MATHEMATICAL MODELING AND COMPUTING, Vol. 11, No. 3, pp. 893–903 (2024)

Application of machine learning algorithms to enhance blockchain network security

Solomka I. R., Liubinskyi B. B., Torshyn V. V.

Lviv Polytechnic National University,
12 S. Bandera Str., 79013, Lviv, Ukraine

(Received 24 April 2024; Revised 25 September 2024; Accepted 26 September 2024)

This paper embarks on a detailed examination of the inherent security challenges faced by blockchain networks, including fraudulent transactions, double-spending, and 51% attacks, among others. Using recent advancements in ML, it presents a novel methodology for real-time anomaly detection, predictive threat modeling, and adaptive security protocols that leverage data-driven insights to fortify the blockchain against both known and emerging threats. By analyzing case studies and empirical data, this study illustrates the effectiveness of ML techniques in enhancing the resilience and integrity of blockchain systems. Furthermore, it explores the implications of these innovations for future blockchain applications, proposing a framework for the integration of ML into blockchain security strategies. This article aims to serve as a cornerstone for researchers, technologists, and cybersecurity professionals, offering insights into the future of secure blockchain ecosystems powered by the intelligent capabilities of machine learning.

Keywords: *blockchain; machine learning; security; consensus mechanism.*

2010 MSC: 94-04, 94A05, 94A60 **DOI:** 10.23939/mmc2024.03.893

1. Introduction

The primary issue addressed by this article is the increasing complexity and sophistication of security threats targeted at blockchain networks. Despite inherent security features of blockchain such as decentralization and cryptographic hashing, there still exist vulnerabilities that malicious actors can exploit. These vulnerabilities include fraudulent and anomalous transactions, double-spending threats, "51%" attacks, exploitation of smart contracts, and privacy breaches. The article aims to address the problem of proactively detecting, predicting, and mitigating these threats in real-time to support the integrity, reliability, and resilience of blockchain systems. It explores how machine learning algorithms can be used to enhance blockchain security by analyzing patterns, detecting anomalies, and adapting to new threats more effectively than traditional security measures. The ultimate goal is to develop a framework that integrates machine learning into blockchain security strategies, thereby reducing the risk of attacks and ensuring the long-term resilience of blockchain infrastructure.

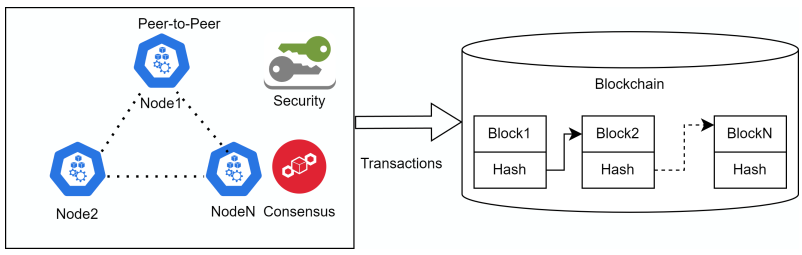

Fig. 1. Architecture of a decentralized blockchain network.

Blockchain technology represents a paradigm shift in how information is exchanged, verified, and recorded in decentralized networks. Originally developed to support the digital currency Bitcoin, blockchain has extended beyond its financial origins to impact various sectors including healthcare, supply chain management, and digital identity verification. Fundamentally, blockchain is a distributed ledger technology (DLT) that maintains a continuously growing list of records, called blocks, securely

linked together using cryptography. This architecture (Figure 1) ensures the integrity, transparency, and immutability of the data stored within the blockchain, making it resistant to modification and fraud.

The essence of blockchain lies in its ability to facilitate peer-to-peer transactions without the need for a central authority. This decentralization is achieved through consensus mechanisms such as Proof of Work (PoW) [1] and Proof of Stake (PoS) [2], which allow network participants to agree on the validity of transactions. As a result, blockchain creates an environment of trust among users, who can conduct transactions directly with one another, ensuring the security and continuity of their exchanges.

2. Analysis of recent research and publications

Among the most critical security threats to blockchain [3] is the potential for a 51% attack [4], where a single entity or coalition acquires more than half of the network's mining power, thereby gaining the ability to manipulate transaction verification and block creation. Such dominance over the network can facilitate dishonest actions, such as double-spending threats, where identical digital tokens are fraudulently obtained multiple times [5]. Examples include Ethereum Classic, which suffered several 51% attacks in 2020, leading to double expenditures of ETC worth millions of dollars.

