should respond with output in a finite bounded period, denoted by T, which should not exceed a timeout T_{max}. OC_2 denotes that the scheme is rendered fair when the number of fake transactions exceeds genuine transactions. The scheme would have less accuracy when T_f would exceed genuine transactions in the ecosystem. C_3 talks about a deterministic property of T_f classification $C(T_f)$, that it would always output $\{0, 1\}$, which is a Boolean identifier to classify the transaction as genuine (1) or fake (0). Any other state is not acceptable. OC_4 indicates that T_{loss} should not exceed a threshold training loss, which is decided in real time based on previous inputs and outputs to the model. OC_5 indicates conditions for SC execution, which signifies that SC should be only executed (C_e) when the account wallet has sufficient funds (in terms of gas limit), which are denoted by $G(C_e)$. Thus, it should be less than the total fund in the wallet ($G(Acc)$). OC_6 denotes that the time to add the update of SC execution to IPFS

and block mining in BC should again be finite and should not exceed a maximum timeout $\Delta(maxT)$ set by the public BC network.

4. *RaKShA*: The Proposed Scheme

This section presents the working of the proposed system, which is presented as a layered model into three sublayers: the data layer, the AI layer, and the BC layer. Figure 2 presents the systematic overview of the architecture. The details of the layers are as follows.

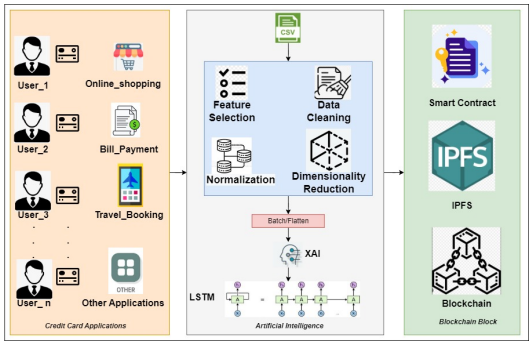


Figure 2. Architectural view of the proposed scheme (*RaKShA*).

4.1. Data Layer

At the data layer, we consider E_U, which have mapped CCs (one-to-one or one-to-many mapping), and these users conduct transactions from their CCs at multiple venues. We consider that the CC firms can track conditions of lost, inactive, or stolen CC, and thus, any transaction made after a complaint lodged by E_U should be marked as T_f. The challenge lies in identifying fraud transaction patterns P_f, which look identical to genuine patterns P_r. The real-time data at this layer are collected and compiled to form the CC transaction dataset, which is added as a CSV file to the AI layer.

We consider that E_U uses different applications (portals) to carry out transactions (both normal and abnormal). We consider transaction instances $\{T_1, T_2, \ldots, T_l\}$ for n users with q CC, with the trivial condition $q \geq n$. We consider any user makes w transactions in the billing cycle from CCs, which are mapped to U_{id}, and the overall amount A is computed at the billing cycle. Thus, the mapping is denoted as follows.

$$\begin{aligned} U_{id} &\xrightarrow{\text{performs}} T_l \\ T_l &\xrightarrow{\text{maps}} q \\ U_{id} &\xrightarrow{\text{bill}} A \end{aligned} \tag{9}$$

Specifically, any U_{id} contains the following information.

$$U_{id} = \{CC_{num}, B_{id}, Txn, TA, CC_{lim}, OD_{lim}\} \tag{10}$$

where CC_{num} denotes the CC number, B_{id} denotes the bank identifier which has issued the CC, Txn denotes the total transactions carried out on CC_{num} in the billing cycle, TA denotes the total amount (debited) in the billing cycle, CC_{lim} denotes the spending limit (different for online transactions and offline transactions), and OD_{lim} denotes the overdraft limit set for U_{id} on CC_{num}.

In the case of genuine transactions T_r, the values of Txn over the billing period are not sufficiently high, which indicates that CC is used frequently. In addition, the location of the CC swipe (online based on gateway tracking and offline based on geolocation) should not be highly distributed (different locations indicate anomaly). Furthermore, TA should not normally exceed the CC_{lim}, and OD_{lim} should not reach the maximum OD set for CC_{num}.

Fake transactions T_f have a high probability of violating any of these conditions, which the AI model captures based on credit histories.

For all users, we collect the transaction details and store them in a CSV file, denoted as CV_d, and T_i represents the transaction data.

$$\forall T_i \in CV_d \tag{11}$$

In the CC fraud detection dataset, there are 31 attributes, where attributes $V_1 - V_{28}$ denote features that resulted from transformation via principal component analysis (PCA) and are numerical values. T denotes the elapsed time between the current transaction and the first transaction, and a class C attribute signifies whether a transaction is T_f or T_r.

$$A = \{\{V_1, V_{28}\}, T, sC\} \tag{12}$$

The prepared CSV file is then sent to the AI layer for analysis.

4.2. AI Layer

At this layer, the CSV file is sent to the XAI module, whose main aim is to reduce the dataset dimensionality and maximize the accuracy of finding the important features $Imp(F_s)$ of the dataset, which in turn would maximize $A(O)$, predicting the required output. The dataset dimension is modeled as $r^{P \times Q}$, where P denotes rows, and Q denotes columns. Thus, the goal is to select $Imp(F_s)$ over Q so that only important features are presented to the LSTM model and it achieves high accuracy.

4.2.1. Data Preprocessing

This sublayer considers the data preprocessing techniques, including data cleaning, normalization, and dimensionality reduction. In data cleaning, we correct or delete inaccurate, corrupted, improperly structured, redundant, or incomplete data from the dataset D obtained from the CSV file. Not applying data preprocessing techniques leads to inefficient accuracies and high loss while classifying CC frauds. For instance, not applying data normalization leads to a range problem, where the particular value of a column is higher or lower than the other value in the same column. Furthermore, missing values are not formally accepted by the AI models; hence, they must be filled with either 0 or central tendency values, i.e., mean value. Similarly, it is essential to find the best features from the dataset; otherwise, the AI model will not be trained efficiently and not improvise the accuracy and loss parameters. Toward this aim, we utilize the standard preprocessing techniques on the CC fraud data. In place of NaN values, 0 is inserted, and then, the label encoding technique is used to transform string values in columns into a numeric representation. Finally, we consider C to denote the different target classes in D, and a group transformation is applied, represented by ω and δ, which divides the data into different timestamps.

$$\delta_l = \omega_l(\delta_{\text{instances} \times \text{features}}) \quad \forall l \epsilon C \tag{13}$$

The NaN data are denoted as $[\varnothing]$. We first use the *isnull()* function to find the overview of NULL values in the dataset, and the operation is denoted by N_s, for NaN Data$[\varnothing]$.

$$N_s = isnull(CV_d) \tag{14}$$

From the data, the NULL values are then dropped using the *drop()* function, which is denoted as D_c. The function is denoted as R_i for a particular row.

$$R_i \longrightarrow D_c \tag{15}$$

Next, all NaN values are updated in the column for which we compute the mean using the $fillna()$ function. The model replaces categorical data, whereas the mean and

median are used to replace numerical data. The column of data is denoted as c_i. After filling in the empty data, the cleaned data are denoted as C_D.

$$\begin{aligned} A &= \frac{1}{n}\sum_{i=1}^{n} c_i \\ C_D &\longleftarrow c_i \end{aligned} \tag{16}$$

Each target class has a string data type and needs to be transformed using the one-hot encoded vector. Consider y as the original target class column, with Γ as unique classes. y has the shape $(k, 1)$ and the data type string. y is subjected to a single hot encoding transformation.

$$y_{k,\Gamma} = \begin{bmatrix} 1 \\ 0 \\ . \\ 0 \\ 0 \end{bmatrix} \cdots \begin{bmatrix} 0 \\ 1 \\ . \\ 0 \\ 0 \end{bmatrix} \cdots \begin{bmatrix} 0 \\ 0 \\ . \\ 0 \\ 1 \end{bmatrix}$$

Normalization is typically required with attributes of different scales. Otherwise, the effectiveness of a significant and equally significant attribute (on a smaller scale) could be diminished as other qualities have values on a bigger scale. Statistics refers to the process of reducing the size of the dataset so that the normalized data fall between 0 and 1.
where $(\forall C_D \in X_{\text{normalized}})$

$$X_{\text{normalized}} = \frac{(X - X_{\text{minimum}})}{(X_{\text{maximum}} - X_{\text{minimum}})} \tag{17}$$

The technique of minimizing the number of random variables or attributes taken into account is known as dimensionality reduction. In many real-world applications, high-dimensionality data reduction is crucial as a stage in data preprocessing. In data mining applications, high-dimensionality reduction has become one of the crucial problems. It is denoted as D_r.

$$\begin{aligned} D_r &= \{x_i, y_i\} \\ x &\in \mathbb{IR}^y \end{aligned} \tag{18}$$

4.2.2. XAI Module

In the work, we propose an XAI module that uses a boosting approach, combining several weak classifiers to form a strong classifier. The $r^{P\times Q}$ data function contains 284,808 rows and 31 columns. The XAI function (Ψ) determines the highest priority on Q.

