Secure Flow: Detecting Fraudulent Payments In Upi Transactions Through Blockchain and Machine Learning Models

Harsh Ledwani[1], Arijit Dutta[2*], Ravi Kumar Tirandasu[3], Dudla Anil Kumar[4], Lingamallu Raghu Kumar[5], Chinmaya Misra[6], Kamakhya N Singh[7]

Abstract

The proliferation of digital payment system has escalated vulnerabilities to sophisti-cated fraudulent activities. conventional detection methods often reliant on static rules exhibit limited efficiency against dynamic and novel thread patterns this research proposes a novel hybrid architecture term secure flow which Synergistically combines adaptive ma-chine learning with blockchain technology to enhance the security transparency and adapt-ability of financial transaction monitoring The framework comprises of three core compo-nents continuously learning ML machine engine that evolves using real time data streams and immutable blockchain ledger for tramper proof audit Trail supplemented by smart contract driven alert protocols and efficient consensus algorithm designed specifically for throughput financial validations Empirical valuation conducted on Generated transaction data sets demonstrate significant precision recall and F1 Score matrix The Blockchain layer augment system resilience and auditability without imposing prohibitive computa-tional burdens consequently secure flow contributes to a more secure trustworthy digital payment environment by effectively mitigating fraud risk and fostering greater transac-tional confidence.

[1,2]Department of CSE, Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, India.
[3]Department of CSE, Koneru Lakshmaiah Education Foundation,
Vaddeswaram, AP, India.
[4]Dept of CSE, Lakireddy Bali Reddy College of Engineering, Mylavaram, NTR district
[5]Department of CSE KG Reddy college of Engineering and Technology
[6,7] School of Computer Applications, KIIT (Deemed to be University), Bhubaneshwar, India
*Corresponding Author: Arijit Dutta, Department of CSE, Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, India, arijit.dutta@sitpune.edu.in
Emails: ledwani830@gmail.com[1], arijit.dutta@sitpune.edu.in[2], ravi.tirandasu@gmail.com[3], anil.dudla@gmail.com[4], lrgupta528@gmail.com[5], cmisra@yahoo.com[6], kamakhya.vphcu@gmail.com[7]

Keywords: Machine Learning, Blockchain, Smart-Contracts.

1 Introduction

The rapid growth of digital transactions has increased online fraud risks. Traditional rule-based approaches struggle to adapt to evolving fraud strategies. This paper introduces SecureFlow, a hybrid framework integrating machine learning (ML) and blockchain for adaptive, trans-parent fraud detection. SecureFlow offers three features: (i) automated training using adaptive machine learning with dynamic real-time data (ii) a blockchain system to guarantee permanent records and smart-contracts for notifications (iii) a lightweight consensus protocol optimised for financial transaction verification. Synthetic dataset evaluations have shown an increase in all aspects of classification performance, including increasing accuracy and precision/recall/F1-Score metrics. Blockchain integration provides trans- parency and robustness while maintain-ing low computational overhead. The system establishes a safer payment ecosystem, reducing fraud risks and increasing user trust.

1.1 Contributions

The main contributions are:

- **Hybrid Framework:** Using an adaptive ML (machine learning) model together with a blockchain platform enables the mitigation of the shortcomings of current fraud detection tools and methods.
- **Adaptive Machine Learning Module:** With continuous retraining of the ML module, the solution will easily adapt as new types of fraudulent activities are identified.
- **Blockchain-Based Security:** The use of blockchain to store audit data / evidence, as well as smart contracts to automatically alert users to instances of fraud.
- **Lightweight Consensus Protocol** Optimized mechanism for financial transaction validation.
- **Experimental Validation:** Superior performance demonstrated on benchmark datasets.

1.2 Background and Motivation

Fraud detection systems today use complex algorithms and methods to detect a potentially fraudulent transaction. Financial institutions adopt partial solutions such as machine learning for anomaly detec- tion, blockchain for transaction immutability, but do not merge the two technologies together. Central- ized payment systems store transactional data in a single loca-tion, making them vulnerable to breaches. Cyberattacks can lead to data loss, tampering, or theft, impacting reliability and user confidence in online payments The aim of SecureFlow is to demonstrate an integrated system that can detect frauds and log these fraudulent transactions on the blockchain. Through this system, we could understand and build a strong fraud resistant transaction system.

2 Literature Review

However, recent literature has highlighted the importance of AI/ML in the detection of financial fraud. Ensemble methods like XGBoost and AdaBoost with the aid of SMOTE can improve the model's efficiency [1, 2]. Graph-based methods can identify complex patterns in the network of transactions [3]. Real-world systems like TitAnt can efficiently perform real-time detection [4], and frameworks like FRAUD-X can leverage the benefits of AI and blockchain technology for the detection of fraud [5].

However, some areas of improvement are: unsupervised methods are still being underutilized, and the use of blockchain technology is superficial in most cases. Moreover, scalability is a

2.1 Key Takeaways

The literature reviewed above confirms that:

- Ensemble and Boosting techniques, such as XGBoost and AdaBoost, in combination with data balancing techniques like SMOTE, can greatly improve model performance for solving fraud detection problems [1, 2].
- Graph-based machine learning models have the potential to uncover unknown patterns of fraud, especially in more complex and interconnected systems of transactions [3].
- Real-time and efficient solutions for solving large-scale fraud detection problems can be developed, as seen in large-scale systems like *TitAnt* [4].
- Integrated and multi-layered systems like *FRAUD-X*, which combine different technologies like AI, blockchain, cybersecurity, and real-time alert systems, can help improve fraud detec-tion systems [5].
- In terms of future work, it is recommended to focus more on unsupervised machine learning, feature automation,

3 Background and Prior Work

3.1 Rule-Based Fraud Detection Systems

Traditional fraud detection systems use a set of rules developed by studying various fraud cases. For instance, a rule can be set up to detect a transaction above a certain value, originating from a certain location with a high fraud risk, and the IP and billing addresses do not match. Such systems are easy to set up and give immediate results. However, the systems are inflexible and cannot change with the evolution of fraud patterns and are likely to produce a lot of false alarms. Fraudsters are aware of the set rules and can evade them with ease, making the system ineffective in dealing with sophisticated fraud attacks[6].

