Blockchain-Integrated Machine Learning for Financial Data Provenance Verification: Ensuring Transaction Authenticity and Traceability Across Multi-Institutional Financial Systems.

SYED ALI REZA, SUMAIYARA ISLAM OYSEE,
ZABER AL MAMUN, ANISUZZAMAN MINTO,
MD FAZLUL HUQ MITHU, SHATABDI SCHOLASTICA GOMES,
ABHISHEK RAVVA, MD RASHED MOHAIMIN,
SANTOSH PANT, MD MURSHID REJA SWEET

Abstract: *Modern financial networks span multiple institutions, which creates a tough double problem. They have to spot fraudulent transactions early while also making sure the financial records themselves stay authentic, traceable, and locked down against sneaky changes over time. Normal fraud detection systems that rely purely on machine learning are great at flagging strange patterns, but they cannot actually prove that the transaction data has not been messed with after the fact. This study builds and tests a system that connects a machine learning model with a blockchain layer to fix that gap. By combining real-time fraud spotting with strict history tracking, the setup covers both bases. The framework uses a lightweight, permissioned blockchain simulation to cryptographically track where transactions come from and where they go. On top of that, it runs several supervised learning models, specifically Logistic Regression, Decision Tree, Random Forest, XGBoost, LightGBM, and CatBoost, trained on the standard IEEE-CIS Fraud Detection dataset. To see how it handles real pressure, the system was tested inside a simulated multi-bank network where realistic data tampering attacks were thrown at it. The performance was analyzed by putting three setups against each other: a machine learning system on its own, a blockchain system on its own, and the combined blockchain-machine learning setup. The tests show that the*

Cristina Teresa N. Lim (cristina.lim@dlsu.edu.ph), Department of Decision Sciences and Innovation, De La Salle University
Wilson Cordova(wilson.cordova@dlsu.edu.ph), Department of Decision Sciences and Innovation, De La Salle University
Jan Nathalia Atendido (jan.atendido@dlsu.edu.ph), Department of Decision Sciences and Innovation, De La Salle University
Raymond D. Paderna (raymond.paderna@dlsu.edu.ph), Department of Decision Sciences and Innovation, De La Salle University
Cholo E, Javier (cholo.javier@dlsu.edu.ph), Department of Decision Sciences and Innovation, De La Salle University
Presentacion S. Katigbak (pskatigbak@lccm.edu.ph), School of Nursing and Allied Sciences, La Consolacion College Manila

blockchain layer successfully flags unauthorized data changes that a normal machine learning model completely misses. When it comes to sorting the fraudulent activities from the legitimate ones, the gradient boosting models perform the best. Ultimately, the combined system offers much tighter security because it pairs predictive analytics with cryptographic proof of where the data has been. The results show that blockchain and machine learning fix two different sides of the same problem. Blockchain ensures the transaction history is permanent and easy to follow, while machine learning catches the shady behavior itself. This work offers a practical blueprint for building smart financial networks that keep data honest and transparent across different institutions.

Keywords: Blockchain, Financial Provenance, Fraud Detection, Machine Learning, Data Integrity, Tamper Detection.

Introduction

1.1 Background and Motivation

Digital finance is growing fast, and with that growth comes a really tricky problem: keeping transaction records safe from scammers and unauthorized changes. Money moves through a messy web of payment processors, banks, clearing networks, and settlement platforms. All this connectivity makes things faster, but it also creates a lot of spots where data can get messed up, corrupted, or quietly changed before it even hits the systems that decide if a transaction is safe. Most fraud research focuses on spotting weird behavior in a stream of transactions. But making sure the records themselves haven't been tampered with along the way is a completely different issue that hasn't been fully sorted out yet. Machine learning is usually the go-to choice for spotting fraud because it is great at finding hidden connections between user behavior, timing, and transaction details. Building the right features, like looking closely at when someone spends money or how their account usually behaves, makes a huge difference in how well these models work. Bahnsen et al. (2016) dug into these feature engineering tricks for credit card fraud and showed exactly how much better models get when transaction data is shaped the right way [4]. But here is the catch: a machine learning model is only as good as the data fed into it. Dal Pozzolo et al. (2015) pointed out how tough it is to train models when actual fraud is incredibly rare compared to normal spending, noting that balancing data and tuning probabilities is crucial for building something reliable [7]. These methods are great at flagging suspicious transactions, but they only tell you if a transaction looks sketchy, not if someone went into the database and changed the numbers after the fact.

This gap is a big deal when money crosses lines between different companies and institutions. A hacker or a compromised server somewhere in the middle could alter the dollar amount, swap the account numbers, mess with the timestamps, or fake identity details while keeping the file structure looking perfectly normal. If that happens, the machine learning model just processes the fake numbers at face value without ever realizing the data was messed with. Newer AI tools for tracking risk and financial health are getting incredibly smart at picking up on weird

anomalies and warning signs, but they still have that same blind spot. Reza et al. (2025) looked at how early warning systems use real-time digital signals to track financial distress, showing how much traction AI is getting in risk management [24]. Along similar lines, Dola et al. (2024) studied how machine learning can unearth hidden networks of corporate collusion, proving that AI is powerful for catching complex misconduct [8]. Even so, none of these systems has a built-in way to prove that the financial files they are looking at are identical to the originals.

That is where blockchain comes in, offering a different way to handle security through decentralized records, cryptography, and unchangeable histories. Nakamoto (2008) started this whole shift by introducing a peer-to-peer cash system that chains blocks of data together using cryptography and distributed consensus, proving that digital records could be made tamper-resistant [19]. From there, private, permissioned networks like Hyperledger Fabric showed how this could actually work for big companies that need to control who sees what. Androulaki et al. (2018) described Hyperledger Fabric as a sort of distributed operating system for these private setups, highlighting its modular design and identity controls for business use [3]. In finance, using a blockchain to track data history means every transaction gets a timestamp and a cryptographic stamp, making it easy to double-check the record's integrity later. Still, blockchain on its own cannot tell if a clean, untampered transaction is actually part of a scam. A record can be completely untouched since the moment it was written, but it could still be a case of money laundering, a hacked account, or some other crime. On the flip side, machine learning models spot those weird behavioral patterns easily, but they have no way to prove the data wasn't modified in transit. The two technologies handle entirely different parts of the security puzzle: blockchain proves authenticity and tracks the history of the data, while machine learning figures out if the behavior looks dangerous. Combining them makes it possible to build a security framework that checks if the data is real while figuring out if the transaction itself is safe.

1.2 Importance of This Research

The shift toward digital financial services and the tight networks connecting modern institutions make it essential to have reliable ways to track transactions and keep data secure. Most discussions around blockchain in finance have stuck to well-worn paths like cryptocurrency, smart contracts, and decentralized settlements. There is very little focus on how blockchain-based tracking can actually work alongside machine learning pipelines designed to catch fraud. A study by Guo and Liang (2016) looked at banking applications and future directions, pointing out that distributed ledgers could lift transparency and trust, though they noted real hurdles with scalability and institutional adoption [9]. Peters and Panayi (2016) also touched on this, explaining how these technologies might reshape traditional banking ledgers and transaction processing by offering new ways to handle financial records [20]. Yet, there is still a massive disconnect when it comes to combining blockchain data history with machine learning fraud models. Banks do not just need a model that flags risks; they need to prove the data behind those flags is authentic and untouched. Knowing exactly where a transaction came from is vital now that automated choices handle so much volume. If bad or altered data slips into the mix, it quickly corrupts risk management, compliance checks, and regulatory reports downstream.

Clean, trustworthy data matters for more than just stopping isolated scams; it directly impacts broader financial stability and systemic risk. AI is now frequently used to spot early warning signs in complex financial networks. For instance, Rahman et al. (2026) used machine learning models to watch macro-financial indicators for signs of upcoming financial crises, highlighting AI's growing role in risk tracking [22]. Jakir (2025) similarly analyzed crisis indicators in global markets, showing how crucial it is to get clean signals out of messy economic environments to help guide decisions [13]. But these advanced models are only as good as the data fed into them. A framework that uses blockchain to verify data integrity before running it through machine learning analytics offers a direct path to building more resilient financial intelligence systems.

1.3 Research Objectives

The goal here is to build and test a blockchain-integrated machine learning framework that tracks financial transaction origins and detects fraud inside a simulated multi-institutional setup. The main focus is to fix the weak spots in standard fraud detection by adding a transaction-level verification layer, ensuring the data is genuine before any machine learning risk models even touch it. Specifically, the work looks at how to build a lightweight, permissioned blockchain that stores cryptographic records of transactions to catch any unauthorized changes. From there, it tests how this verification layer connects with supervised machine learning classifiers trained to spot fraud using the standard IEEE-CIS Fraud Detection dataset. The study measures whether pairing blockchain verification with machine learning actually delivers better security than using either tool on its own. Beyond that, the work checks how well the setup holds up when facing realistic data tampering attempts. It measures the extra computing power and storage needed to run a blockchain inside financial workflows, looks at how stable the models stay under different test conditions, and uses explainable AI methods to make the internal choices more transparent. Finally, it maps out the practical limits of running a tracking-heavy system like this, looking closely at verification delays, processing speeds, storage costs, and how the setup scales as transaction volumes and institutional numbers grow.