In this work, we used the feature importance technique of XAI to detect any CC fraud. Using XAI algorithms obtains better prediction over the other results. The XAI gives the highest priority feature of the dataset columns *(Q)*. XAI function (Ψ) is applied to the dataset, which is denoted as follows.

$$\zeta = \Psi\{\forall r^{284808\times 31}\} \tag{19}$$

After applying the XAI function on the dataset, we obtain the important feature, and dimensionality becomes reduced. The new function of XAI is denoted as ζ, and $\mathbb{R}$ is a new dimension of the dataset.

$$\zeta \longleftarrow (\mathbb{R}^{284808\times 20}) \tag{20}$$

4.2.3. LSTM Module Post-XAI: The X-LSTM Approach

Once XAI selects $Imp(F_s)$, we shift toward sending the features as inputs to the LSTM model. We use a technique of flattening to feed information from multiple vectors into the classification model as a 1D array. The preprocessed data are called CV_p. The parameter is

immediately updated after computing the gradient for one training sample in stochastic gradient descent. It is a mini-batch with a batch size of 1. We repeat the two procedures—flattening and mini-batch selection—in all the training samples. Mini-batch is represented as $\mathbb{B}$, which is presented as follows.

$$\hat{y} = \theta^T \mathbb{B} + r \tag{21}$$

The loss function is used to evaluate the proposed algorithm, which predicts the highlighted dataset. It is determined as μ. All preprocessed feature data move on the LSTM block. A hidden LSTM unit has several gates, each completing a certain function. The tanh and sigmoid activation functions manage the gates input and output. Within the LSTM cell, the activation functions sigmoid and tanh are utilized for the input vector x. Figure 3 shows the details of the LSTM cell for processing our XAI output.

$$\mu(\theta, B) = \sum_{i=1}^{n} -y \log(\hat{y}) \tag{22}$$

subject to,

$$\theta_{k+1} = \theta_k - \alpha \nabla J_j(\theta) \tag{23}$$

where θ denotes the weights, and B is the bias value. The actual class is y, and $\hat{y}$ is the predicted class. α is the learning rate and ∇ denotes the partial derivative.

$$CV_p \longrightarrow B_s \tag{24}$$

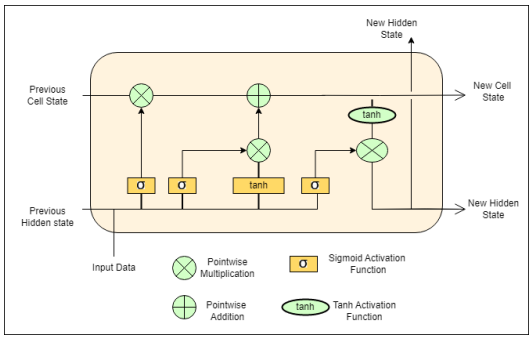


Figure 3. LSTM architecture.

The RMSProp optimizer function is considered, and the loss function is MSE. The algorithm addresses the C_1, C_2, and C_3 conditions of P_f under the operational constraints OC_1, as our proposed model training preprocesses the data. Thus, the training response is in a finite bounded period (does not exceed T_{max}). The algorithm effectively classifies T_f from T_r transactions, and OC_2 is satisfied. A deterministic output $A_y = \{0, 1\}$ is obtained from the LSTM block, which satisfies our OC_3 condition. T_{loss} is under an experimentally observed T_{thresh}, which is obtained from successive process runs on different CC datasets. This satisfies the OC_4 condition.

4.3. BC Layer

At this layer, the classification output A_y obtained from the CC fraud detection dataset with the transaction details is stored in an SC. We consider the transaction details X, where every row corresponds to a user transaction, and the column corresponds to features. The LSTM model outputs A_y, based on the classification function $F(X) = Y$. In the SC<, we consider a function *storeDetails*(X, Y), which takes inputs X and Y from LSTM output. The SC is executed, and the details are published on IPFS, from which we generate a javascript object notation (JSON) file, which is denoted as $JS(X, Y)$. Next, we use the IPFS API to

publish $JS(X, Y)$ on the IPFS network, which generates the content key $CK(IPFS)$ and the hash of $CK(IPFS)$, which is denoted as H_{CK}. H_{CK} is stored as a transaction on a public BC network, using the *storeHash(H_{CK})* function.

Algorithm 1 presents the details of the SC. In the case of fraud transaction detection, all nodes of the BC are notified of the account from which it is detected.

Algorithm 1 SC to store fraud transactions in BC network

Input: $\{A_y\}$ LSTM output
Output: A flag F to denote fraud transaction is detected

```
procedure STORE FRAUD TRANSACTIONS
    Eth.acc ← Connect Ethereum Network
    A_U ← Call_ProcedureFraud Notification
    address owner
    Event E FraudDetect (M_s, uint256, T)
    Call constructor()
    owner = M. sender
    U ← Call_Notify_Fraud_Detected (M_s)
    Require(M.sender == owner)
    emit FraudDetected(M, block.T)
    if (W(owner) > min.bal) then
        Deploy contract
        Notify F ← 1
        Notify 'Fake Transaction detected'
        Generate JSON file JS(X, Y)
        Publish contact on IPFS and generate CK(IPFS)
        H_CK ← SHA-256(CK(IPFS))
        C ← storeHash(H_CK)
    else
        Notify F ← 0
        Notify 'Transaction is genuine'
        Generate JSON file JS(X, Y)
        Publish contact on IPFS and generate CK(IPFS)
        H_CK ← SHA-256(CK(IPFS))
        C ← storeHash(H_CK)
    end if
end procedure
```

Initially, we connect to the Ethereum network using Web3.js and send a transaction to SC [44]. Next, an account is created to which the function $FraudNotification()$ has access and message M is added with timestamp T in a block in case function $Notify\_Fraud\_Detected$ returns *true*. In such a case, the required gas fee $G(C_e)$ should be in the owner's address to deploy the contract. This satisfies the OC_5 condition of P_f. Next, the executed contract produces bytecodes, and input data are supplied to the contract. The contract data are also published on the IPFS network, and CK and H_{CK} are generated, which links the published contract with the BC network. Once complete, the function $close()$ closes a connection to the Ethereum network and frees up the EVM memory.

For the experiment, we perform an SC function such as the $getFraudStatus()$ function, which returns a boolean indicating the fraud status of the *cardholder* associated with the Ethereum address that is calling the function. It retrieves the cardholder struct associated with the caller's address through the cardholders mapping and returns the *isFraud* boolean value of the *cardholder* struct. A true value indicates that the cardholder has engaged in fraudulent activities, while a false value indicates that the cardholder has not. $checkFraudTransactionStatus()$ is a public view function in the FraudDetection contract that takes an Ethereum address as input, retrieves the associated *cardholder* struct, and returns a boolean indicating if the cardholder has engaged in fraudulent activities. A

true value means that the cardholder has engaged in fraud, while a $false$ value means the opposite. $getTransactionsAmounts()$ and $getTransactionsLocations()$ are public view functions defined in the FraudDetection contract that retrieve and return the transaction amounts and locations, respectively, of the cardholder that is making the function call. Both functions access the $allTransactionAmounts$ and $allTransactionLocations$ arrays stored in the cardholder struct that is associated with the cardholder's Ethereum address.

5. *RaKShA*: Performance Evaluation

In this section, we discuss the performance evaluation of the proposed scheme. First, we discuss the tools and setup and present the CC datasets (dataset 1 and dataset 2) used for simulation. Then, we present the simulation parameters for the LSTM model and the BC setup, which are followed by X-LSTM performance analysis, SC design, and performance analysis of BC metrics. Finally, the details are presented as follows.