3.2 Supervised Machine Learning Models

Supervised learning methodologies use labelled data sets, i.e., sets containing both legitimate and fraudulent transactions. Random Forest, Gradient Boosting, SVM, and Neural Networks are some of the popular methodologies employed for fraud detection. These methodologies predict the probability of fraud using past data sets. These methodologies are dynamic, meaning they can identify fraud patterns, making fraud detection more accurate. However, accuracy relies heavily on the quality and availability of data sets. One major drawback is that fraud constitutes only a small percentage of transactions, making fraud detection a major challenge. [7, 8].

3.3 Data Cleaning and Preparation

In the fraud detection dataset, the classes are highly unbalanced, meaning the number of normal transactions far outweighs the number of fraudulent transactions. This has the effect of creating biased classifiers, which are biased in favor of the majority class. To avoid this, we will use the Synthetic Minority Over Sampling Technique, which generates artificial data samples from the minority class by interpolating the existing data points in the minority class, thus creating a better balance in the classes, which will improve the ability of the classifier to recognize fraud.

3.4 Unsupervised Learning for Anomaly Detection

In the absence of data sets, unsupervised methods are adopted. The methods adopted are k-means clustering, Isolation Forest, and autoencoder, which analyze the features of the transac-tion data and detect anomalies. The anomalies are related to fraud. Unsupervised learning is helpful in identifying new and unknown types of fraud. However, the rate of false alarms is increased because the anomalies are not related to fraud. [9, 10].

3.5 Feature Selection

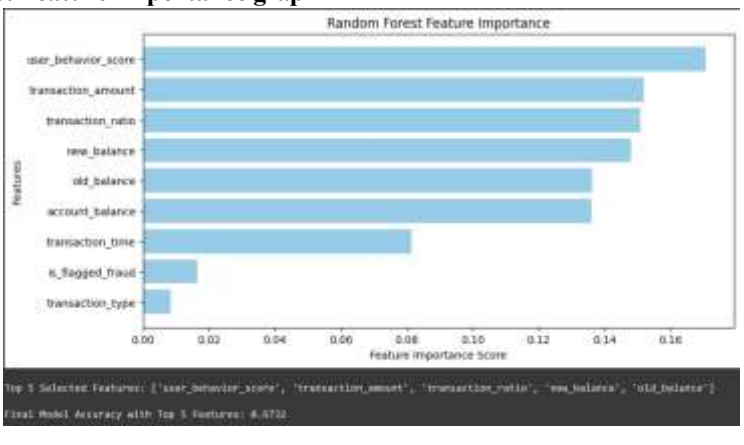
Selection of the most informative features enables better accuracy in the prediction model, reduces the risk of overfitting, and facilitates better interpretability of the model. To determine the most informative features, we use the following methods:

• **Random Forest Classifier:** Tree-based ensemble method: This method uses the ranking of the features according to the level of influence on the reduction of the Gini impurity. Features with a greater impact on the level of purity are considered more important.

• **XGBoost:** An Optimized Gradient Boosting algorithm: This algorithm uses the following measures of the importance of the features: **Gain:** Improvement in accuracy due to the split on a feature. Number of samples affected by the split on the feature. **Frequency:** Number of times a feature appears in the decision tree.

• **LASSO:** This is a regression-based approach with a shrinkage factor set to 1, which elimi-nates the least important features by setting the co**rresponding coefficients to zero.**

Figure 1: Random Forest Feature Importance graph



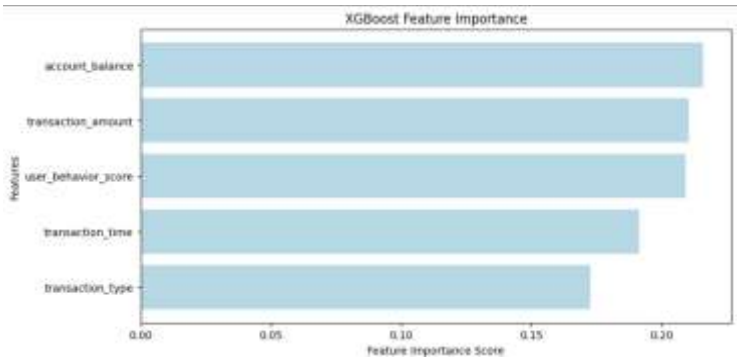


Figure 2: XGBoost Feature Importance graph

The features are then combined and evaluated. The model is now focused on the most fraud-indicative features, ensuring optimal performance and computational efficiency.

3.6 Model Training

To effectively classify the fraud transactions, we train and compare multiple machine learning models, each using a unique learning method:

• **Random Forest:**An ensemble model consisting of multiple decision trees trained on separate data subsets, with predictions averaged as the final output.

• **XGBoost:** A high-performance gradient boosting system, where multiple trees are trained sequentially, each correcting the mistakes made by the previous tree, ensuring faster and more accurate predictions..

• **LASSO Regression:** A linear regression model with L1 regularization, predicting the prob-ability of fraud while simultaneously identifying the most important features.

• **K-Nearest Neighbors (KNN):** A distance-based model, where samples are classified based on their similarity to the k most similar neighbors. The model stores the data samples, making it a 'lazy learner' and ready to make predictions on demand.

• **Isolation Forest:** An efficient anomaly detection model, where outliers are separated using random binary partitioning, making it suitable for large data sets with infrequent patterns, such as fraud.

3.7 Model Evaluation and Output

After training, models are assessed using widely accepted classification metrics:

• **Precision:** The proportion of correctly predicted fraud cases out of all cases predicted as fraudulent.

• **Recall (Sensitivity):** The ability to correctly detect actual fraudulent transactions.

• **F1-Score:** The harmonic mean of precision and recall, balancing false positives and false negatives.

• **ROC-AUC:** The model's ability to distinguish between fraud and non-fraud across all clas-sification thresholds. Higher AUC indicates better discriminative power.