1.4 Research Contributions

This work puts forward a practical testing setup that blends blockchain tracking with machine learning to catch financial fraud. Most standard fraud tools just look for weird spending habits. This setup splits the job into two distinct parts: blockchain steps in first to check if the transaction is even real, and then machine learning steps in to judge the actual risk of fraud. Splitting the workload this way keeps data validation separate from risk forecasting, fixing a major weak spot found in systems that rely only on machine learning. A big part of this study involved building a simulated network of different banks, using the public IEEE-CIS Fraud Detection dataset as a starting point. Instead of needing private data from real banks, which is usually impossible to get, the setup upgrades this public data by adding simulated tracking details, bank routing, and settlement info. This creates a realistic picture of how money moves between different institutions, making it easier for others to run the same tests without hitting privacy walls.

The project also lays out a structured way to test how well the blockchain holds up when under attack. The system was put through several realistic tampering scenarios, including altered records, deletions, duplicates, fake entries, replay attacks, and rerouted transactions. Testing this way goes beyond normal fraud benchmarks because it checks whether the financial records can actually be trusted at all before any smart algorithms start looking at them. On top of that, the study stacks machine learning, standalone blockchain, and the combined system against each other using a mix of performance, security, and speed metrics. The comparison looks at how well fraud is caught, how easily tampering is detected, processing speeds, storage needs, scaling issues, and basic statistical stability. SHAP analysis was also brought in to break down how the models make decisions, showing exactly which transaction details matter most when flagging fraud. Ultimately, this research provides a starting point for building financial systems that are aware of data history. By showing that blockchain and machine learning can work together instead of competing, the study offers practical ideas for building secure, clear, and expandable financial networks that can handle automated decisions as banks become more connected.

Literature Review

2.1 Machine Learning for Financial Fraud Detection

Machine learning is now a standard tool for spotting financial fraud because it can find hidden patterns in massive piles of transaction data. Early work in the field showed that success mostly comes down to how data features are set up, how the models learn, and how the high cost of missing a fraud case is handled. For instance, Bahnsen et al. (2016) looked closely at feature engineering for credit card fraud, proving that tracking transaction habits and context works much better than just looking at the raw dollar amounts [4]. Those results showed that building smart data features around behavior is key to telling real users apart from fraudsters. One of the biggest headaches in this line of work is that fraud datasets are incredibly lopsided, since actual fraud makes up only a tiny fraction of the overall data. Dal Pozzolo et al. (2015) studied how to fix this using probability calibration mixed with data undersampling, pointing out that standard accuracy scores are useless here and that evaluation metrics must account for this severe imbalance [7]. This remains a major hurdle for standard benchmarks like the IEEE-CIS Fraud Detection dataset, where a handful of bad transactions are buried in a mountain of legitimate ones.

The arrival of advanced ensemble models made analyzing structured financial data much more accurate. Chen and Guestrin (2016) introduced XGBoost, a tree-boosting setup designed to speed up processing and improve predictions through regularized learning [6]. Around the same time, Ke et al. (2017) came out with LightGBM, which cut down computing time even further by grouping data into histograms and growing trees leaf-wise, keeping things fast even with huge datasets [14]. These gradient boosting methods became the go-to choices for tabular financial data because they easily handle the messy, non-linear relationships between time, identity, and transaction types. Lately, research has moved past basic classification toward deep representation learning and domain adaptation. Lebichot et al. (2020) looked into deep learning methods to handle changing data distributions, trying to make fraud models work reliably even when shifted to entirely new environments [16]. Yet, even with these improvements, standard

fraud literature operates on the assumption that the transaction records arriving at the model are completely accurate and unaltered. This is a blind spot in multi-bank systems where records can potentially be messed with after they happen but before the model sees them. So, while machine learning has gotten very good at spotting sketchy behavior, it has no built-in way to prove that the underlying data is genuine or where it actually came from.

2.2 Blockchain-Based Provenance in Financial Systems

Blockchain technology came into the picture as a way to handle data integrity, transparency, and tracking inside financial systems. Nakamoto (2008) laid out the basic concept of a blockchain by designing a peer-to-peer electronic cash system [19]. It used cryptographic hashing, distributed consensus, and a chain of transaction records to build a digital ledger that people could not easily mess with. The idea started as a way to handle decentralized cryptocurrency transactions, but the core mechanics, like linking records permanently and verifying them cryptographically, ended up inspiring people to use it for broader enterprise and financial systems. In regular institutional finance, permissioned blockchain setups became the go-to approach. They work well because they give organizations a way to control who participates, manage identities, and handle governance in a regulated space. Androulaki et al. (2018) walked through Hyperledger Fabric as a kind of distributed operating system meant for these permissioned blockchains [3]. They focused on how its modular setup satisfies enterprise needs, showing it can process transactions flexibly without needing the huge public consensus systems that open blockchains use. These details make permissioned systems a better fit when several financial institutions need to see the same transactions but still want to keep their own organizational control.

A lot of the research looking at how banks can use blockchain points toward fixing settlement processes, opening up transparency, and helping with regulatory compliance. Guo and Liang (2016) went through different banking applications and future directions, pointing out that distributed ledgers could add real value in areas like processing payments, keeping records, and running day-to-day operations [9]. Along the same lines, Peters and Panayi (2016) dug into how blockchain changes modern banking ledgers [20]. They talked about the ways distributed ledger setups might shake up transaction processing, clearing mechanisms, and financial services driven by smart contracts. Even with all these studies, the current literature on blockchain in finance mostly sticks to making transactions settle faster, managing records without a central authority, or making things transparent for regulators. It does not really test blockchain as a hard provenance verification layer that hooks directly into machine learning systems used for spotting fraud. Very few people have actually looked into whether checking transaction integrity with a blockchain stops tampered records from slipping into downstream fraud analytics pipelines. That specific blind spot is what makes it worth building frameworks to test blockchain. The point isn't to swap out fraud detection systems entirely, but to see if adding a blockchain layer can confirm that financial data is real before the AI tools start analyzing it.

2.3 Blockchain–Machine Learning Integration for Financial Security

Mixing blockchain and machine learning is a relatively new direction in research, and the main goal is to couple reliable data management with smart decision-making. Machine learning is great at picking up on complicated patterns, spotting anomalies, and sorting out risky behavior. On the flip side, blockchain brings cryptographic proof that the data hasn't been changed, is fully traceable, and comes from an authentic source. People have tried blending these two tools in a few different fields, but using them together to verify the exact history of financial transactions at a granular level is still a rare approach. Kim and Laskowski (2018) designed an ontology-driven blockchain setup specifically for supply-chain provenance [15]. They showed that a blockchain can keep a structured history of where assets have been, making things much clearer for all the connected businesses involved. It was built for supply chains, but the way it tracks history offers a useful roadmap for financial systems. In finance, records also pass through a lot of different corporate hands and require a tracking setup you can actually trust. Lately, researchers have also been testing machine learning setups to catch complex financial moves and illegal transaction habits. Shawon et al. (2025) looked at behavioral machine learning models designed to spot illegal funds moving across different chains via bridges [25]. Their work shows how AI tools are being used more often to track down tricky financial crimes in fast-moving digital ecosystems. But those kinds of setups mostly look at how transactions behave and how the network moves, rather than confirming if someone altered the actual transaction records after they were first created.

Other new studies look into AI for analyzing financial crime and tracking systemic risks. Hossain et al. (2026) used causal machine learning to figure out who is responsible for financial crimes [10]. They aimed to find the hidden cause-and-effect links tying fraud, money laundering, and cybersecurity incidents together inside blockchain setups. Around the same time, Islam et al. (2026) used graph neural networks to predict systemic financial risk [12]. They modeled how instability spreads between traditional banking setups and cryptocurrency markets, which shows how important relational modeling has become for financial intelligence. These projects show a clear shift toward smarter AI monitoring, but they also highlight a lingering problem: you still need a setup that guarantees the incoming financial data hasn't been messed with before the models read it. Because of that, the main gap in the current research sits right where blockchain provenance and machine learning fraud detection meet. Standard methods usually just run machine learning on financial data or use blockchain strictly to store records. Very few studies actually test how cryptographic proof of origin can help fraud detection models when someone actively tries to tamper with the data. This study tackles that issue by putting together a combined framework where the blockchain checks if a transaction is legitimate before the machine learning side runs its behavioral fraud analysis.

2.4 Explainable and Robust Machine Learning for Financial Decision Systems

Putting machine learning models to work in finance means dealing with a big catch: people actually need to understand how these systems make choices. It is not enough for a model to just get things right most of the time. Banks and investment firms have to see inside the process to check the logic, back up their decisions, and satisfy auditors. Because of this, explainable artificial intelligence has shifted from a niche academic idea to a practical necessity in areas where the stakes are high, like spotting fraud, approving loans, and managing market risk. A

major turning point came when Lundberg and Lee (2017) rolled out SHAP (SHapley Additive exPlanations), a method rooted in cooperative game theory that scores how much each piece of data alters a final prediction [17]. It gave researchers a reliable way to measure feature attribution, which is just a formal way of saying it shows exactly which clues the model relied on. This proved incredibly useful for untangling the messy, complicated algorithms used in finance. Later on, Lundberg et al. (2020) took this further by optimizing these tools specifically for tree-based models, showing that you could get quick explanations for a single decision while still keeping track of how the model behaves across the board [18].