5.1. Experimental Tools and Setup

For the experiment, we used the Google Collab platform with Python and imported the required set of libraries: Numpy for linear algebra, Fourier transforms and matrices, Pandas for ML-related tasks, and Matplotlib for visualizing the data. SC is developed in Solidity programming language on Remix IDE for BC simulation. LSTM with parameters such as epochs, batch size, and loss function is defined. We compared different optimizers for the model's accuracy [45].

5.2. Dataset

Two open-source CC datasets are analyzed for fraud transaction detection [46]. The dataset contains transactions from different CC from September 2013 from European CC users. The dataset is first made balanced, and the explicit features are hidden via PCA, and $\{V_1, V_2, \ldots, V_{28}\}$ features are present. Other features are time (the elapsed time between each transaction and the initial transaction) and transaction amount. This dataset has 0.17% class 1 data for the prediction of futuristic data. There are 284808 data available in the dataset. We experiment with techniques such as XAI data and without XAI data on this dataset to obtain better predictions between the techniques.

Another CC dataset is considered from the UCI repository [47], with CC applications. The attributes and names are hidden (due to privacy preservation and to assure user confidentiality). The dataset contains continuous, nominal, and large values, with 690 instances and 15 attributes. This dataset has attributes that help predict the class labels. There are 22.1% class 1 and 78% Class 0 data for prediction. Here, we also analyze different vectors to understand the behavior of the data. Figure 4 is the visualization of the vector performance concerning the amount. Each point in the scatter plot represents a transaction in the dataset. The amount variable represents the amount of money involved in the transaction, while each V variable is a transformed feature that the model uses to detect fraud. The X-axis represents the values of the V variable, and the Y-axis represents the transaction amount. By plotting the amount against each V variable, we can see if there is any relationship between these variables and the transaction amount. For example, if there is a strong positive correlation between the amount and a particular V variable, transactions with higher values might be more likely to involve larger amounts of money. Conversely, if there is a negative correlation between the amount and a V variable, then transactions with lower values of that variable might be more likely to involve larger amounts of money.

5.3. Simulation Parameters

For predicting the output, parameter selection plays an important role. In work, we have considered two epoch values: 50 and 500 epochs. Furthermore, the batch size is a hyperparameter that defines the number of samples to work through before updating the internal model parameters. When training neural networks, batch size regulates how

accurately the error gradient is estimated. The details of the hyperparameters are presented in Table 2. On similar lines, Table 3 presents the BC and SC setup parameters.

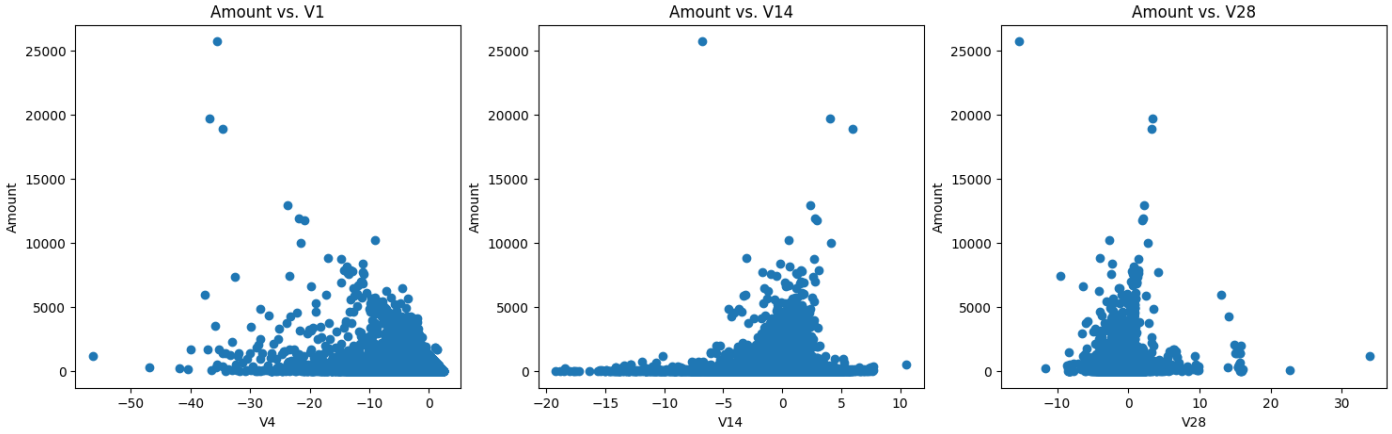


Figure 4. Vector visualization of dataset.

Table 2. Hyperparameters for the LSTM network.

Parameter	Value
Epochs	50 and 500
Batch Size	200
Optimizer	Adam, RMSprop,
Loss Function	MSE
Activation Function	Sigmoid, RELU

Table 3. BC and SC parameters.

Parameter	Value
Solidity Compiler	v0.8.17
Remix IDE	v0.28.1
Number of CC Users	100
Gas Limit	3,000,000
IPFS Key	2048 bits
Hash	256 bits
Consensus	PoW

5.4. X-LSTM Analysis

In this section, we present the simulation results of the proposed X-LSTM network based on the tuning hyperparameters. We first present the details of the XAI feature selection process, which exploits boosting mechanism to create a strong classifier from weak classifiers. We consider that XGBoost handles the relationships between data and associated distribution. Initially, we consider the Shapley addictive explanations (SHAP) model on the CC fraud dataset [46].

To validate the results obtained from the SHAP beeswarm plot, we plot the variable importance plot on the same dataset. Figure 5a presents the details. This plot shows the importance of a variable (attribute) in output classification. Thus, the plot signifies how much accuracy is affected by the exclusion of a variable. The variables are presented in decreasing order of importance. The mean on the x-axis is the mean decrease in the Gini coefficient, and thus, the higher the values of the mean decrease in the Gini score, the

higher the importance of the variable in the output prediction. From the figure, it is evident that attributes V_{14}, V_{17}, and V_{12} are more important, and attributes V_{21}, V_6, and V_2 are the least important. The plot closely synchronizes with the SHAP beewswarm plot in most instances, thus validating our cause of selection of important attributes to the LSTM model.

Figure 5b shows the details of the beeswarm SHAP plot. The results show the important features of different features. In the plot, the y-axis represents the features, and the x-axis shows the feature's importance. For example, features V_{14}, V_{17}, and V_{10} have a high SHAP value, which signifies a positive impact on CC fraud prediction. Similarly, features V_{20}, V_8, and V_{15} have a negative impact on the SHAP value and thus are not so important to the output prediction.