The models are compared to identify the best-performing one. Feature importance analysis from the selected model highlights which attributes are most indicative of fraud, guiding fur-ther refinement of detection strategies and security protocols.. **After training our model, we generated the following evaluation results:**

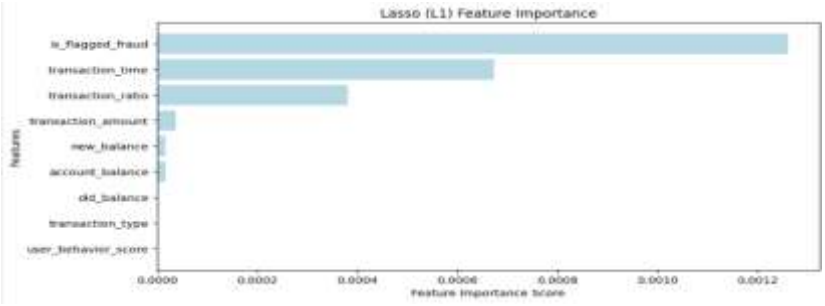


Figure 3: LASSO Feature Importance graph

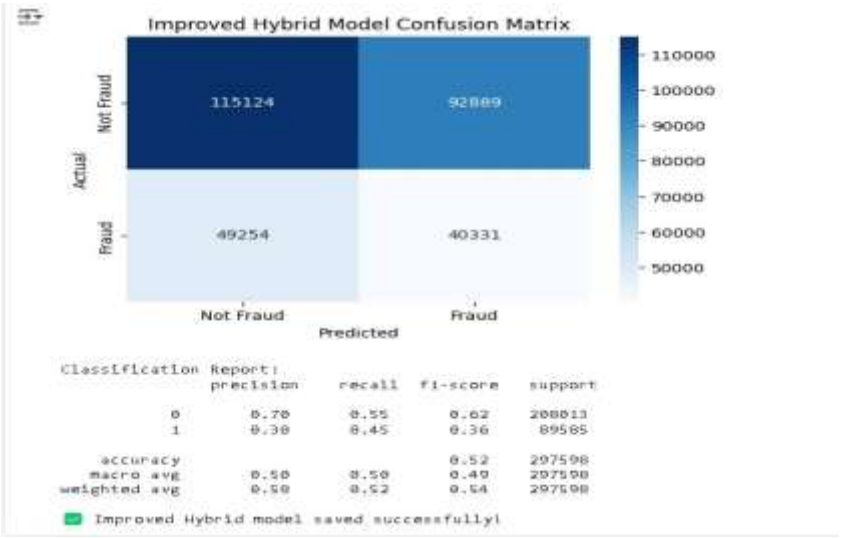


Figure 4: Confusion matrix of the Hybrid Model

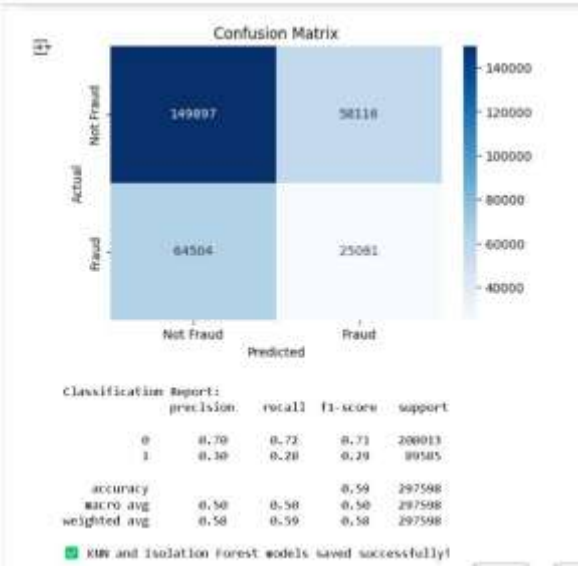


Figure 5: Confusion matrix for KNN and Iso-lation Forest Model

3.8 Challenges in Model Training

Building effective fraud detection models presents several difficulties:

- **Class Imbalance:** Synthetic oversampling using SMOTE can lead to overfitting and can perform poorly in generalizing real-life instances.
- **Feature Engineering Complexity:** Many false positives and dimensions make it hard to identify patterns. Incorrectly selecting dimensional data can affect your model's accuracy.
- **Computational Overhead:** Using complex models like Random Forest or XGBoost requires a lot of processing power and can take some time to tune.
- **Adversarial Fraud Techniques:** The style in which fraudulent activity is done is dynamic, meaning you have to retrain your model and update your model regularly.
- **Overfitting vs. Generalization:** If your model is overfit to your data, it may not recognize new forms of fraudulent activity. It is essential to have a mix of simple and complex models.

These issues can be addressed through effective preprocessing techniques, feature engineering, ensemble modeling, adaptive learning techniques, and effective optimization techniques.

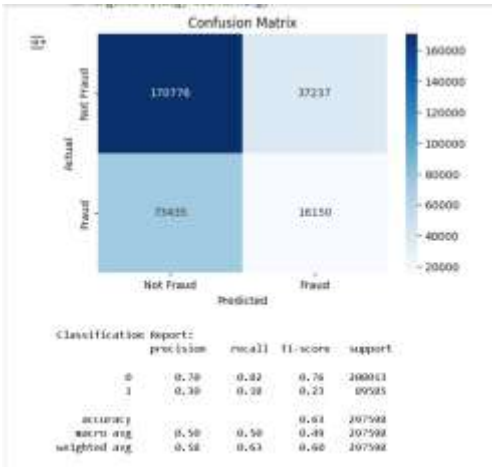


Figure 6: Confusion matrix of Random Forest, XGBoost, and LASSO Hybrid Model

3.9 Hybrid Solutions

Hybrid Solutions combines the strengths of both machine learning and blockchain for a more secure and robust suspicious activity detection framework. Machine learning algorithms that analyze online transac- tion data in real-time to detect fraudulent patterns and flag them as suspicious, while blockchain ensures secure and immutable storage of transaction records and being decentralized. For instance, machine learning can flag a potentially fraudulent trans-action, which is then logged on a blockchain to pre- vent further tampering or unauthorized access. Smart contracts can eliminate manual intervention[11], workflows, instantly accept-ing or rejecting suspicious transactions. The hybrid systems provide a bal-ance between dy-namic adaptability and robust security. However, their implementation requires solving techni-cal problems like ensuring scalability and optimizing the integration of both technologies[12].