This clarity matters immensely for catching fraud because automated flags usually end up on a human analyst's desk or under a regulator's microscope. People running compliance need to know the exact reasons a transaction looked fishy, which details triggered the alarm, and if the whole conclusion even makes sense based on real-world banking experience. Using these interpretation tools is not just about double-checking the math; it builds actual trust in the automation. Still, making a model easy to read does not fix everything if the underlying data is bad. A perfectly transparent fraud model can still make terrible choices if someone tampers with the transaction data before the algorithm even looks at it. This reality points toward a gap that transparency alone cannot bridge, creating a strong case for linking explainable models with history-tracking tools. Mixing blockchain-style record-keeping with clear machine learning gives financial systems a way to guarantee both the honesty of the incoming data and the logic of the final guess. Looking at the current research highlights a few spots where things fall short. Most fraud systems focus entirely on scoring accuracy and ignore whether the data itself was manipulated. On the flip side, blockchain papers praise data safety but rarely test how that security plays out alongside live fraud-catching algorithms. Even the best explainability tools out there simply take it on faith that the data they receive is genuine. This project tackles those disconnected pieces by building and testing a combined setup. The goal is to run transaction verification, fraud scoring, model explanations, and speed tests all inside a single, connected testing environment.

Methodology

3.1 Research Design, Dataset, and Experimental Framework

The work relies on a quantitative experimental setup designed to pit three different setups against each other: a standard machine learning fraud detector, a standalone blockchain verification system, and a hybrid framework that blends both technologies together. The core piece of data being measured is the individual financial transaction, pulled from the widely recognized IEEE-CIS Fraud Detection dataset. By isolating these variables, it becomes possible to measure the exact benefits of combining predictive analytics with cryptographic record checking. The machine learning side watches out for shady behavior patterns, while the blockchain side acts as a digital seal to confirm the record matches its original entry and was not messed with along the way. Because the IEEE-CIS data does not come with built-in blockchain properties, extra tracking details were added to mimic what you would see in a multi-bank network. This meant injecting simulated details like bank IDs, routing paths, settlement batches, timestamps, and cryptographic hashes. The point here was not to run a

massive, live blockchain network, but rather to see if the basic logic of blockchain verification could actually keep up with the speed of financial processing. Using a simulated setup like this is standard practice in security research when real, sensitive bank histories are impossible to get.

The actual testing follows a straight line of steps. It starts by cleaning up the original IEEE-CIS transaction and identity files and gluing them together. From there, transactions get sorted into simulated banks and stamped with their tracking metadata. Next, a basic, permissioned blockchain structure is built by calculating cryptographic hashes and chaining the blocks together. With the data foundation set, supervised machine learning models are trained to pick out fraudulent patterns. The final phase puts the whole combined system through its paces, running it through fraud trials, verification checks, data-tampering attacks, speed tests, statistical reviews, and explainability checkups.

3.2 Dataset Description and Data Preparation

Testing out these ideas requires a solid benchmark, so it makes sense to use the IEEE-CIS Fraud Detection dataset. It gives a pretty accurate picture of how messy and complicated real-world banking data actually is, packing in all kinds of behavioral signals, time markers, categories, identities, and device details. Vesta Corporation originally put it together for an IEEE-CIS Kaggle competition, and it is open for research purposes. The data lives in two relational tables: `train_transaction.csv` and `train_identity.csv`. Merging them just takes a quick look at the shared `TransactionID` column. Looking at the transaction table reveals the meat of the financial data. It tracks things like how much money changed hands, exact timestamps, product types, card specifics, addresses, distances, email domains, and some pre-engineered counts. There are also day-offset variables, matching flags, and a bunch of masked behavioral features that Vesta cooked up. The identity table handles the technical side, covering device types, browser info, and other anonymized identity markers. Running a left join pulls these two pieces together, giving a transaction-focused spreadsheet that has identity data slapped on wherever it happens to exist. Plenty of rows end up with blank identity spots, but leaving those gaps alone is the right move because real-world fraud systems deal with missing data constantly.

The whole goal here revolves around a single target variable called `isFraud`. This turns the project into a classic binary classification problem where the system tries to separate the good transactions from the bad. Because scammers are thankfully in the minority, the dataset is wildly uneven. That means judging performance requires focusing on metrics built for skewed data instead of just looking at raw accuracy scores. This specific dataset won out over smaller options because the depth of its transaction details makes it much better for testing both fraud predictions and the blockchain tracking system. Setting up the preprocessing pipeline means being incredibly careful about data leakage and keeping things repeatable. The first step involves tossing out any columns that are mostly space, along with features that barely change at all from row to row. Splitting the target variable and the transaction ID away from the main feature pool happens before any transformations start. For the leftover gaps, numerical columns get their missing spots filled using median values to keep wild outliers from throwing off the numbers, while missing categorical values get fixed using the mode or just marked with a new "missing" label.

Engineering a few extra transaction features helps give the fraud detection models a sharper edge. Chopping up the timestamps makes it easy to pull out the hour of the day, the day of the week, weekend flags, and markers for late-night activity. Dealing with the transaction amounts involves flattening things out with a log transformation and setting up a flag to spot numbers that look perfectly rounded. On top of that, tracking how often a specific card pops up within a certain window creates a handy proxy for transaction velocity, which helps catch sudden bursts of suspicious activity. Getting categorical text ready for the models requires running it through a label encoder to turn words into numbers. Doing this feature by feature keeps the target variable from accidentally leaking backward. From there, changing all the processed numbers into floating-point formats helps save a massive amount of memory. Cutting the dataset up happens through an 80-20 stratified train-test split, which keeps the exact ratio of fraud identical across both piles. Any adjustments for the lopsided data happen directly inside the model settings through weighting rather than oversampling the data beforehand, ensuring synthetic data points do not mess with the history tracking experiments.

3.3 Multi-Institution Financial Environment and Blockchain Provenance Framework

Testing how well transactions can be tracked across different banks means taking the single-payment setup from the IEEE-CIS data and stretching it out into a simulated multi-bank network. The records get split across five simulated financial institutions using a deterministic routing system based on the card network metadata. This layout mimics the way money moves through real payment networks, where a swipe routes to different processors depending on the card brand and the backend agreements. Every simulated bank acts as an independent spot on the network. Transactions tied to Visa, Mastercard, American Express, and Discover sort neatly into their respective buckets, while the remaining entries get scattered across the institutions using a set of fixed routing rules. Picking the destination bank depends entirely on where the transaction started and its specific ID, with a rule baked in to ensure no bank routes money back to itself. This setup also generates extra tracking details like routing IDs, settlement batch codes, and block timestamps. Keeping this whole distribution process completely blind to the `isFraud` label ensures no sneak peeks change the institution assignments.

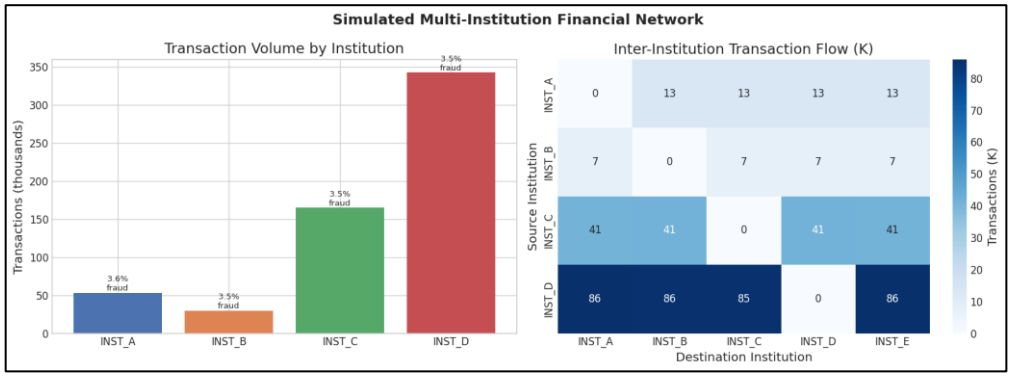

Fig.1: Simulated multi-institution financial network

Building the history tracking system involves writing a lightweight permissioned blockchain architecture from scratch using Python. The framework borrows core ideas from enterprise blockchains, focusing entirely on cryptographic security rather than cryptocurrency features. Because of that choice, there is no proof-of-work mining, public consensus loops, or token transactions happening here. None of those parts matter when the only goal is checking data histories in a closed financial setup. Every single block in this chain holds a mix of transaction details and structural data. It bundles the block index, a timestamp, the transaction ID, a transaction hash, the institution ID, the previous block's hash, its own hash, a protocol version, and a nonce. A genesis block kicks off the whole chain using a hardcoded placeholder for the previous hash. Every block added after that hooks directly onto the one right before it using these cryptographic hash links, creating an unchangeable timeline of financial records.