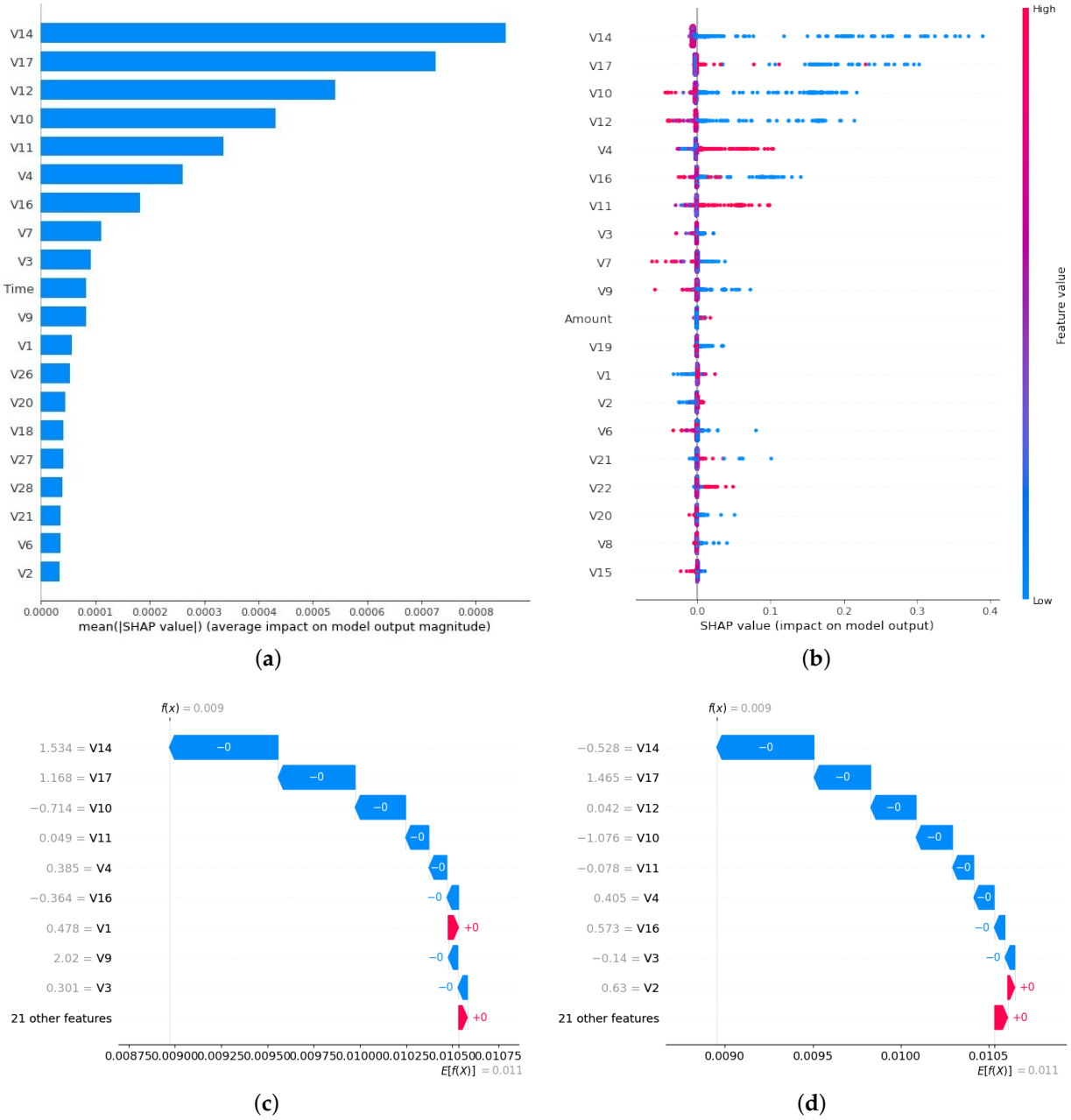


Figure 5. SHAP model performance parameter. (**a**) Variable importance plot on CC-FF dataset. (**b**) SHAP XAI model on CC-FF dataset. (**c**) Waterfall SHAP XAI model on CC-FF dataset. (**d**) SHAP XAI model on CC-FF dataset.

The SHAP model has different types of graphs for feature importance; similar to a waterfall, SHAP works by first computing the baseline value of the output, which is the expected value of the output when all input features have their average or most common values. Then, for each instance to be explained, it calculates the contribution of each feature to the difference between the actual output and the baseline output. In Figure 5c,d, each feature's contribution is represented as a vertical bar that either adds or subtracts from the baseline value. The height of the bar represents the magnitude of the contribution. Figure 6 is a Force SHAP model. It also shows the interaction effects between features [48]. These interaction effects are represented as connections between the features, and the thickness of the connection represents the strength of the interaction.

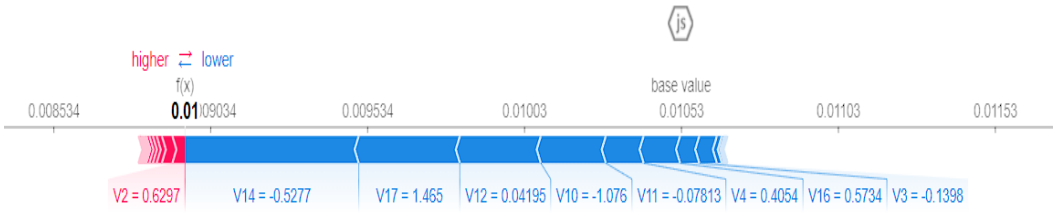


Figure 6. Force SHAP model.

In the study of feature selection, the Eli5 model is also used to present the feature importance of the data. Eli5 stands for Explain Like I'm Five, and it is used for model interpretation and explanation of machine learning models [49]. These methods help to identify the most influential features and how they affect the model's output. To examine and decipher ML classifiers, ELI5 prepares decision trees by weights for tree-based models using the Gini index [48,50]. The tabulated weights determined for each parameter are displayed in Figure 7a. The features are ranked and given weights according to their significance (the most important parameter is at the top).

LIME is a model interpretation and justification method applied to machine learning [48]. Figure 7b presents the LIME graph for CC fraud detection. In the figure, the green color represents the features are positively correlated with the local values, and red color shows the opposite correlation. The fundamental goal is to create a collection of "local surrogate models" that may be used to explain how the original model makes predictions in a specific situation. To accomplish this, LIME first creates a collection of "perturbed" instances that each have slightly different feature values from the original instance. A local surrogate model, such as a linear model or decision tree, is then trained using these perturbed instances to mimic the behavior of the original model in the immediate vicinity of the instance to be explained. It is also a model for presenting the feature importance for better prediction.

5.4.1. LSTM Performance without XAI Input Selection

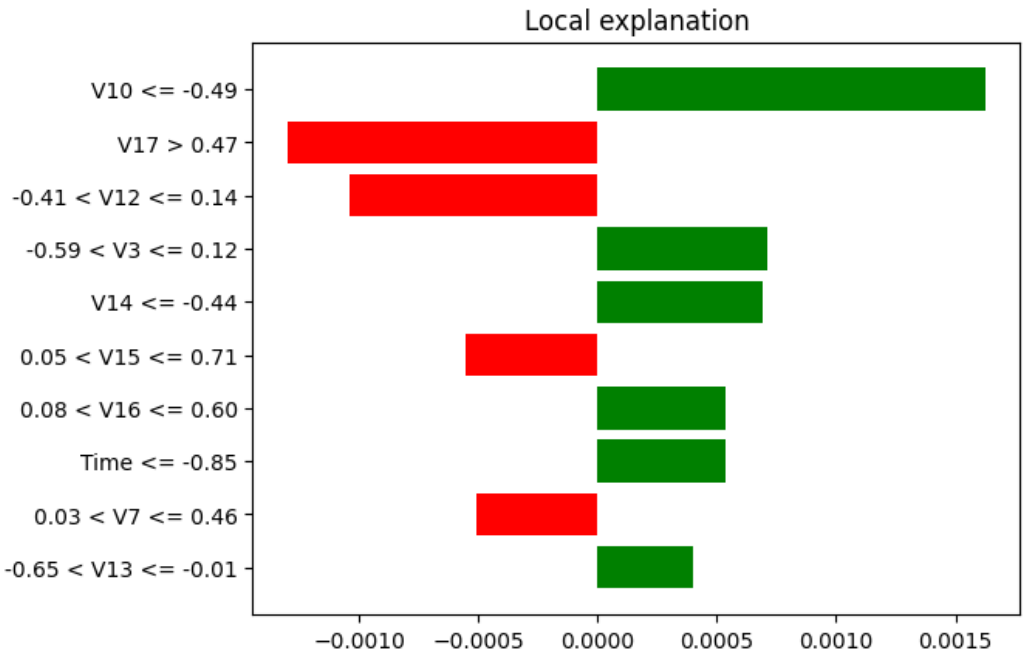
Firstly, we present the accuracy comparison by directly applying the model without considering the XAI output. For LSTM, we check the accuracy based on the epochs size, such as 50 and 500. In addition, we check the parameters such as the batch size, which is 200. We have applied the LSTM model on both datasets [46,47]. Figure 8a shows the accuracy and loss graphs for LSTM for 50 epochs. A maximum accuracy of 60% is achieved with RMSProp optimizer on the CC fraud detection dataset. For 500 epochs, Figure 8b shows the results. The model gives 85% accuracy with the RMSProp optimizer. Similarly, for the UCI dataset, Figure 8c reports an accuracy of 76% with 50 epochs and 80% accuracy for 500 epochs. Figure 8d demonstrates the results.