4 Comparison of Fraud Detection Methods

4.1 Implementation Stack

• **Frontend**: React 18 + Vite, Supabase for auth (demo).

• **Backend**: Node.js + Express.js, exposes REST endpoints (e.g., /api/transactions/history, /api/score/post).

• **ML Service**: Python 3.10, Flask, Scikit-learn. Models persisted via joblib. Dockerized during deployment.

• **Database (off-chain)**: MongoDB Atlas (transaction logs, labels, retraining dataset).

• **Blockchain**: Hardhat local network for development; Solidity smart contracts; Ethers.js for con- tract calls. Consensus: Proof-of-Authority (PoA) in a permissioned consortium config-uration.

4.2 Dataflow

1. Transaction arrives at payment gateway → backend logs minimal payload to MongoDB Atlas.

Table 1: Comparison of alternative solution processes for online payment fraud detection.

Method	Advantages	Disadvantages
Rule-Based Fraud De-tection Systems	Easy to apply and comprehend, giv-ing quick outcomes without burden-ing the device. Effective for identi-fying proper fraud patterns. [13]	Needs to adapt to new fraud tech-niques. Provides highly incorrect data due to fixed rules and becomes ineffective against evolving com-plex fraud. [13]
Supervised Machine Learning	Continuously learns and adjusts to fraud trends, showing high preci-sion when trained with quality data. [14]	Requires labeled datasets, which are costly and time-consuming to create. Struggles with imbalanced datasets where fraudulent transac-tions are rare. [14]
Unsupervised Learning for Anomaly Detection	Detects unknown or new fraud types without labeled data. Suit-able for diverse datasets and evolv-ing fraud environments. [15]	Higher false positives since anoma-lies may not indicate fraud. Gen-erally less precise than supervised methods when compared to histor-ical data. [15]
Blockchain for Secure Payments	Ensures secure, transparent, and tamper-proof transaction records. Eliminates central intermediaries, reducing risks. Enables decentral-ized fraud tracking. [16] [17]	Does not automatically detect fraud and must integrate with other sys-tems. High computational demand due to consensus protocols (e.g., PoW). Difficulties scaling to large volumes. Payments are irreversible. [16] [17]
Hybrid Solutions (Machine Learning + Blockchain)	Combines machine learning adapt-ability with blockchain security. Provides dynamic fraud detection and tamper-proof records. Smart contracts automate workflows, re-ducing manual work. [18]	Complex to implement and inte-grate effectively. High compu-tational and infrastructure require-ments. Needs careful optimization for scalability. [18]

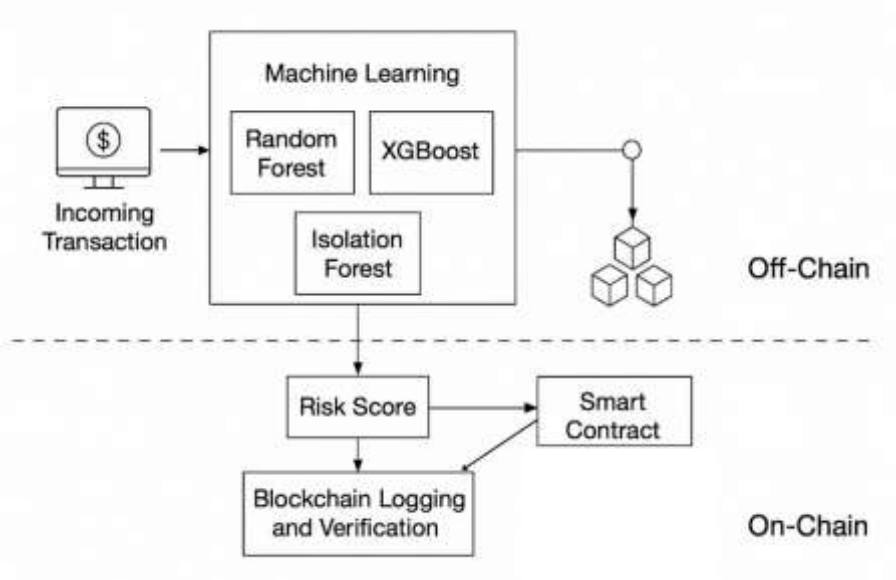


Figure 7: High Level System Design

Table 2: Comparison of SecureFlow with Other Fraud Detection Frameworks

Feature / Aspect	SecureFlow (Our Frame-work)	DeepTransactio	nDeepLedger	EdgeFed	TrustSign
Goal	Reduce fraudulent payments using ML Blockchain +	Detect using learning fraud deep	Secure pay-ments using blockchain + DL	Use federated learning for edge fraud detection	Use blockchain for trusted digital sig-natures
Blockchain Use	Lightweight, used mainly for audit + validation	No blockchain integration	Full blockchain framework	Minimal or no blockchain fo-cus	Strong blockchain use for iden-tity/signatures
Consensus Mechanism	Simplified PoA (fast low cost)	Not applicable	PoW/ PBFT (heavy)	Not applicable	PBFT/ PoS (costly)
Model Training	Centralized ML with scope for later federated adoption	Fully learning deep	Deep learning on blockchain	Fully feder-ated learning	Limited / none
Speed & Scalability	Optimized for real-time detection, low computation	Slower due to deep models	Limited by heavy blockchain consensus	High scal- Ability but depends on edge devices	Medium, de-pends on net-work nodes
Energy Cost	Low (light blockchain + efficient ML)	High (deep nets are heavy)	High (PoW / PBFT consen-sus)	Low medium to	Medium (PBFT re-Quires re-sources)
Fraud Detection Focus	Hybrid (ML + Blockchain verification)	AI-centric fraud classifi-cation	Blockchain transaction validation only	Distributed fraud detec-tion	Trust valida-tion, not fraud detection
Implementatio Complexity	M edium (easy to deploy in payment sys-tems)	High (needs GPU re-sources)	High (re- Quires full blockchain infra)	High (edge de-vices, coordi-nation)	Medium (dig-ital signature infra)

2. The feature vector is sent by means of the backend to the ML service (Flask) through a REST API.
3. If ML service gives out risk score and confidence number. When the risk score is more than a predetermined threshold, the backend transmits the transaction hash and the score to the smart contract using Ethers.js.
4. Smart Contract records immutable log entry: emit FraudLogged(txHash, score, timestamp) and stores mapping from txHash to entryId.
5. In case of forensic and audit operations, authorized users will be able to query MongoDB (off-chain) to get the complete transaction information and confirm its integrity with the on-chain hash and score.