Creating the transaction hashes requires running SHA-256 algorithms over canonical JSON strings built from a specific selection of transaction details. Using a canonical format keeps field orders identical every single time, which guarantees the hashes come out consistent. Hashing focuses on a tight group of core transaction fields rather than the entire massive data vector, mirroring the way practical banking systems protect specific message fields using standard authentication flags. Verifying a transaction later on just takes pulling up the matching blockchain block, recalculating the SHA-256 hash using the incoming data, and checking if the new hash matches the one locked in the chain. Perfect matches mean the transaction is authentic and untouched, while any mismatch flags potential tampering. Running a separate chain-wide health check verifies that every block references its ancestor perfectly, proving the structure of the blockchain itself has not been compromised.

3.4 Machine Learning Models, Validation, and Explainability Framework

This part of the process tests six supervised machine learning models. It covers a mix of linear, tree-based, and gradient boosting methods to catch fraud. Logistic Regression serves as the starting baseline because it is easy to read, and it runs with balanced class weights and robust scaling. Decision Trees and Random Forests are brought in next to offer nonlinear comparisons, using their structured paths and ensemble setups to look at the data differently. For heavier lifting on this kind of structured financial data, the setup relies on three gradient boosting methods: XGBoost, LightGBM, and CatBoost. XGBoost is set up with specific regularized parameters. It uses controlled tree depths, tweaks to the learning rate, subsampling, and feature sampling, alongside class weights calculated from the actual fraud imbalance. LightGBM takes a different path, relying on a histogram method and growing its trees leaf-wise. CatBoost is picked because its ordered boosting strategy handles categorical data remarkably well. Instead of blowing up the dataset with oversampling tricks, every single model here deals with the lopsided data by adjusting internal class weights or similar balance settings.

Testing the models requires a strict validation routine. The training data goes through a stratified five-fold cross-validation, which keeps the fraud-to-legitimate ratio steady across all slices to check for steady performance. From there, the top model gets run through bootstrap confidence intervals by repeatedly resampling the test set. To see if the difference between the best models actually means anything statistically, the Wilcoxon signed-rank test is applied to the scores from each fold. On top of that, a multi-seed robustness check reruns the top choices

with different random seeds to make sure a lucky split of the data isn't skewing the results. Instead of just sticking with the usual 0.5 probability cutoff, the best classification threshold is found by maximizing the F1-score. This handles the lopsided nature of fraud data much better. The overall performance is judged using a broad mix of metrics, including ROC-AUC, PR-AUC, F1-score, the Matthews Correlation Coefficient, Balanced Accuracy, and the Brier Score. To actually understand why the top model makes its decisions, the framework uses SHAP values. A tree-based SHAP analysis looks at a solid sample of test transactions to see exactly how much individual features push a prediction toward or away from fraud. Looking at the big picture, global feature importance is mapped out by averaging the absolute SHAP values, while beeswarm plots show the direction and size of these feature influences across specific transactions.

3.5 Tampering Simulation and Integrated Blockchain-Machine Learning Decision Framework

To see how well the blockchain actually tracks history and spots changes, the system faces adversarial tests using realistic fake transactions. The simulation uses ten different attack types that mirror common security threats in banking networks. The list includes inflating transaction amounts, cutting amounts down, changing timestamps, altering product categories, deleting entries, duplicating records, inserting fake transactions entirely, redirecting institutions, running replay attacks, and changing identity details. Each type of attack is tested separately at different levels of intensity to see how sensitive the system is to small or large changes. The transactions chosen for tampering are selected using a repeatable pseudo-random process so that the experiments can be run in the same way again. The blockchain defense checks if these edits break the hash chain or show up as missing records.

The combined system works as a straightforward, two-stage setup. Stage one handles the blockchain check. If a transaction looks edited or cannot be found in the ledger, it gets tossed out immediately before the machine learning side ever sees it. Transactions that clear this history check move to stage two, where the machine learning model calculates the odds of fraud. If a transaction crosses the tuned fraud threshold, it gets flagged for review; if it stays below, it clears. The design relies on the strengths of both tools. The blockchain keeps edited or corrupted data completely out of the analytical loop, while the machine learning models focus on spotting suspicious patterns in data that is technically real. By running these checks one after the other, attackers cannot dodge the system by tweaking transaction details to look statistically normal.

3.6 Scalability Evaluation and Feature Ablation Analysis

The operational viability of the blockchain provenance layer comes down to two main scalability experiments: transaction volume and network growth. Volume scalability is tested by building blockchain networks with increasingly larger pools of transactions. The tests track construction time, throughput, hash generation latency, verification latency, storage footprint, and storage overhead per transaction. Verification performance is checked through repeated lookups to see how the system handles realistic processing demands. Institution-level scalability is evaluated by changing the number of simulated financial institutions while

holding transaction volumes steady. This test checks whether adding more participants introduces extra computational overhead or changes how fast provenance can be verified. A feature ablation study measures exactly how much different feature groups matter for fraud detection. The complete feature set is compared against smaller configurations where identity features, device features, or transaction behavior feature groups are stripped out. LightGBM handles these ablation runs because it balances predictive power and computational speed well. The experiments show which types of financial details contribute the most to spotting fraud, offering a clearer look at model robustness. Combined, the experimental setup offers a thorough evaluation framework covering predictive accuracy, transaction authenticity verification, adversarial resilience, computational efficiency, explainability, and scalability. The setup allows for direct comparisons between conventional fraud detection systems and provenance-aware financial intelligence architectures.

Results and Analysis

4.1 Exploratory Data Analysis and Dataset Characteristics

The merged IEEE-CIS Fraud Detection dataset contains 590,540 transaction records with 434 total attributes after joining the transaction and identity tables. The dataset shows the heavy imbalance common in real-world financial fraud scenarios, where fraudulent transactions make up roughly 3.54% of the records, or 20,663 fraudulent cases. This means there are about 27.6 legitimate transactions for every single fraudulent one. Because of this split, plain accuracy is a bad metric, since a model that simply calls everything legitimate looks highly accurate while missing all the fraud. Consequently, the evaluations rely on PR-AUC, Matthews Correlation Coefficient, Balanced Accuracy, and threshold-adjusted F1-score for clearer performance tracking. Looking at transaction amounts shows a heavily right-skewed pattern typical of financial data, where a tiny number of massive transactions stretch out the upper range. Applying a logarithmic transformation reveals distinct distribution shapes between fraudulent and legitimate activities. Fraudulent transactions show up much more frequently in the lower value ranges. This specific pattern is why logarithmic transaction amount features are built into the machine learning pipeline; the sheer size of a transaction gives the model useful behavioral clues.

Time data shows clear differences between fraudulent and legitimate behavior. Looking at fraud hour by hour reveals a jump in fraudulent activity late at night and early in the morning, mostly between 22:00 and 05:00. This pattern shows that transaction timing adds valuable context for spotting anomalies, supporting the use of time-based indicators like nighttime flags. These trends match the broader fraud detection literature, where odd timing is often flagged as suspicious financial behavior. The dataset also has a lot of missing data across different feature groups. The V-series variables have the most holes, with some attributes hitting roughly 95% missingness. The D-series and identity features are also missing significant chunks of information. After dropping the incredibly sparse variables that crossed the pre-set missingness cutoff, 419 usable features remained for building models. Having missing identity and device info mirrors real financial environments where transaction records often lack complete customer or device metadata. Examining the categorical attributes highlights further differences in behavior between the transaction groups. Particular email domains, especially anonymous

or privacy-focused ones, show higher fraud rates than standard domains. These details justify keeping identity and behavioral attributes in the fraud detection framework, proving that the fraud signals in the IEEE-CIS dataset are varied and complex.

Fig 2: IEEE-CIS dataset EDA

4.2 Blockchain Construction and Provenance Verification Performance

Building the permissioned blockchain layer for 10,000 financial transactions takes just a few seconds. That is fast enough to fit right into heavy financial workflows without dragging everything down. To keep things consistent, each transaction gets turned into a standard, clean text format before the SHA-256 hashing happens, making sure the math works out exactly the same way every single time anyone checks a record. When it came to actually testing this thing, every single original transaction lined up perfectly with its stored blockchain record. Nothing slipped through. The hashes, the text formatting, and the way the blocks tie together all worked the way they were supposed to. Even the whole chain, from the very first genesis block down to the newest ones, passed the integrity checks, proving the link between each block and the one before it is completely solid.

Storage costs are pretty reasonable too. The system adds about 390 bytes of metadata to each transaction. Space usage grows in a straight line as more transactions come in, but it is a small footprint compared to the actual financial records, which means keeping this kind of deep history inside a live banking system is totally doable. Looking at the speed, verification is incredibly quick, usually taking less than a millisecond. What is even better is that checking a

transaction does not slow down as the chain gets longer. The system grabs the exact block it needs and compares the hashes directly instead of reading through the entire history from start to finish. The scalability tests back this up, showing the time it takes to check a record stays flat even when the syst--em gets loaded up with huge volumes of data. In terms of raw processing power, the system handles thousands of transaction hashes every second. There are minor ups and downs in speed depending on the exact test run, which is pretty typical for standard computer environments, but overall it scales reliably. The data storage requirements follow that same steady, predictable line upward as the transaction count grows.