5.4.2. LSTM Performance with XAI Input Selection

Next, we present the performance comparison of the model with inputs considered from the XAI output. We ran the model for 50 and 500 epochs on the CC fraud detection dataset [46]. Figure 8e shows the result on the CC-FF dataset for 50 epochs, and Figure 8f shows the result for 50 epochs. Table 4 presents the comparative analysis for the CC-FF-dataset (with and without XAI) for 50 and 500 epochs, respectively. Furthermore, the proposed work is compared with [41], where they used the same dataset to detect CC fraud patterns. However, their work is carried out without applying XAI, which implies that their AI model has not analyzed essential features of CC fraud. Hence, their approach offers 96% training accuracy. Contrary, the proposed work adopts staggering properties of XAI that offer an accuracy of 99.8% without overfitting the LSTM model (as shown in Table 5). In addition, the authors also want to mention that once the AI models classify the data, it requires the data to be secure from data manipulation attacks. Nevertheless, we realize that [41] has not adopted any security feature in their proposed work. On the contrary, we have used an IPFS-based public blockchain for secure storage against data manipulation attacks. This improves the security and privacy concerns of the proposed scheme.

Weight	Feature
0.2173 ± 0.0089	V14
0.0741 ± 0.0023	V10
0.0550 ± 0.0019	V17
0.0535 ± 0.0019	V12
0.0309 ± 0.0059	V26
0.0261 ± 0.0023	V4
0.0225 ± 0.0015	V16
0.0223 ± 0.0018	V11
0.0119 ± 0.0003	V27
0.0102 ± 0.0012	V28
0.0090 ± 0.0026	V7
0.0087 ± 0.0022	V3
0.0083 ± 0.0017	V9
0.0078 ± 0.0014	Time
0.0072 ± 0.0021	Amount
0.0061 ± 0.0015	V1
0.0057 ± 0.0004	V6
0.0036 ± 0.0019	V21
0.0035 ± 0.0030	V20
0.0023 ± 0.0006	V8
0.0021 ± 0.0011	V22
0.0018 ± 0.0009	V2
0.0016 ± 0.0005	V19
0.0014 ± 0.0003	V23
0.0014 ± 0.0013	V24
0.0013 ± 0.0027	V15
0.0012 ± 0.0005	V25
0.0006 ± 0.0005	V13
0.0002 ± 0.0009	V5
0.0001 ± 0.0004	V18

(**a**)



(**b**)

Figure 7. Comparison of Eli5 and LIME XAI models. (**a**) Eli5 Model. (**b**) Lime XAI Model.

Table 4. Accuracy and loss comparison of X-LSTM.

Epochs	Accuracy	Loss
50 epochs without XAI	60%	47%
500 epochs without XAI	85%	23%
50 epochs with XAI	86.2%	3.6%
500 epochs with XAI	99.8%	1.5%

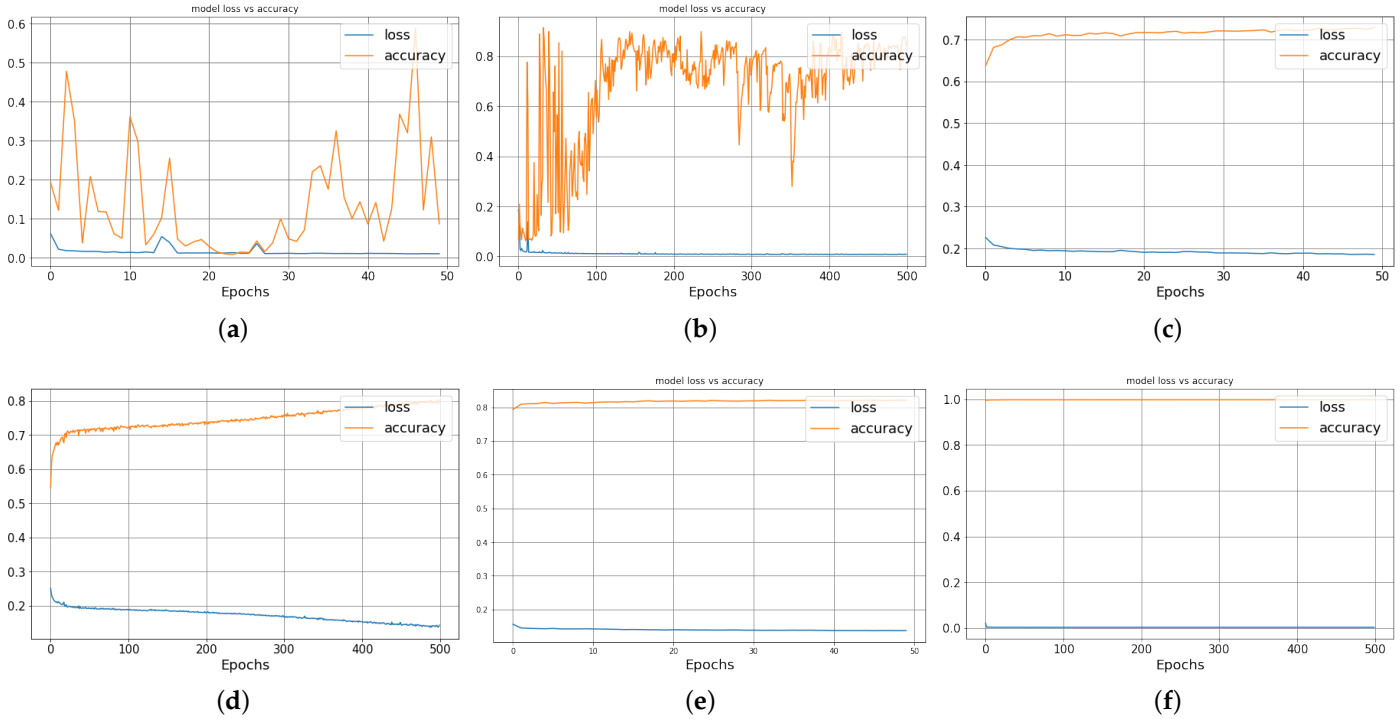


Figure 8. Performance of AI algorithm based on datasets 1 and 2. (**a**) 50 epochs LSTM RMSprop optimizer of dataset 1. (**b**) 500 epochs LSTM RMSprop optimize of dataset 1. (**c**) LSTM 50 Epochs RMSprop Optimizer of dataset 2. (**d**) LSTM 500 Epochs RMSprop Optimizer of dataset 2. (**e**) LSTM and XAI model on 50 epochs of CC-FF dataset. (**f**) LSTM and XAI model on 500 epochs of CC-FF dataset.

Table 5. Performance analysis of XAI models for 500 epochs.

XAI Models	Accuracy	Loss	Precision	Recall
SHAP	99.8%	0.0068%	99%	98%
LIME	98%	0.17%	98%	98%
ELi5	97%	0.18%	93%	96%

5.4.3. Evaluation Metrics

In AI, precision, recall, and accuracy are crucial performance metrics that enable us to quantitatively assess a model's capacity to accurately classify positive and negative instances. These parameters allow us to compare the performance of various models, identify particular areas where a model may need improvement, and are simple enough for both technical and non-technical audiences to comprehend. Overall, important factors that are heavily considered when assessing the effectiveness of binary categorization models include precision, recall, and accuracy.

- Precision ($\mathfrak{P}$): Out of all the positive predictions produced by the model, precision is the percentage of actual positive predictions. In other words, precision assesses the reliability of optimistic forecasts. A high precision number means that the model almost never predicts something that will actually happen.

$$\mathfrak{P} = \frac{\psi}{\psi + \Xi} \tag{25}$$

- Recall ($\mathfrak{R}$): Out of all the real positive instances in the dataset, recall is the percentage of true positive predictions. Recall, then, gauges the model's ability to recognize every

positive instance in the dataset. A high recall number means that the model almost never misses any successful examples.

$$\mathfrak{R} = \frac{\psi}{\psi + \xi} \tag{26}$$

- Accuracy ($\mathfrak{A}$): The percentage of accurate forecasts among all the model's predictions is known as accuracy. In other terms, accuracy assesses how well the model can categorize positive and negative instances accurately. A high accuracy rating shows that the model can accurately classify the majority of the dataset's instances.

$$\mathfrak{A} = \frac{\psi + \varsigma}{\psi + \varsigma + \Xi + \xi} \tag{27}$$

where true positive, true negative, false positive, and false negative are represented as $\psi, \varsigma, \Xi, and\ \xi$.