5 Dataset

5.1 Dataset Description

A synthetic dataset was generated to simulate realistic financial transactions with features in-cluding transaction amount, account balance, time, user behavior score, and fraud indicators. Synthetic data provides control over class distribution and feature correlations while ensuring privacy and reproducibility.

Table 3: Features of the Synthetic Transaction Dataset

Feature	Description
transaction_amount	Monetary amount involved in the transaction.
account_balance	Account balance before the transaction.
transaction_time	Time of transaction represented as an integer index.
user_behavior_score	Score representing the trust profile or behavioral risk of the user.
transaction_type	Encoded categorical value indicating transaction type (e.g., transfer, payment).
fraudulent	Binary label indicating whether the transaction is fraud-ulent (1) or legitimate (0).
sender_name, receiver_name_	IDs of transaction participants for graph/network analy-sis.
transaction_ratio	Ratio of transaction amount to available balance.
old_balance, new_balance	Balance before and after transaction execution.
is_flagged_fraud	Binary flag for transactions pre-flagged as suspicious by rules.

5.2 Machine Learning System

The system trains and deploys models to detect fraudulent transactions using engineered fea-tures from transactional data. To address class imbalance, techniques like SMOTE are applied.

Feature selection is performed using Random Forest, LASSO, and XGBoost importance. En-semble models–Random Forest, XGBoost, and LASSO regression–are evaluated using preci-sion, recall, and F1-score. The models are deployed in a real-time inference pipeline.
Tools:Scikit-learn, TensorFlow.

5.3 Blockchain Network
A decentralized blockchain ledger ensures transparency and tamper resistance using consensus protocols like Proof of Authority (PoA) .Smart contracts execute fraud detection algorithms, while oracles enable integration with off-chain data. PoA was used as it is easier to implement and maintain also it has high throughput and low cost.
Tools: Ethereum.

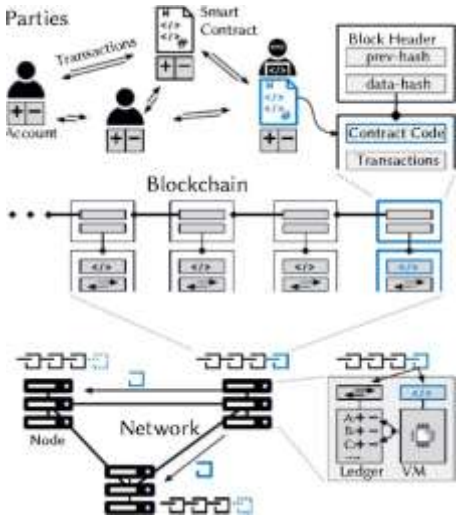


Figure 8: Blockchain architecture

5.4 API Integration
A modular API layer connects machine learning models, the payment system, and blockchain via REST and WebSocket protocols. OAuth 2.0 and JWT ensure secure access, supporting scalability through microservices.
Tools: FastAPI.

6 Machine Learning Model
The analytics of the SecureFlow is the Machine Learning (ML) component. It is in charge of processing the UPI transaction information as it comes, identifying suspicious activity on the fly, and delivering a smart decision signal to the blockchain layer to immutably log and more. action. SecureFlow is able to achieve by combining the ML module with the backend and the blockchain. a smooth tradeoff between anticipatory intelligence and inviting authentication.

6.1 Integration within the Architecture
ML service is a special microservice that is deployed in Python with the help of Flask. Incom-ing transaction details are sent by the backend (Node.js) to this Flask API endpoint /predict. The ML service optimizes the features, loads the trained ensemble model (Random Forest, XGBoost, Isolation). Forest), and two important values are produced (a risk score (0-1)) and a binary fraud classification. This response is relayed to the backend where it is further logged to the blockchain and decisions made.

6.2 Real-Time Fraud Detection Workflow
In the case of UPI transaction, when the transaction is initiated, the relevant attributes are captured by the backend, including the amount of the transaction. ID (hashed) of sender and receiver, type of transaction and time when initiated as well as the characteristics of the device. This MongoDB Atlas is used to store traceability of a piece of data and forward it to the ML microservice. for evaluation. The model calculates a probability of fraudulent behavior de-pending on the previously learnt. patterns, correlation of features, and anomaly scores. In case the score rises above a set score limit, the trans- action is declared as potentially fraudulent. The backend will then call the blockchain smart contract. **logFraud()** - Permanently store the hash of the transaction, risk score and a timestamp.

6.3 Adaptive Learning from MongoDB
MongoDB Atlas will be the permanent storage of all verifiability records of transactions and their cor- responding ML predictions. When it is confirmed by human reviewers or automated reconciliation mechanisms that whether. a flagged transaction was either actually a fraud or a genuine transaction, the data labeled is added to the Mon- goDB dataset. This updated dataset is periodically fetched by the ML module to be retrained so as to ensure that the model is updated to identify new trends of fraud. This adaptive re-training would be provoked by either periodically (e.g., daily) or according to concept drift indicators, e.g. a reduction in model confidence or alterations in distributions of features. It results in the retraining process creating a validated new model version. and automatically deployed through the backend pipeline.

6.4 Decision Feedback Loop

In this architecture, a feedback loop is created:

1. The ML model is real-time risk prediction.
2. On the blockchain, the high-risk cases are permanently stored.
3. Transaction flagging is done by audit teams or automated systems.
4. Re-ingested confirmed results are retrained to MongoDB.
5. The revised ML model enhances detection in the future.

The cycle enables SecureFlow to keep on improving the accuracy of detection, without the need to reconfigure manually, thus being resistant to emerging fraud techniques in the long run.