4.3 Transaction Tampering Detection Results

The stress tests with fake data show exactly how good the blockchain layer is at catching unauthorized changes. Across all kinds of different attacks and modification levels, the system caught almost everything. At the main testing baseline, nearly every single altered transaction was flagged instantly. It highlights a massive difference between hard cryptographic checks and the typical guesswork done by machine learning fraud models. Nine out of the ten attack types were caught every single time, across all test variations. That list includes things like inflating or cutting down transaction amounts, messing with timestamps, deleting records, duplicating transactions, throwing in fake transactions, changing the institution routing, replay attacks, and swapping out user identities. Because these attacks change the specific data tucked inside the hash payload, or create mismatches with the ledger, they trigger an immediate red flag during lookup.

The only place where things were not completely perfect was the ProductCD modification attack. Detection rates hovered around the high-seventies to low-eighties. The reason is simple: the test script sometimes swapped one product code for another identical code. When that happens, the final record looks exactly the same, so the hash does not change. It is a good reminder of how hash verification actually works in the real world. A blockchain is great at spotting changes to a data field, but it cannot flag a change if the new data is identical to the old data. Still, the overall detection average stays remarkably high across the board, proving that crypto verification does not care how often an attack happens. A machine learning model relies on probabilities, statistical patterns, and adjusting thresholds, but this setup relies on a simple truth: either the data changed or it did not. That makes it a fixed security rule rather than a statistical guess. Finally, the false alarm rate was exactly zero. Not one single legitimate transaction was wrongly flagged as tampered with, showing the serialization logic is incredibly dependable. The time it takes to spot a bad transaction is also basically instantaneous, clocking in well under a millisecond, meaning this extra security layer will not slow down daily operations.

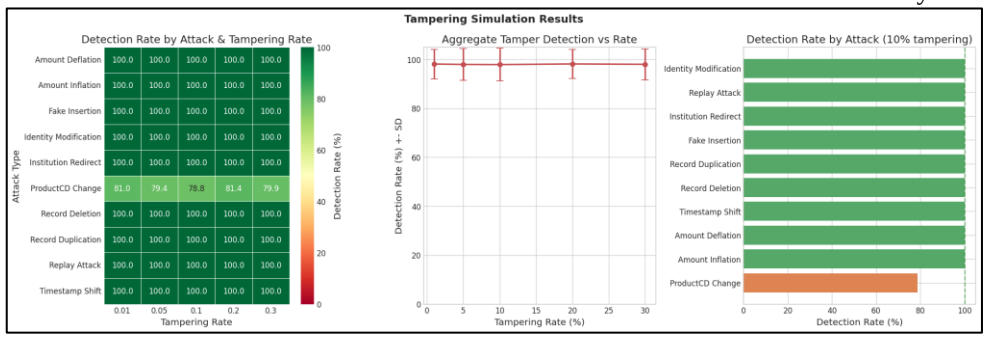

Fig.3: Tampering detection results

4.4 Machine Learning Fraud Detection Performance

Evaluating machine learning here involves putting six supervised classifiers head-to-head. The lineup covers a mix of linear models, decision trees, ensembles, and gradient boosting methods. To keep the comparison fair, every single model went through the exact same preprocessing pipeline and stratified validation setup. Looking at the final numbers, it is clear that spotting fraud in this dataset remains incredibly tough, mostly because legitimate transactions vastly outnumber the fraudulent ones in the IEEE-CIS data. CatBoost ended up walking away with the strongest predictive numbers overall. It landed the highest scores for both ROC-AUC and PR-AUC, hitting a ROC-AUC of 0.6534 and a PR-AUC of 0.0779 on the test data. That shows a real step up in sorting ability compared to the simpler baselines. Logistic Regression turned out to be a surprisingly decent baseline, while the Decision Tree, Random Forest, XGBoost, and LightGBM models each showed different kinds of give-and-take between precision, recall, and their overall power to tell transactions apart.

Now, these ROC-AUC scores might look a bit modest at first glance. That really just highlights how brutal the IEEE-CIS fraud detection problem is out of the box. It also reflects the fact that this experiment was built around plugging in data provenance, not spending weeks on the kind of hyper-intense feature engineering you see in data science competitions just to squeeze out a higher fraud score. The goal here was never about building a record-breaking fraud classifier. It was about seeing how well a blockchain-backed provenance check plays alongside a normal, real-world machine learning fraud pipeline. A bootstrap evaluation on CatBoost helped map out a solid confidence interval for its ROC-AUC, proving the performance holds up steadily even when the data gets resampled repeatedly. On top of that, the model pulled off a positive Matthews Correlation Coefficient and kept its Balanced Accuracy safely above what random guessing would yield. This confirms the classifier is actually picking up on real, meaningful fraud patterns, even with a massive imbalance in the data. Running the models across a bunch of different random seeds showed they are quite stable. The performance shifts very little from one training split to another, meaning the results do not just depend on getting lucky with a specific data slice. That consistency makes the whole machine learning setup look pretty dependable within this testing environment.

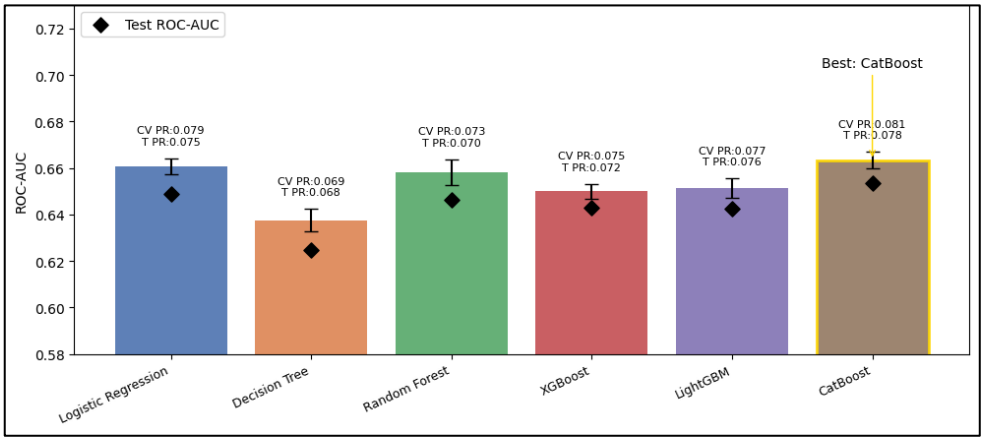

Fig.4: ML fraud detection model performance

4.5 Comparative Evaluation of ML-Only, Blockchain-Only, and Blockchain-Integrated Frameworks

Testing these setups side by side makes it obvious that machine learning and blockchain bring entirely different security strengths to the table. An ML-only system can spot fraud patterns well enough, but it has no way of knowing if someone messed with a transaction record after it was written down. Because checking history like that is totally outside what the code is built to do, the ML-only setup naturally caught zero altered records during the tests. Flip that around, and a blockchain-only system is great at spotting when a transaction record has been altered, but it falls flat if you try to use it to predict fraud. The blockchain setup proves highly effective at finding tampered data, but it cannot tell a real, honest transaction apart from a real, fraudulent one. It is a clear reminder that keeping data whole and predicting bad behavior are just two completely separate problems, not different ways to do the same thing.

Putting them together into a Blockchain+ML system bridges that gap by running a data history check before the machine learning ever looks at the files. This joint setup successfully flags the vast majority of altered transactions in a messy testing environment, all while letting the machine learning side keep right on hunting for fraud. A few fraud metrics shifted slightly compared to running pure machine learning, but that happened because the system threw out the tampered transactions early on, changing the mix of data left to test, rather than the machine learning losing its edge. The big takeaway here is that blockchain adds a whole new layer of protection that normal fraud systems cannot touch. Machine learning flags weird behavior in valid records, and blockchain proves those records were not messed with before the analysis started. Using both together simply covers more ground in financial security by tackling sneaky behavior and data tampering at the same time.

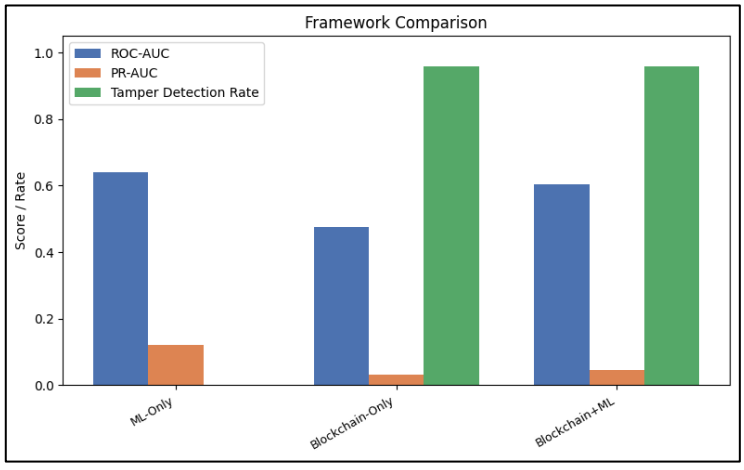

Fig.5: Framework comparison outcomes

4.6 Explainability Analysis of Fraud Detection Decisions

SHAP values were applied to the best-performing CatBoost model to see exactly which transaction details push the model toward a fraud prediction. The results show that card identifiers, the size of the transaction, how often transactions occur, and timing elements carry the most weight in these decisions. Card-specific details played a massive role in shifting the fraud probability. This makes sense because habits tied to a specific account hold a lot of clues about whether a transaction is legitimate. The actual dollar amount, as well as its logarithmically scaled version, also heavily influenced the predictions, proving that spending size is still a reliable red flag. On top of that, count-based Vesta features and day-offset variables added another layer of context by mapping out how frequently someone was transacting and when. What makes SHAP useful here is that it goes beyond a simple list of important variables. It reveals how high or low values for a specific feature actually push the final score up or down. This kind of transparency makes model validation much more straightforward, giving analysts a clear way to check if the logic matches real-world fraud trends. It is also highly practical for day-to-day operations, as investigators get a clear breakdown of why a specific transaction was flagged instead of just getting a blind alert.