A binary classification model's effectiveness is graphically depicted by the Receiver Operating Characteristic (ROC) curve. At various categorization criteria, it plots the true positive rate (TPR) vs. the false positive rate (FPR). The true positive rate (TPR), often called sensitivity or recall, is the percentage of true positives (positive examples that were classified correctly) among all actual positive instances. The FPR, on the other hand, is the ratio of false positives (negative cases that were wrongly categorized as positive) to all true negative instances. The ROC curve depicts the trade-off between these two rates at different categorization thresholds. The area under the ROC curve (AUC) is a metric that quantifies the overall performance of the model across all possible classification thresholds. The AUC ranges from 0 to 1, where a perfect classifier has an AUC of 1, while a random classifier has an AUC of 0.5. Generally, a higher AUC indicates a better performance of the model. Our model achieves the 0.97 roc_auc shown in Figure 9.

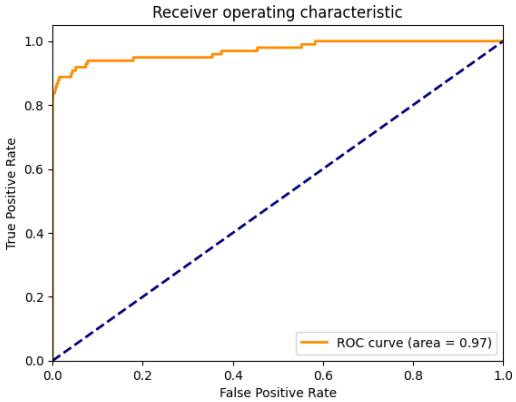


Figure 9. ROC-AUC Graph.

5.5. Smart Contract Design

In the proposed scheme, the LSTM classification output is published via SC, which helps any public user to find whether the new transaction is fake or real. In the SC, we have considered transaction detail, amount, the sender and receiver address, the location of the transaction, and the transaction timestamp. The fraud conditions are kept based on anomalies reported by the X-LSTM model. Figure 10 presents the capture of the fraud transaction (Call_Notify_Fraud_Detected(M_s) function), as depicted in Algorithm 1, and is indicated as RED box in the figure. Some common operating conditions include the execution of a transaction from a new location (not close to the user location), the transaction amount exceeding a specified threshold, and account debits amounting to multiple unknown parties. In the SC, there are two boolean functions *checkFraudTransaction*, which

checks the transaction as fraud or not based on the LSTM classification, and *getFraudStatus*, which reports the fake transaction details.

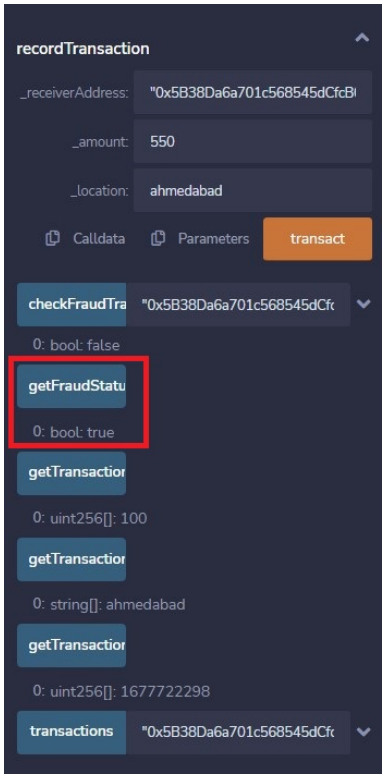


Figure 10. Fraud transaction SC functions.

5.6. BC Performance Metrics

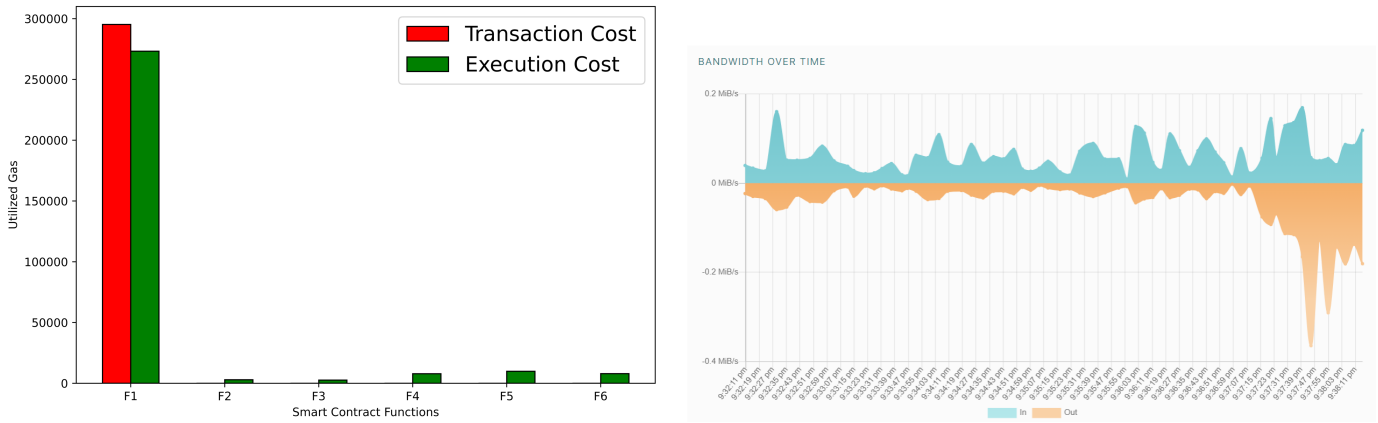
In this section, we discuss the performance of the BC, which stores the information of the SC details. We consider the gas value consideration for the SC design and the IPFS bandwidth for the analysis. The details are presented as follows. We forward the non-attack data to be stored on IPFS and the content hash on public BC post-classification [51]. Financial stakeholders authenticate the non-attack data in the contract, and the contract is executed.

5.6.1. Gas Value for SC

Gas is a unit of measurement for the computing work needed to execute the code in an SC on the BC network. Figure 11a presents the gas cost of transaction and execution. The intricacy of the code within an SC determines how much gas is needed for the contract to function. The quantity of gas a user is ready to pay for the transaction to be carried out is specified when they start a transaction that interacts with a smart contract. The transaction might fail, and all fees paid would be forfeited if the gas limit was too low. Conversely, the user will pay more fees than necessary if the gas limit is too high.

5.6.2. IPFS Bandwidth

IPFS is a peer-to-peer network where the data are stored and exchanged between nodes in the BC network. Figure 11b indicates the IPFS transfer and receive bandwidth over a while. When a user requests data from IPFS, the data are retrieved from the network of nodes rather than from a centralized server. This indicates that the data are dispersed across many nodes, which can speed up data retrieval. However, it also means that bandwidth is an important consideration for IPFS users, as the speed of data transfer will depend on the available bandwidth of the nodes on the network.



(**a**) (**b**)

Figure 11. (**a**) Gas cost consumption. (**b**) IPFS bandwidth utilization.

5.6.3. Scalability Graph

The Transactions Per Second (TPS) speed offered by the Blockchain Network (BN) is what determines how scalable a blockchain is. The suggested system's BC (Ethereum) and conventional blockchain (BCN) network scalability comparison graph is shown in Figure 12. The X-axis in this graph shows transaction time in milliseconds, and the Y-axis lists the number of transactions. The suggested method enables more transactions to be added to the BC. Moreover, IPFS can store a lot of data and fetch data much more quickly. Data are kept in IPFS, and IPFS data's hashes are sent to the BC. The proposed strategy using Ethereum Blockchain (EB) outperforms the conventional approach using bitcoin, according to graph visualization. This occurred as a result of bitcoin's lack of advanced technological features offered by the EB.

5.7. Potential Limitations and Future Scope

In this section, we discuss the potential limitations and future improvements of the proposed *RaKSha* scheme. The scheme offers the benefits of CC-FF detection via a proposed X-LSTM approach and then storing the results on SC executed on a public BC network. However, there are some potential limitations that we need to consider in the approach.

Firstly, using public BC for real-time financial data analysis might not be feasible owing to a large amount of real-time data collection (transactions) by financial stakeholders. Secondly, financial data are highly confidential and are subjected to financial laws, and thus, the privacy preservation of the records becomes a critical issue.

Secondly, the proposed approach requires a significant amount of computing power and resources to assure user scalability. Thus, it requires access to cloud servers for resources, which again jeopardizes the privacy and security of data storage. Thirdly, the proposed X-LSTM approach must be resilient to detect emerging CC fraud patterns. In this case, the model needs to continuously train and update itself to recognize the zero-day patterns, which might make the model bulky over time and limit its effectiveness. Thus, the presented limitations must be carefully studied while designing practical solutions for financial ecosystems.