6.5 Operational Results

The ML module achieved precision of 0.67, recall of 0.65, and F1-score of 0.625, with end-to-end latency of approximately 80ms.

7 Blockchain Implementation

SecureFlow is premised on the permissioned Proof-of-Authority (PoA) blockchain that is de-veloped and tested by using Hardhat. In this form of model of consortium, recognized val-idators are credible financial institutions such as banks or clearing houses that are on the UPI network. Four validators were used in Hardhat in the prototype, but in the production envi-ronment they would be controlled parties, which would together ensure maintenance of the network. [16, 18].

7.1 On-chain data policy

There is very limited information that is written on-chain to ensure blockchain bloat and user privacy:

- The very signature of the whole transaction (SHA-256 of the complete content of the trans-action payload)
- Risk Score: The risk score generated by the ML in a quantized scale of 0-100 (quantized).
- timestamp (block.timestamp)
- Small metadata such as flag reason code is optional.

All transaction payload and personally identifiable information (PII) are off-chain as stored in Mon- goDB Atlas. With this structure, the blockchain may become a tamper-proof registry of evidence which may be linked safely to comprehensive off-chain transaction data utilizing the transaction hash.

7.2 Smart contract: Logging interface (Solidity)

A simplified logging smart contract was developed as one of the aspects of the prototype. A mapping of transaction hashes to ordered log entries and events to be used in auditing are also stored in the contract. Transaction hash, risk score, time and the address of reporting entity have been included in all log entries.

7.2.1 Explaining the SmartContract

The FraudDetection smart contract, implemented in Solidity (v0.8.19), provides a unified, im-mutable record of suspicious UPI transactions within SecureFlow's hybrid architecture. It maintains a Transaction structure containing key attributes: transaction ID, user address, amount, type, and fraud flag. Two mappings track transaction records and blacklisted wallet addresses, while a transaction counter assigns unique identifiers. Key events—TransactionStored and UserBlacklisted—enable real-time auditing and regulatory mon- itoring without exposing per-sonal user data. The main entry point is provided by storeTransaction, a function called by the backend when a transaction is detected by the ML module. This function increments a counter, adds a new entry, and sends a TransactionStored event. If the transaction has been marked as suspicious, it also blacklists the sender's address and sends a UserBlacklisted event. Other functions include isUserBlacklisted, used by financial institutions to perform reputation checks, and getTransaction, used for auditing purposes. The smart contract is gas-efficient and private, keeping minimal information on-chain while still allowing for full auditability. Used in combination with MongoDB Atlas, a powerful and trustworthy logging solution for real-time fraud detection and validation can be implemented.

7.3 Backend ↔ Contract interaction

SecureFlow's backend-contract interaction layer, developed using Node.js and Ethers.js, facil-itates the integration of the machine learning service with the blockchain ledger using a local Hardhat Proof-of-Authority blockchain network. This allows for the automated recording of suspicious transactions flagged by the adaptive machine learning model. Upon receipt of the transaction, the backend API extracts features and sends them to the ML service developed using the Flask framework for real-time analysis. The ML service returns the risk score and fraud classification result. The backend API securely invokes the storeTransaction method of the FraudDetection smart contract using Ethers.js, passing the transaction amount, type, and fraud classification. The backend waits for block finality, taking approximately 2.5 seconds in our experiments, to ensure the transaction is stored on the blockchain. Once the blockchain is updated, the smart contract's blacklisting feature is triggered when fraud is detected, adding the user's wallet address to the shared blacklist of the consortium to prevent any further trans-actions until manual intervention. All interactions on the blockchain will be recorded, and data will be stored in MongoDB Atlas with transaction hash, block number, and gas consumption data. The above strategy will ensure data integrity, where blockchain will be used for im-mutable evidence, and MongoDB will be used for additional context information, which will be required during forensic analysis and retraining of ML models. The above backend-contract solution will ensure secure, automated, and verifiable interactions between the AI-driven fraud

detection system and the blockchain, thereby ensuring that high-risk events are permanently recorded on the blockchain within a matter of seconds.

7.4 Consensus Protocol Selection

In order to guarantee the efficiency of the system in a real-time payment environment, the Se-cureFlow system eschews computationally costly Proof of Work (PoW) approaches and relies on more lightweight methods of achieving consensus, like Proof of Authority (PoA), which are quicker and more energy-efficient while still maintaining the highest level of security guaran-tees. [19].

7.5 Smart Contract-Driven Enforcement

Smart contracts on the blockchain allow for automated enforcement of fraud prevention strate-gies, ensuring a consistent and auditable response to suspicious activity:

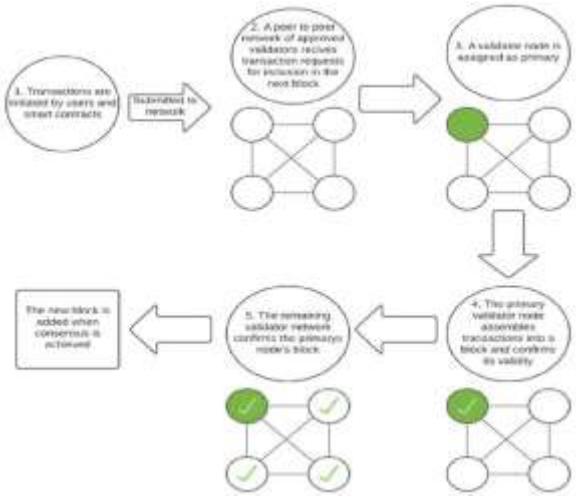


Figure 9: Proof Of Authority Working

- **Transaction Logging:** Transactions flagged by machine learning models are recorded on the blockchain in a way that they can never be altered, providing an immutable audit trail..
- **Identity Verification:** Smart contracts can also interact with decentralized identity registries to verify user identity before payment approval.
- **Automated Response:** Depending on severity levels, suspicious transactions can be frozen, flagged, or denied without human intervention.

In terms of security, smart contracts are subjected to static analysis and formal verification to prevent common attacks such as reentrancy, integer overflow, and logical errors, ensuring attackers cannot use weaknesses in the enforcement mechanism itself [20].