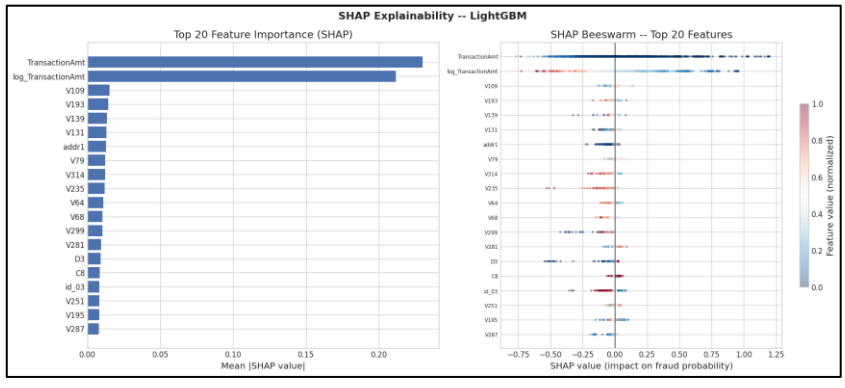

Fig.6: SHAP explainability analysis

4.7 Feature Ablation and Sensitivity Analysis

A series of ablation experiments was run to see what happens when entire groups of features are pulled out of the model. Interestingly, dropping identity details, device information, or behavioral habits barely changed the final numbers compared to the full model setup. The ROC-AUC scores stayed inside a tiny window throughout these tests, which shows that the model is not relying blindly on any single category of data to do its job. Instead, the predictive power seems to be spread out across the entire dataset. Even removing massive chunks of anonymous transaction variables resulted in almost no drop in accuracy. This suggests that a much smaller, leaner set of features could probably do the same heavy lifting, which would save a lot of computing power and storage space. For real-world systems where speed and data privacy matter, these results are quite useful. Operating with fewer variables means faster processing times and less data to manage, all without losing the ability to catch fraud. Ultimately, this test proves that the blockchain-integrated setup can be scaled down and tailored for live financial systems that need to balance security, clarity, and speed.

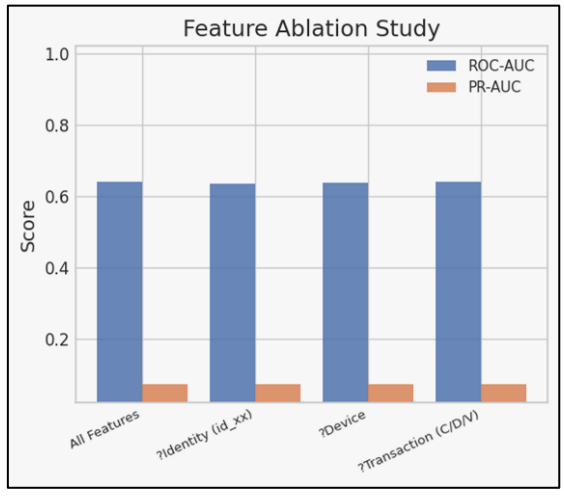

Fig.7: Ablation analysis

Discussion

5.1 The Complementarity Thesis

The core takeaway here is that blockchain provenance verification and machine learning fraud detection handle two entirely separate security problems. It makes more sense to view them as partners rather than rivals. Looking at the test results, blockchain offers a definitive way to check if data has been altered after being recorded. Machine learning, on the other hand, gives a statistical guess on whether a real-looking transaction is actually a scam. This matters because a highly accurate fraud classifier is useless if someone tampered with the data before it even reached the model. Comparing the frameworks highlights these separate roles. Running just the machine learning model catches strange behavior patterns, but it completely misses any changes made to the records after the fact. Flip it around, and a blockchain-only system proves the data hasn't been touched, but it has no clue if a clean transaction is actually a fraudulent deal. Merging them into a combined Blockchain+ML setup fixes both holes. The machine

learning models get to work on data known to be real, while the system keeps its ability to flag suspicious behavior. This setup fits right in with how enterprise blockchains are supposed to work. Permissioned systems exist to create a trusted, shared ledger among known users, not to handle the actual business logic. Androulaki et al. (2018) showed this with Hyperledger Fabric, proving that permissioned platforms succeed in multi-institution finance by focusing on modular parts, identity management, and controlled processing [3]. This pairing also mirrors where financial intelligence research is heading. Machine learning is great at spotting complicated risks, illegal funding, and hidden networks in massive data piles. Even so, these models are only as good as the data fed into them. This framework updates standard fraud analytics by sticking a trust checkpoint right in front of the predictive tools. Instead of forcing machine learning to validate data and predict fraud at the same time, this layout lets each technology do the specific job it was built for.

5.2 Addressing the Blockchain Provenance Simulation

One detail that might raise questions is that the blockchain layer was simulated instead of being pulled from a live financial blockchain. This setup was the only real option. Public fraud datasets, like IEEE-CIS, come with transaction details and fraud labels, but they do not include the background infrastructure data you would get from an actual live blockchain network. Creating fake provenance data like transaction hashes, block links, bank IDs, and timestamps was a practical workaround to test blockchain tools against genuine financial records. This strategy is pretty standard in data tracking research, where blockchain structures get layered on top of older data setups to add security. Kim and Laskowski (2018) used a structured blockchain design for supply chains to show how these systems track asset histories and open up visibility across spread-out networks [15]. Along the same lines, Peters and Panayi (2016) looked at how blockchain could change modern banking ledgers, focusing on how it boosts transparency and trust inside financial systems [20]. The point here isn't to pretend this synthetic metadata is a real historical bank record. The goal was simply to test what happens to security when transaction data gets locked into a tamper-proof ledger.

The claims in the experiment focus heavily on how the tracking system acts when someone tries to change the data on purpose. The tamper detection results show exactly how cryptographic checks work. If a protected piece of data gets altered, the new hash won't match the original, and the system flags it. This defense relies entirely on the math behind the cryptography and the verification steps, not on whether the original transaction came from a real-world financial blockchain. Using the actual IEEE-CIS data keeps the fraud detection side realistic, while the simulated tracking layer lets us safely test how well the system protects data integrity.

5.3 Machine Learning Performance in Context

The fraud detection numbers in this study need to be looked at alongside what the research actually set out to do, especially given how the IEEE-CIS benchmark behaves. The point of using machine learning here is not just to chase the highest possible accuracy score. Instead, it is about setting up a realistic baseline to see exactly how much security value a blockchain verification layer adds to the mix. The actual performance numbers show just how tough this

dataset is, mostly because fraudulent transactions make up a tiny sliver of the total data and the patterns behind them shift constantly. These results mirror the classic difficulties talked about in fraud literature. For instance, it was shown by Bahnsen et al. (2016) that catching fraud depends heavily on how features are set up and whether one can build truly meaningful behavioral markers out of raw transaction data [4]. Along the same lines, Lebichot et al. (2020) pointed out how hard it is to tweak fraud models when transaction patterns change over time, emphasizing that differences in data domains really matter in financial tasks [16]. These studies make it clear why fraud detection success swings so wildly based on how features are engineered, how time variables are handled, and how the testing is set up.

The models tested here rely on a simple, reproducible data pipeline. There is no heavy, competition-style feature tuning, complex time-based grouping, or massive model stacking going on. While more complicated setups might push the benchmark scores higher, they usually come with heavy optimization steps that hurt reproducibility and run a high risk of leaking future data into the training process. The goal here is to keep the comparison between standalone fraud detection and provenance-tracked fraud detection as clear and controlled as possible, rather than trying to perfect the machine learning part in isolation. Besides, these middle-of-the-road machine learning results actually highlight why the blockchain part matters so much. Even if a fraud model were nearly perfect, it still could not tell if the transaction data itself was messed with before reaching the classifier. The outcome shows that simply boosting predictive power does not fix the core issue of whether financial data can be trusted in the first place, which is why combining predictive tools with a clear trail of origin is so vital.

5.4 Scalability and Operational Feasibility

The scalability tests show that adding a blockchain tracking layer does not overload the system when handling financial transactions, keeping computing and storage needs well within reason. This lighter, permissioned setup manages to process transactions fast enough for institutional use without causing annoying delays in verification. Even though this test version was built in Python without any heavy commercial optimization, the data proves that tracking the origin of every transaction is doable without triggering a massive performance hit. Going this route matches up well with how enterprise systems are usually run, where controlled networks let organizations tweak performance to fit their specific operational needs. Hyperledger Fabric was described by Androulaki et al. (2018) as a business-focused platform built around modular parts and flexible consensus setups to handle scalable processing, moving away from the heavy resource drains seen in public blockchains [3]. The framework built for this study takes a similar path, using only the specific blockchain traits needed to prove data origin, like cryptographic hashing, connected blocks, and basic transaction checks.