Thus, the presented limitations open up new avenues for the future expansion of the proposed scheme. To address the issue of privacy preservation, the proposed scheme needs to incorporate random noise (differential privacy) to thwart any possible linkage attacks. To address the issue of the X-LSTM model learning and improve the model accuracy, more specific features must be generated by XAI models, which leads to the design of optimized XAI models that could improve the LSTM output. Finally, the proposed SC can be further optimized to improve the IPFS bandwidth, improving public BC networks' transaction scalability.

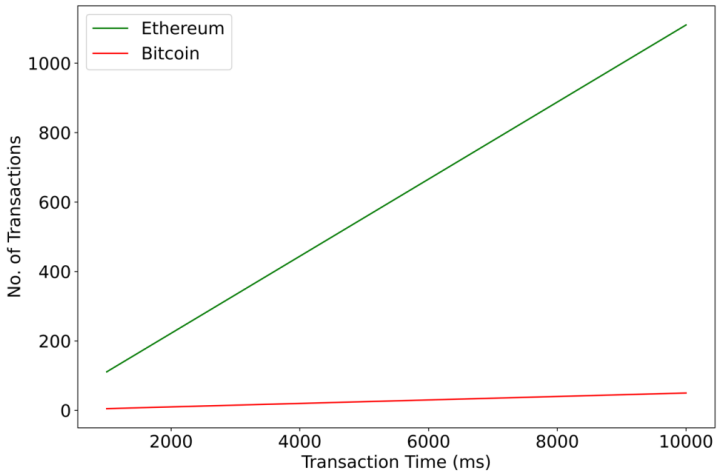


Figure 12. Scalability graph.

6. Concluding Remarks

The paper proposed a novel CC fraud detection scheme, *RaKShA*, in which we proposed an integration of XAI with the LSTM (X-LSTM) model, and the output is verified via SC. The results are stored in IPFS, which is referenced on the public BC network. The proposed approach addressed the limitation of traditional fraud detection by providing model interpretability, improved accuracy, security, and transparency. Modeling X-LSTM augmented the power of the LSTM model in CC-FF detection and made the scheme scalable and adaptable, which helps users to prevent themselves from FF. We validated the proposed layered reference scheme against two CC datasets and presented a comparative analysis of LSTM accuracy and loss (with and without XAI interpretation). For 500 epochs, an accuracy of 99.8% is reported via XAI, which shows an improvement of 17.41% on the simple LSTM model. The use of SC and public BC ensures that the fraud detection data are accessible and verifiable by all users, which makes the proposed scheme a useful CC-FF auditing tool at a low cost.

The presented scheme opens exciting opportunities to improve financial ecosystems' security and transparency barriers. The scheme applies not only to CC frauds but is extensible to insurance, tax evasion, and web transaction frauds. In different use cases, the underlying semantics remain common; however, fine-tuning the proposed scheme according to use case practicality needs to be considered for optimal solutions.

Author Contributions: Conceptualization: J.R., P.B., N.K.J., G.S., P.N.B. and S.T.; writing—original draft preparation: M.E., A.T., J.R., A.T. and M.S.R.; methodology: S.T., G.S., A.T. and P.N.B.; writing—review and editing: S.T., P.B., J.R., N.K.J. and M.S.R.; investigation: M.S.R., S.T., G.S. and M.E.; supervision: S.T., P.N.B. and P.B.; visualization; P.B., J.R., M.E., N.K.J. and M.S.R.; software: A.T., J.R., S.T. and M.E. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Researchers Supporting Project Number (RSPD2023R681) King Saud University, Riyadh, Saudi Arabia and also funded by the University of Johannesburg, Johannesburg 2006, South Africa.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Luo, G.; Li, W.; Peng, Y. Overview of Intelligent Online Banking System Based on HERCULES Architecture. *IEEE Access* **2020**, *8*, 107685–107699. [CrossRef]
2. Srivastava, A.; Singh, S.K.; Tanwar, S.; Tyagi, S. Suitability of big data analytics in Indian banking sector to increase revenue and profitability. In Proceedings of the 2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA) (Fall), Dehradun, India 15–16 September 2017; pp. 1–6. [CrossRef]
3. Yildirim, N.; Varol, A. A Research on Security Vulnerabilities in Online and Mobile Banking Systems. In Proceedings of the 2019 7th International Symposium on Digital Forensics and Security (ISDFS), Barcelos, Portugal, 10–12 June 2019; pp. 1–5. [CrossRef]
4. HAYES, A. Blockchain Facts. 2022. Available online: https://www.investopedia.com/terms/b/blockchain.asp (accessed on 2 March 2023).
5. Patel, S.B.; Bhattacharya, P.; Tanwar, S.; Kumar, N. KiRTi: A Blockchain-Based Credit Recommender System for Financial Institutions. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 1044–1054. [CrossRef]
6. Jiang, C.; Song, J.; Liu, G.; Zheng, L.; Luan, W. Credit Card Fraud Detection: A Novel Approach Using Aggregation Strategy and Feedback Mechanism. *IEEE Internet Things J.* **2018**, *5*, 3637–3647. [CrossRef]
7. Cao, R.; Liu, G.; Xie, Y.; Jiang, C. Two-Level Attention Model of Representation Learning for Fraud Detection. *IEEE Trans. Comput. Soc. Syst.* **2021**, *8*, 1291–1301. [CrossRef]
8. Fan, K.; Li, H.; Jiang, W.; Xiao, C.; Yang, Y. Secure Authentication Protocol for Mobile Payment. *Tsinghua Sci. Technol.* **2018**, *23*, 610–620. [CrossRef]
9. Wang, L.; Li, J.; Zuo, L.; Wen, Y.; Liu, H.; Liu, W. T-Tracer: A Blockchain-Aided Symbol Mapping Watermarking Scheme for Traitor Tracing in Non-Repudiation Data Delivery. In Proceedings of the BSCI'22: Fourth ACM International Symposium on Blockchain and Secure Critical Infrastructure, Nagasaki, Japan, 30 May–3 June 2022; pp. 23–34. [CrossRef]
10. Bhatttacharya, P.; Patel, K.; Zuhair, M.; Trivedi, C. A Lightweight Authentication via Unclonable Functions for Industrial Internet-of-Things. In Proceedings of the 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM), Gautam Buddha Nagar, India, 23–25 February 2022; Volume 2; pp. 657–662. [CrossRef]
11. Ferreira, F.G.D.C.; Gandomi, A.H.; Cardoso, R.T.N. Artificial Intelligence Applied to Stock Market Trading: A Review. *IEEE Access* **2021**, *9*, 30898–30917. [CrossRef]
12. Choi, D.; Lee, K. An artificial intelligence approach to financial fraud detection under IoT environment: A survey and implementation. *Secur. Commun. Netw.* **2018**, *2018*. [CrossRef]
13. Chauhan, K.; Jani, S.; Thakkar, D.; Dave, R.; Bhatia, J.; Tanwar, S.; Obaidat, M.S. Automated Machine Learning: The New Wave of Machine Learning. In Proceedings of the 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India, 5–7 March 2020; pp. 205–212. [CrossRef]
14. Xiuguo, W.; Shengyong, D. An Analysis on Financial Statement Fraud Detection for Chinese Listed Companies Using Deep Learning. *IEEE Access* **2022**, *10*, 22516–22532. [CrossRef]
15. Alarfaj, F.K.; Malik, I.; Khan, H.U.; Almusallam, N.; Ramzan, M.; Ahmed, M. Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms. *IEEE Access* **2022**, *10*, 39700–39715. [CrossRef]
16. Zhang, A.; Zhao, X.; Wang, L. CNN and LSTM based Encoder-Decoder for Anomaly Detection in Multivariate Time Series. In Proceedings of the 2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Xi'an, China, 15–17 October 2021; Volume 5, pp. 571–575. [CrossRef]
17. Saraswat, D.; Bhattacharya, P.; Verma, A.; Prasad, V.K.; Tanwar, S.; Sharma, G.; Bokoro, P.N.; Sharma, R. Explainable AI for Healthcare 5.0: Opportunities and Challenges. *IEEE Access* **2022**, *10*, 84486–84517.[CrossRef]
18. Tjoa, E.; Guan, C. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4793–4813. [CrossRef] [PubMed]
19. Mankodiya, H.; Obaidat, M.S.; Gupta, R.; Tanwar, S. XAI-AV: Explainable Artificial Intelligence for Trust Management in Autonomous Vehicles. In Proceedings of the 2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI), Beijing, China, 15–17 October 2021; pp. 1–5. [CrossRef]
20. Mankodiya, H.; Jadav, D.; Gupta, R.; Tanwar, S.; Alharbi, A.; Tolba, A.; Neagu, B.C.; Raboaca, M.S. XAI-Fall: Explainable AI for Fall Detection on Wearable Devices Using Sequence Models and XAI Techniques. *Mathematics* **2022**, *10*, 1990. [CrossRef]
21. Akram, S.V.; Malik, P.K.; Singh, R.; Anita, G.; Tanwar, S. Adoption of blockchain technology in various realms: Opportunities and challenges. *Secur. Priv.* **2020**, *3*, e109. [CrossRef]
22. Gupta, R.; Nair, A.; Tanwar, S.; Kumar, N. Blockchain-assisted secure UAV communication in 6G environment: Architecture, opportunities, and challenges. *IET Commun.* **2021**, *15*, 1352–1367. [CrossRef]
23. Gupta, R.; Shukla, A.; Tanwar, S. BATS: A Blockchain and AI-Empowered Drone-Assisted Telesurgery System Towards 6G. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2958–2967. [CrossRef]
24. Ketepalli, G.; Tata, S.; Vaheed, S.; Srikanth, Y.M. Anomaly Detection in Credit Card Transaction using Deep Learning Techniques. In Proceedings of the 2022 7th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 22–24 June 2022; pp. 1207–1214.
25. Arun, G.; Venkatachalapathy, K. Convolutional long short term memory model for credit card detection. In Proceedings of the 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 5–7 November 2020; pp. 1168–1172.