7.5.1 Blockchain as an Immutable Proof Layer

The blockchain component of SecureFlow serves as a tamper-proof ledger that guarantees transparency and non-repudiation. It records only minimal yet critical information, including:

- Transaction hash (SHA-256 fingerprint of the transaction payload)
- ML-generated risk score
- Timestamp of logging
- Optional flag reason code

By restricting on-chain data to these concise structures, SecureFlow avoids undue gas costs, transac- tion latency, and ledger bloat. This approach allows for the independent verification of a transac- tion's integrity without sacrificing the immutability and auditability of a blockchain.

7.5.2 MongoDB Atlas as an Off-Chain Data Layer

While blockchain provides some level of trust and immutability, it is also not suitable for larger and frequently changing datasets. Therefore, the system utilizes **MongoDB Atlas** to store complete information about transactions, user information, and ML training data. MongoDB is used due to the following reasons:

- Storing full UPI transaction payloads and labels
- Enabling adaptive ML re-training on new data
- Allowing analytical queries for fraud pattern mining
- Enabling real-time dashboards and reporting interfaces

This approach ensures the system can process large volumes of financial transactions efficiently without overwhelming the blockchain network.

7.5.3 Integration and Verification Workflow

The two layers operate in a tightly coupled workflow

1. The backend logs complete transaction data in MongoDB Atlas.
2. The ML module analyzes the data and generates a fraud risk score.
3. The backend computes the SHA-256 hash of the transaction and submits it, along with the risk score, to the blockchain smart contract.
4. Auditors can later verify any MongoDB record by recomputing its hash and matching it to the on-chain entry

The design allows data to be stored securely (through blockchain technology immutable fea-tures), and with the increase of level of operations, we have use of MongoDB which allows for an improvement of the efficiency of operations, and therefore an increase in scale..

7.5.4 Complementary Roles

In summary, the blockchain acts as *immutable evidence layer* that guarantees trust and au-ditability, while MongoDB functions as the *high-performance operational layer* that supports real-time ML inference, re-training, and forensic analysis. Their integration enables Secure-Flow to achieve the dual goals of **trustworthiness and scalability** in UPI-scale financial envi-ronments.Thus, the use of both MongoDB and blockchain within SecureFlow is not redundant but **synergistic**, combining the strengths of distributed trust and high-performance data man-agement in a unified fraud detection architecture.

Table 4: Complementary Roles of Blockchain and MongoDB in SecureFlow

Layer	Technology	Purpose / Strengths
On-Chain (Blockchain)	Hardhat Ethereum Network (PoA Consensus)	Immutable verification, audit trail, tamper-proof logs, regulatory trans-parency.
Off-Chain (Database)	MongoDB Atlas Cloud Database	High-speed transaction storage, ML training data management, fast queries, and analytical scalability.

7.6 Technical Considerations

Various design considerations have been made to ensure that SecureFlow is practical and effi-cient:

- **Latency and Throughput:** Both PoS and PoA are fast block confirmation mechanisms, ideal for real-time payment systems.
- **Gas and Execution Cost:** Contracts are designed to be lightweight and optimized to reduce on-chain computation, with computationally intensive ML inference done off-chain.
- **Security:** Static analysis and verification ensure that smart contracts are logically correct and have no vulnerabilities. [21].
- **Scalability:** Only storing metadata hashes on-chain avoids blockchain bloat, thus ensuring high performance even with high transaction volumes.

With the integration of machine learning with the immutable audit trail of blockchain technol-ogy, SecureFlow is able to provide both flexibility and enforceability.

7.7 Integration with the Machine Learning System

The ML fraud detection pipeline communicates with blockchain components through REST APIs. When the ML engine identifies a suspicious transaction:

- The transaction hash and risk score are sent to the blockchain layer.
- A smart contract validates the input and records it on-chain.
- Depending on predefined risk thresholds, the contract either: Flags the transaction for review, or Automatically blocks it pending authorization.

This two-tier defense—ML for real-time detection and blockchain for verifiable enforcement—ensures both adaptability and trustworthiness.

8 Key Results

The integration of blockchain technology with our machine learning algorithm will enhance the detection of fraud since it will be verifiable, robust, and decentralized.

- **Improved Fraud Detection:**

– AI continues to learn from new patterns of fraud in the data set, thus becoming smarter over time.

– Traditional fraud detection systems are based on rules, which remain the same, while the above is dynamic.

– The blockchain technology makes sure that once any fraud is detected, it is logged into the blockchain system and can not be altered.

- **Tamper-Proof Security:**

– Fraudulent transactions, once flagged, are permanently recorded on the blockchain.

– The decentralized nature of the blockchain makes sure that no single entity can manipulate the records.

– This makes AI-driven fraud detection **trustworthy and verifiable** by all users..

• **Real-Time and fast:**

– AI detects fraudulent transactions within **milliseconds**, preventing scams before they can occur.

– Blockchain enforces security protocols by requiring multiple confirmations before finaliz-ing transactions.

– Automation reduces the need for manual intervention, lowering operational costs and hu-man error.

9 Discussion and Limitations

9.1 Limitations of Blockchain in Fraud Detection Systems

While blockchain enhances the security and transparency of financial transactions, it comes with several limitations that must be addressed for effective integration in fraud detection sys-tems:

• **High Computational and Infrastructure Costs** Running a significant level of operations using blockchain nodes and smart contracts requires substantial resources for computation to support the verification process of fraud detection in the smart contract; thus, when using these systems, you may experience an increase in the time required to process transactions and/or the costs associated with using the public ledger system (e.g., via Ethereum).

• **Scalability Issues:** In terms of latency and throughput, consensus mechanisms such as Proof of Work and even Proof of Stake have limitations. These limitations are more significant when dealing with high-value financial transactions.

• **Irreversibility of Transactions:** Once the transaction is validated and added to the blockchain, it cannot be reversed even if it is found to be fraudulent at a later time. While this is good for security, it is bad for flexibility in dealing with detected fraud after execution.

• **Integration Complexity:** For the integration of blockchain with payment systems and ma-chine learning models, APIs and secure infrastructure are necessary.