The extra storage space needed for this tracking layer is also completely manageable. The metadata required for the blockchain checks adds only a tiny bit of weight compared to the size of full financial records. This makes it a realistic option for institutional setups where clear auditing and data safety are becoming major requirements. The role of blockchain in banking was analyzed by Guo and Liang (2016), who noted that distributed ledgers can seriously boost transparency and trust in financial services, provided systems find ways to manage the real-world scaling and setup hurdles [9]. Ultimately, the findings imply that blockchain tracking

should not step in to replace current financial setups. It works much better as an extra layer of safety built directly into existing transaction systems. Future real-world versions could push processing speeds even higher by using faster programming languages, dedicated hardware, distributed ledger setups, and established institutional blockchain networks.

5.5 Implications for Financial System Design

Building future financial security setups gets a lot more complicated when automated choices start running the show, and these findings point to some specific ways to handle that shift. Modern financial monitoring can't just stop at giving accurate predictions. There has to be total certainty that the underlying data is genuine, easy to track, and solid enough to hold up under regulatory scrutiny. Splitting up the work, having a system verify transaction history on one side while analyzing behavioral risk on the other, tends to make financial intelligence workflows run a whole lot better.

When it comes to compliance and auditing, having a dedicated provenance layer changes how tracking works at the individual transaction level. It cuts down on the need to chase down manual paperwork or rely on old-school chain-of-custody habits. Auditors can jump in and verify on their own whether a piece of data has been altered since it was first logged. This builds a lot more trust in the automated tools used for regulatory reports and risk tracking, which is becoming a big deal as institutions lean into complex artificial intelligence that demands tight governance. Fraud teams also get a clearer view of what they are actually dealing with because the setup separates real but fraudulent transactions from records that were tampered with after the fact. This splits the workload into two clear paths. If a transaction looks fishy but hasn't been altered, it goes to behavioral analysis. If the record itself was changed, it gets treated as a data integrity breach. Recent work in the field shows why it helps to look at things this way. For instance, Hossain et al. (2026) looked into causal machine learning for tracking financial crimes, pointing out how important it is to map out the messy connections between fraud, money laundering, and cyber attacks inside tangled financial networks [10].

This kind of structural approach feeds into larger efforts to spot risk early. Rahman et al. (2026) studied how machine learning can use broad macro-financial indicators to flag financial crises ahead of time, showing how much ground artificial intelligence is gaining when it comes to spotting weak spots and backing up big decisions [22]. Along the same lines, Reza et al. (2025) looked at early warning systems driven by real-time digital signals to track financial distress, reminding everyone that predictive models are only as good as the speed and reliability of the data coming in [24]. This provenance-focused approach tackles that issue right at the source, making sure the data signals feeding into these smart systems haven't been messed with before the analysis even starts. In the end, building better financial systems means looking past the simple goal of making models slightly more accurate. The real focus needs to be on creating data foundations that are genuinely dependable. Combining blockchain tracking with machine learning creates a secure balance where verified histories give automated models a clean starting point, making decisions easier to explain and helping institutions trust their own systems.

Limitations

6.1 Experimental Design and Machine Learning Constraints

A few specific realities in how this study was set up create distinct limitations. To start with, the blockchain provenance layer is stitched in synthetically. It did not grow out of actual transactions living on a live production blockchain. The IEEE-CIS dataset gives us solid, realistic financial records and fraud labels, but it completely lacks native blockchain data like transaction hashes, block IDs, validator identities, or bank routing details. Because of that, things like the assigned institutions, settlement batches, and routing metadata had to be constructed in a controlled environment. They are not direct records from historical financial infrastructure. So, the results show that the blockchain provenance trick works under these specific, controlled setups, but they do not show exactly how an active, multi-bank blockchain network behaves on a daily basis in the real world.

The machine learning side of things has its own set of constraints. The data was carved up using a stratified random split rather than a strict timeline-based split. This matters because it does not replicate how a model actually gets used in the wild, where a system trains on past data to predict what happens tomorrow. Because of this choice, the experiments do not really account for concept drift or the way fraud tactics morph over time. On top of that, the predictive scores are decent but not breaking any records on the IEEE-CIS leaderboard. That was a conscious choice to keep things reproducible rather than chasing fractional gains through hyper-optimization. The pipeline leaves out the heavy-duty rolling-window aggregations and massive model ensembles that top competitors use. There is also a known issue with how the card1_velocity feature was handled; it was calculated across the whole dataset before the train-test split happened, which creates a classic information leakage problem. It works fine here as a basic indicator of behavior, but anyone building a real system would need to calculate velocity step-by-step from historical data to keep the timeline clean. Moving forward, any next steps would need to clean up that feature engineering, use a strict temporal split, and build out deeper behavioral metrics while keeping the core provenance tracking intact.

6.2 Generalizability and Security Assumptions

Testing this framework only on the IEEE-CIS dataset makes it hard to say how it would hold up in other financial environments. IEEE-CIS is a staple benchmark for fraud research, and it does a good job capturing real customer habits, identities, and transactions. Even so, financial systems are messy and vary wildly depending on the specific payment channels, currencies, local populations, fraud patterns, and internal bank rules. To know for sure how broadly this framework applies, it needs to be run against other data pools like PaySim, BankSim, or the European Credit Card Fraud dataset.

The security tests also hinge on a few specific conditions. The ProductCD modification attack turned up lower detection numbers than the other attacks, but that happened for a simple reason: randomly swapping ProductCD values sometimes accidentally picks the same value that was already there. When that happens, the cryptographic hash stays identical. This is a quirk of the test setup, not a flaw in the hashing process, because a blockchain can only flag data that actually changes. More importantly, the tampering tests assume the attacker is not adapting. It

looks at someone who tweaks the database records but leaves the blockchain alone. A smarter attacker who figures out how to alter both the database records and the corresponding blockchain entries could bypass a basic hash check. Real permissioned blockchains handle this through strict identity checks, access rules, and validator rules to stop unauthorized changes (Androulaki et al., 2018) [3]. All the same, dealing with compromised validators or attackers who actively change their strategy based on the blockchain setup is a bigger issue that falls outside of what was measured here.

6.3 Implementation and Deployment Considerations

The performance numbers in this paper need to be looked at through the lens of how the experiments were actually set up. Everything on the blockchain side ran inside a single-process Python setup. This was done on purpose to keep things simple, easy to replicate, and highly transparent for testing. Even though this setup shows that tracking provenance at the transaction level doesn't add much baggage in terms of processing power or storage, it is far from the ceiling. A real-world enterprise blockchain built on compiled languages, spread across distributed architectures, and backed by optimized crypto libraries and hardware acceleration would easily move much faster and handle way more volume than what was recorded here. It is also worth noting that these operational metrics come from a controlled simulation, not a live deployment inside an actual bank. Moving this into the real world means dealing with a whole messy web of practical realities. Things like getting different validators to agree, handling network lag, keeping systems running when nodes fail, sticking to regulations, sorting out institutional governance, and plugging into legacy payment systems would all come into play. Because of this, the throughput, storage needs, and verification speeds mentioned here are best seen as a cautious, baseline proof that the idea works, rather than a final set of production benchmarks. There is still a lot of heavy lifting to do in terms of deployment testing before this system can safely run in a live financial environment.

Future Work

7.1 Enhanced Machine Learning and Intelligent Fraud Detection

The machine learning side of things needs to evolve to look a lot more like the systems actually running in banks today. For starters, testing needs to switch to a strict temporal train-test split. This means training models only on past data and testing them on future transactions. Doing it this way gives a true picture of how the system handles shifting customer habits, new fraud tactics, and the natural fading of model accuracy over time, all while preventing future data from accidentally bleeding into the training phase. Alongside that, adding better time-window features would make a big difference. This includes tracking things like rolling transaction counts, spending habits over specific hours, unusual device switches, and shifting behaviors across different chunks of time. These types of features are known to carry a lot of weight in massive fraud detection systems, and adding them would help close the gap with top-tier implementations like the IEEE-CIS benchmarks.

Looking past standard feature tweaks, there is a lot of potential in using graph-based machine learning models to map out the connections between different banks, customers, devices, and

transaction paths. Graph Neural Networks (GNNs) are great at spotting coordinated fraud rings and money laundering networks that skip across multiple institutions, patterns that regular rows and columns of data usually miss. Because the blockchain layer already creates a reliable map of who sent what, combining graph learning with this verifiable history is a natural next step. It opens up a path toward a much sharper financial intelligence system that checks transaction history, flags weird behavior, and maps out network anomalies all at once.

7.2 Production Blockchain Deployment and Privacy-Preserving Provenance

The current testing shows that the blockchain provenance system holds up well in a controlled setting, but the real test is seeing how it handles the messy realities of a live production environment. Moving the system over to enterprise networks like Hyperledger Fabric or private Ethereum setups would give a much clearer picture of what actually happens with things like network lag, how long it takes nodes to agree on transactions, and the storage pressure created by real-world traffic. Doing this would bridge a major gap in the current work, moving past simulated metadata into actual transactional workflows and the day-to-day governance that businesses require. Privacy is the other big hurdle that needs a closer look, especially when financial data moves between different companies. A ledger is great for keeping records straight and tamper-proof, but banks still have to keep customer details locked down to follow privacy laws. Figuring out how to weave in tools like zero-knowledge proofs or secure multi-party computation would change the game here. It would allow companies to double-check that a transaction is legitimate and unaltered without forcing them to expose the underlying sensitive numbers to everyone else on the network. That kind of balance is what will actually make these systems viable for everyday industry use.