26. Tingfei, H.; Guangquan, C.; Kuihua, H. Using Variational Auto Encoding in Credit Card Fraud Detection. *IEEE Access* **2020**, *8*, 149841–149853. [CrossRef]
27. Fang, W.; Li, X.; Zhou, P.; Yan, J.; Jiang, D.; Zhou, T. Deep Learning Anti-Fraud Model for Internet Loan: Where We Are Going. *IEEE Access* **2021**, *9*, 9777–9784. [CrossRef]
28. Chen, J.I.Z.; Lai, K.L. Deep convolution neural network model for credit-card fraud detection and alert. *J. Artif. Intell.* **2021**, *3*, 101–112.
29. Balagolla, E.; Fernando, W.; Rathnayake, R.; Wijesekera, M.; Senarathne, A.N.; Abeywardhana, K. Credit Card Fraud Prevention Using Blockchain. In Proceedings of the 2021 6th International Conference for Convergence in Technology (I2CT), Maharashtra, India, 2–4 April 2021; pp. 1–8. [CrossRef]
30. Oladejo, M.T.; Jack, L. Fraud prevention and detection in a blockchain technology environment: Challenges posed to forensic accountants. *Int. J. Econ. Account.* **2020**, *9*, 315–335. [CrossRef]
31. Makki, S.; Assaghir, Z.; Taher, Y.; Haque, R.; Hacid, M.S.; Zeineddine, H. An Experimental Study With Imbalanced Classification Approaches for Credit Card Fraud Detection. *IEEE Access* **2019**, *7*, 93010–93022. [CrossRef]
32. Zheng, L.; Liu, G.; Yan, C.; Jiang, C.; Zhou, M.; Li, M. Improved TrAdaBoost and its Application to Transaction Fraud Detection. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 1304–1316. [CrossRef]
33. Van Belle, R.; Baesens, B.; De Weerdt, J. CATCHM: A novel network-based credit card fraud detection method using node representation learning. *Decis. Support Syst.* **2023**, *164*, 113866. [CrossRef]
34. Ni, L.; Li, J.; Xu, H.; Wang, X.; Zhang, J. Fraud Feature Boosting Mechanism and Spiral Oversampling Balancing Technique for Credit Card Fraud Detection. *IEEE Trans. Comput. Soc. Syst.* **2023**, 1–16. [CrossRef]
35. Labanca, D.; Primerano, L.; Markland-Montgomery, M.; Polino, M.; Carminati, M.; Zanero, S. Amaretto: An Active Learning Framework for Money Laundering Detection. *IEEE Access* **2022**, *10*, 41720–41739. [CrossRef]
36. Esenogho, E.; Mienye, I.D.; Swart, T.G.; Aruleba, K.; Obaido, G. A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection. *IEEE Access* **2022**, *10*, 16400–16407. [CrossRef]
37. Chen, L.; Jia, N.; Zhao, H.; Kang, Y.; Deng, J.; Ma, S. Refined analysis and a hierarchical multi-task learning approach for loan fraud detection. *J. Manag. Sci. Eng.* **2022**, *7*, 589–607. [CrossRef]
38. Ji, Y. Explainable AI Methods for Credit Card Fraud Detection: Evaluation of LIME and SHAP through a User Study University of Skövde, School of Informatics 2021. Available online: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1626230&dswid=4084 (accessed on 11 March 2023).
39. Ileberi, E.; Sun, Y.; Wang, Z. Performance Evaluation of Machine Learning Methods for Credit Card Fraud Detection Using SMOTE and AdaBoost. *IEEE Access* **2021**, *9*, 165286–165294. [CrossRef]
40. Cui, J.; Yan, C.; Wang, C. ReMEMBeR: Ranking Metric Embedding-Based Multicontextual Behavior Profiling for Online Banking Fraud Detection. *IEEE Trans. Comput. Soc. Syst.* **2021**, *8*, 643–654. [CrossRef]
41. Benchaji, I.; Douzi, S.; El Ouahidi, B.; Jaafari, J. Enhanced credit card fraud detection based on attention mechanism and LSTM deep model. *J. Big Data* **2021**, *8*, 151. [CrossRef]
42. Forough, J.; Momtazi, S. Ensemble of deep sequential models for credit card fraud detection. *Appl. Soft Comput.* **2021**, *99*, 106883. [CrossRef]
43. Kumar, M.S.; Soundarya, V.; Kavitha, S.; Keerthika, E.; Aswini, E. Credit Card Fraud Detection Using Random Forest Algorithm. In Proceedings of the 2019 3rd International Conference on Computing and Communications Technologies (ICCCT), Chennai, India, 21–22 February 2019; pp. 149–153. [CrossRef]
44. Bhavin, M.; Tanwar, S.; Sharma, N.; Tyagi, S.; Kumar, N. Blockchain and quantum blind signature-based hybrid scheme for healthcare 5.0 applications. *J. Inf. Secur. Appl.* **2021**, *56*, 102673. [CrossRef]
45. Afaq, S.; Rao, S. Significance of epochs on training a neural network. *Int. J. Sci. Technol. Res.* **2020**, *9*, 485–488.
46. Credit Card Fraud Detection. Available online: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud (accessed on 3 January 2012).
47. Credit Approval Data Set. Available online: https://archive.ics.uci.edu/ml/datasets/Credit+Approval (accessed on 3 January 2012).
48. Khanna, V.V.; Chadaga, K.; Sampathila, N.; Prabhu, S.; Bhandage, V.; Hegde, G.K. A Distinctive Explainable Machine Learning Framework for Detection of Polycystic Ovary Syndrome. *Appl. Syst. Innov.* **2023**, *6*, 32. [CrossRef]
49. Khatri, S.; Vachhani, H.; Shah, S.; Bhatia, J.; Chaturvedi, M.; Tanwar, S.; Kumar, N. Machine learning models and techniques for VANET based traffic management: Implementation issues and challenges. *Peer-to-Peer Netw. Appl.* **2021**, *14*, 1778–1805. [CrossRef]
50. Agarwal, N.; Das, S. Interpretable machine learning tools: A survey. In Proceedings of the 2020 IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, ACT, Australia, 1–4 December 2020; pp. 1528–1534.
51. Gupta, R.; Bhattacharya, P.; Tanwar, S.; Kumar, N.; Zeadally, S. GaRuDa: A Blockchain-Based Delivery Scheme Using Drones for Healthcare 5.0 Applications. *IEEE Internet Things Mag.* **2021**, *4*, 60–66. [CrossRef]