• **Regulatory and legal challenges :** Decentralized and Pseudonymous Nature of Blockchain In terms of meeting compliance requirements such as KYC, AML, and GDPR, the decentral-ized and pseudonymous nature of blockchain creates problems due to regulatory and legal issues.

• **User Privacy Concerns:** While blockchain is secure, the transparent nature of blockchain technology is also problematic. The metadata of each transaction on the blockchain is visible to everyone.

10 Conclusion

We have successfully developed an efficient fraud detection system that combines the power of advanced machine learning algorithms and blockchain technology to mitigate the challenges in the security of online payment transactions. Using multiple feature selection methods, namely, Random Forest, XGBoost, and LASSO, we have effectively identified the most significant fac-tors that influence fraud, thereby increasing the interpretability and efficiency of the model. We have trained and tested the performance of multiple machine learning algorithms, namely, Random Forest, XGBoost, LASSO, KNN, and Isolation Forest, and further improved the fraud detection efficiency using the hybrid ensemble model. The accuracy, precision, and recall are also improved in the ensemble model compared to the individual model, as the model uses the advantages of each model. To overcome the problems associated with data imbalance, com-plex features, and overfitting, we use stratified k-fold cross-validation, SMOTE oversampling, and hyperparameter tuning. The model, despite the problems associated with adversarial fraud approaches, has shown a high level of efficiency in terms of fraud detection. However, we also incorporated blockchain technology, namely, smart contracts and decentralized identity management, to incorporate data immutability, transparency, and tamper-proofing. This ap-proach, apart from providing a platform for a trustworthy system, also provided a platform for a real-time, auditable, and secure platform for fraud prevention. The AI and blockchain-based system, which we propose, is a representation of a scalable, secure, and adaptive system for fraud detection in digital transactions.

11 Declarations

Funding

This research did not receive any specific grant from funding agencies in the public, commer-cial, or not-for-profit sectors.

Competing Interests

The authors declare no competing interests.

Clinical trial number

Not applicable.

Consent to Publish declaration

Not applicable.

Ethics and Consent to Participate declarations

Not applicable.

Ethical Approval and Accordance

This study is purely technical in nature and does not involve human subjects research, biological material, or personal data collection. No personal data was collected, stored, or processed beyond the visual detection outputs.

References

1. Z. Zhao and T. Bai, “Financial fraud detection and prediction in listed companies using smote and machine learning algorithms,” *Entropy*, vol. 24, no. 8, p. 1157, 2022.
2. E. Ileberi, S. Yanxia, and Z. Wang, “Performance evaluation of machine learning methods for credit card fraud detection using smote and adaboost,” *IEEE Access*, vol. 9, pp. 1–15, 2021.
3. R. Li, Z. Liu, Y. Ma, D. Yang, and S. Sun, “Internet financial fraud detection based on graph learning,” *IEEE Transactions on Computational Social Systems*, vol. 9, no. 1, pp. 1394–1406, 2022.
4. S. Cao, X. Yang, C. Chen, J. Zhou, X. Li, and Y. Qi, “Titant: Online real-time transaction fraud detection in ant financial,” *Proceedings of the VLDB Endowment*, vol. 12, no. 10,
5. pp. 1–12, 2019.
6. B. Fetaji, M. Fetaji, A. Hasan, S. Rexhepi, and G. Armenski, “Fraud-x: An integrated ai, blockchain, and cybersecurity framework with early warning systems for mitigating online financial fraud,” *IEEE Access*, vol. 13, pp. 48 068–48 083, 2025.
7. S.-H. Li, D. C. Yen, W.-H. Lu, and C. Wang, “Identifying the signs of fraudulent accounts using data mining techniques,” *Computers in Human Behavior*, vol. 28, no. 3, p. 1002–1013, May 2012. [Online]. Available: http://dx.doi.org/10.1016/j.chb.2012.01.002
8. A. Abdallah, M. Maarof, and A. Zainal, “Fraud detection system: A survey,” *Journal of Network and Computer Applications*, vol. 68, 04 2016.
9. M. Zareapoor and P. Shamsolmoali, “Application of credit card fraud detection: Based on bagging ensemble classifier,” *Procedia Computer Science*, vol. 48, pp. 679–686, 12 2015.
10. M. Ahmed, A. Mahmood, and M. R. Islam, “A survey of anomaly detection techniques in financial domain,” *Future Generation Computer Systems*, 01 2015.
11. A. Pumsirirat and L. Yan, “Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine,” *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 1, 2018. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2018.090103
12. M. Mohammed, M. Boujelben, and M. Abid, “A novel approach for fraud detection in blockchain-based healthcare networks using machine learning,” *Future Internet*, vol. 15, 07 2023.
13. Y. Kou, C.-T. Lu, S. Sirwongwattana, and Y.-P. Huang, “Survey of fraud detection tech-niques,” vol. 2, pp. 749 – 754 Vol.2, 02 2004.
14. R. Bolton and D. Hand, “Statistical fraud detection: A review,” *Statistical Science*, vol. 17, 08 2002.
15. V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Comput. Surv.*, vol. 41, 07 2009.
16. P. Treleaven, R. Brown, and D. Yang, “Blockchain technology in finance,” *Computer*, vol. 50, pp. 14–17, 01 2017.
17. F. Casino, T. K. Dasaklis, and C. Patsakis, “A systematic literature review of blockchain-based applications: Current status, classification and open issues,” *Telematics and Informatics*, vol. 36, pp. 55–81, 2019. [Online]. Available: https:www.sciencedirect.com/science/article/pii/S0736585318306324
18. U. Bodkhe, S. Tanwar, K. Parekh, P. Khanpara, S. Tyagi, N. Kumar, and M. Alazab, “Blockchain for industry 4.0: A comprehensive review,” *IEEE Access*, vol. PP, 04 2020.
19. C. Cachin and M. Vukolic, “Blockchain consensus protocols in the wild,” 07 2017.
20. S. Wang, D. Li, Y. Zhang, and J. Chen, “Smart contract-based product traceability system in the supply chain scenario,” *IEEE Access*, vol. PP, pp. 1–1, 08 2019.
21. N. Kshetri, “Blockchain’s roles in strengthening cybersecurity and protecting privacy,” *Telecommunications Policy*, 09 2017.