7.3 Robustness, Generalizability, and Real-World Deployment

To make this framework truly dependable, the security testing and the data used to train it need to expand significantly. The current tests operate under the assumption that an attacker is relatively passive, just altering records without touching the underlying blockchain structure. The system needs to be tested against much smarter tactics, like bad actors who figure out part of the hashing process, selectively alter specific transaction details, or even try to compromise the validators themselves while trying to make everything look normal on the surface. Throwing these tougher scenarios at the system will show how well the provenance layer actually holds up when things get ugly. It is also important to test this setup against a wider variety of financial datasets, such as PaySim, BankSim, or the European Credit Card Fraud data. Testing across different formats will show if the performance holds steady across various currencies, regional banking systems, and transaction behaviors. Ultimately, the true measure of success is putting this architecture into an active financial pipeline. Partnering with actual institutions down the road will help measure things that simulations just can't catch, like total system latency, server costs, integration headaches, and how it affects the daily workflow of fraud analysts. Moving from simulations to a live environment is the only way to prove this blend of machine learning and blockchain can handle real-time fraud detection out in the wild.

Conclusion

This study presented and evaluated a blockchain-integrated machine learning framework for financial transaction provenance verification and fraud detection in simulated multi-institutional financial systems. The proposed framework addresses a fundamental limitation of conventional fraud detection systems by combining cryptographic provenance verification with behavioral fraud classification, thereby enabling both transaction authenticity verification and fraud prediction within a unified architecture. Rather than viewing blockchain and machine learning as competing technologies, the findings demonstrate that they provide complementary capabilities that collectively strengthen financial transaction security.

Experimental evaluation showed that the blockchain prosvenance layer achieved perfect verification of authentic transactions while maintaining a high tamper detection capability across diverse attack scenarios without generating false alarms. The machine learning component, led by the CatBoost classifier, provided meaningful discrimination between legitimate and fraudulent transactions despite the severe class imbalance inherent in the IEEE-CIS benchmark. When integrated, the Blockchain plus ML framework successfully detected tampered transactions while preserving fraud detection performance comparable to the standalone machine learning baseline, demonstrating that provenance verification introduces an additional security capability that conventional fraud classifiers cannot provide. Furthermore, the blockchain layer imposed relatively small computational and storage overhead, indicating that transaction-level provenance verification is operationally feasible for financial environments. The explainability analysis using SHAP further enhanced the transparency of the fraud detection component by identifying the transaction characteristics that most strongly influenced model predictions, supporting regulatory compliance and model validation.

Beyond its empirical findings, this work contributes a comprehensive and reproducible experimental framework for investigating provenance-aware financial intelligence systems. The proposed methodology integrates a permissioned blockchain simulation, a structured taxonomy of realistic transaction tampering attacks, comparative evaluation against standalone blockchain and machine learning baselines, statistical validation through bootstrap confidence intervals and robustness analysis, scalability assessment, and model explainability within a single experimental pipeline. Collectively, these contributions establish a practical methodological foundation for future research at the intersection of blockchain technology and financial machine learning. As financial institutions continue to increase their reliance on automated decision-making and distributed digital infrastructures, integrating provenance verification with intelligent fraud detection offers a promising direction for improving transaction integrity, operational transparency, regulatory compliance, and trust in next-generation multi-institutional financial systems.

References

Aashish, K. C., Zamil, M. Z. H., Mridul, M. S. I., Akter, L., Sharmin, F., Ayon, E. H., et al. (2025). Towards eco-friendly cybersecurity: Machine learning-based anomaly detection with carbon and energy metrics. International Journal of Applied Mathematics, 38(9s).

Alam, M., Shil, S. K., Sharmin, F., KC, A., Md, A. H., Ali, M., et al. (2026). Hybrid deep learning models for equipment failure prediction in US industrial systems. International Journal of Applied Mathematics, 39(1s).

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Sorniotti, A., Stathakopoulou, C., Vukolić, M., Weed Cocco, S., & Yellick, J. (2018). Hyperledger Fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference (pp. 1–15). Association for Computing Machinery. https://doi.org/10.1145/3190508.3190538

Bahnsen, A. C., Aouada, D., Stojanovic, A., & Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. Expert Systems with Applications, 51, 134–142. https://doi.org/10.1016/j.eswa.2015.12.030

Bhowmik, P. K., Subha, D. T., Rahim, A., Mohammed, A. A., Begum, M., Chowdhury, R., et al. (2026). Self-adaptive machine learning models for financial risk forecasting: Handling non-stationarity in banking and cryptocurrency time series.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). Association for Computing Machinery. https://doi.org/10.1145/2939672.2939785

Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. In 2015 IEEE Symposium Series on Computational Intelligence (pp. 159–166). IEEE. https://doi.org/10.1109/SSCI.2015.33

Dola, A., Begum, S., Antara, U. K., Islam, M. R., Sultana, T., & Zabin, N. (2024). Machine learning models for detecting hidden collusion networks in US corporate finance. Journal of Economics, Finance and Accounting Studies, 6(1), 143–154.

Guo, Y., & Liang, C. (2016). Blockchain application and outlook in the banking industry. Financial Innovation, 2, Article 24. https://doi.org/10.1186/s40854-016-0034-9

Hossain, M. S., Rahman, M. K., Haque, S. U., Jahed, K. A., Mithu, M. F. H., Akter, M., et al. (2026). Causal machine learning for financial crime attribution: Uncovering hidden cause-and-effect relationships among fraud, money laundering, and cybersecurity incidents in blockchain ecosystems. International Journal of Computer Information Systems and Industrial Management Applications, 18(5s), 434–451.

Islam, M. R., Subha, D. T., Pramanik, M. T., Akter, M., Sweet, M. M. R., Robbani, M. S., et al. (2026). AI-driven decision support systems for optimizing working capital and customer experience in the US: A transaction-based simulation framework for SMEs.

Islam, M. Z., Sumsuzoha, M., Islam, M. R., Kawsar, M., Mithu, M. F. H., Pant, S., et al. (2026). Graph neural networks for systemic financial risk forecasting: Modeling cross-market contagion between banking systems and cryptocurrency markets.

Jakir, T. (2025). Signal-to-noise analysis of crisis indicators in global finance using artificial intelligence. International Journal of Applied Mathematics, 38(10s), 1815–1836.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In Advances in Neural Information Processing Systems, 30 (pp. 3146–3154).

Kim, H. M., & Laskowski, M. (2018). Toward an ontology-driven blockchain design for supply-chain provenance. Intelligent Systems in Accounting, Finance and Management, 25(1), 18–27. https://doi.org/10.1002/isaf.1424

Lebichot, B., Le Borgne, Y.-A., He-Guelton, L., Oblé, F., & Bontempi, G. (2020). Deep-learning domain adaptation techniques for credit card fraud detection. In L. Oneto, N. Navarin, A. Sperduti, & D. Anguita (Eds.), Recent Advances in Big Data and Deep Learning (pp. 78–88). Springer.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems, 30 (pp. 4765–4774).

Lundberg, S. M., Erion, G. G., Chen, H., DeGrave, A. J., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. Nature Machine Intelligence, 2, 56–67. https://doi.org/10.1038/s42256-019-0138-9

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

Peters, G. W., & Panayi, E. (2016). Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In P. Tasca, T. Aste, L. Pelizzon, & N. Perony (Eds.), Banking beyond banks and money (pp. 239–278). Springer.

Pramanik, M. T., Reza, S. A., Kamal, M. M., Gomes, S. S., Faysal, A. H. M., Minto, A., et al. (2026). Modeling resilient and sustainable supply chain networks with AI: A comprehensive approach to designing robust supply systems in the USA. International Journal of Computer Information Systems and Industrial Management Applications, 18(4s), 1396–1409.

Rahman, M. K., Hossain, M. S., Haque, S. U., Jahed, K. A., Robbani, M. S., Shati, M. A., et al. (2026). Machine learning models for early warning of financial crises in the US economy using macro-financial indicators.

Rahman, M. S. (2025). Machine learning-enabled early warning system for detecting micro-inflation clusters in the US economy. International Journal of Applied Mathematics, 38(12s), 2743–2769.

Reza, S. A., Rahman, M. K., Rahman, M. D., Sharmin, S., Mithu, M. F. H., Hasnain, K. N., et al. (2025). Machine learning-enabled early warning system for financial distress using real-time digital signals. arXiv preprint arXiv:2510.22287.

Shawon, R. E. R., et al. (2025). Detecting illicit cross-chain fund movement: Behavioral machine learning models for bridge-based laundering patterns. International Journal of Applied Mathematics, 38(12s).

Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., & Leiserson, C. E. (2019). Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics. In Proceedings of the 2019 KDD Workshop on Anomaly Detection in Finance.