One of the most notorious cryptocurrency thefts occurred with the hack of the Mt. Gox exchange in 2014 [6], where approximately 744 408 bitcoins were lost. This incident had significant repercussions for the global cryptocurrency community and has been the subject of numerous studies [7] in information security.

Research has shown that the initial vulnerabilities of the Mt. Gox system can be traced back to 2011, when the exchange's private key remained unencrypted and was likely compromised through insider actions or as a result of a hacking attack. Such conditions allowed perpetrators to covertly withdraw funds, which the system mistakenly interpreted as legitimate transactions or transfers to secure addresses.

Fraudulent anomalous transactions in blockchain are any attempts to deceive or manipulate transactions to gain unauthorized benefits. These transactions exploit the trust and decentralized nature of blockchain networks to conduct illegal activities, including, but not limited to, theft, transaction manipulation, or unauthorized access to assets. The anonymity and irreversibility of blockchain transactions, while being strengths in terms of privacy and security, also pose challenges in combating fraud.

Double spending occurs when an entity spends the same digital currency more than once. This can happen if a malefactor manipulates the network in a way that allows them to reverse a transaction after spending the cryptocurrency, effectively enabling them to retain both the spent currency and the acquired goods or services.

Transaction malleability is a vulnerability that allows changing the unique transaction identifier (or hash) before the transaction is confirmed. This can lead to confusion over whether a transaction has been executed, potentially allowing fraudsters to claim that the transaction never occurred, and then making a double claim.

Furthermore, the Sybil attack presents a significant risk, characterized by a malefactor controlling multiple nodes in the network using numerous pseudonyms. This disproportionate influence aims to undermine the network's operation and consensus mechanisms, threatening the fundamental basis of equal participation in the blockchain [5].

Phishing attacks further exacerbate the security situation by deceitfully inducing individuals to disclose confidential information, such as personal keys or authentication credentials. These attacks exploit the human factor, using forged communications that masquerade as legitimate organizations to execute their schemes. Blockchain's reliance on the broader Internet infrastructure creates vulnerabilities through routing attacks. These attacks exploit weaknesses in Internet routing protocols, such as BGP, to intercept or alter data transmitted between network nodes without compromising the cryptographic protocols that protect the data [4].

The advent of smart contracts as self-executing contractual agreements encoded on the blockchain opens a new vector for potential security flaws. Vulnerabilities in the logic or implementation of smart contracts can be manipulated to alter expected outcomes or siphon funds, underscoring the critical need for thorough code audits and security methodologies [8].

Furthermore, the security of blockchain assets is fundamentally linked to the protection of private keys. Unauthorized acquisition of a user's private key equates to direct control over the user's assets on the blockchain, representing a direct and serious threat to asset security [9].

Finally, the integrity and functionality of blockchain systems largely depend on the reliability of their consensus mechanisms. Flaws or inefficiencies in these mechanisms can lead to vulnerabilities, including unintentional forks, security breaches, and potential centralization pressures, which challenge the decentralized nature of blockchain technology [10].

Based on the analyzed data, a summary table of vulnerabilities in key consensus mechanisms has been constructed.

Table 1. Consensus mechanisms and their vulnerability to major attacks.

Consensus mechanism	51% attack	Sybil attack	Phishing	Routing attack	Smart contract vulnerability	Private key theft
Proof of Work	High	Low	N/A	Medium	N/A	N/A
Proof of Stake	Medium	Medium	N/A	Medium	N/A	N/A
Delegated PoS	Medium	High	N/A	Medium	N/A	N/A
Practical byzantine fault tolerance	Low	Low	N/A	High	N/A	N/A
Directed acyclic graphs	Low	Medium	N/A	Medium	N/A	N/A
Hybrid Mechanisms	Varies	Varies	N/A	Varies	N/A	N/A

The notation "N/A" signifies that the security threat is not directly influenced by the consensus mechanism itself, but rather by other factors associated with blockchain technology and user practices. "Varies" for hybrid mechanisms reflects the diverse nature of these systems, which may combine elements of different consensus types and, thereby, inherit a mixture of their vulnerabilities.

The vulnerability level of each consensus mechanism to specific threats can vary depending on additional factors, such as network size, implementation details, and external security measures. Ongoing research and development efforts are directed towards mitigating these vulnerabilities and enhancing the resilience of blockchain systems against new security threats.

To improve the security of blockchain technologies and mitigate identified vulnerabilities, several methods implemented in the field have been reviewed. A comprehensive analysis of these methods, along with relevant references, provides valuable information about current efforts to protect blockchain systems from potential threats.

One approach to reducing the risk of 51% attacks involves using more decentralized and scalable consensus mechanisms that do not rely solely on proof-of-work (PoW). The implementation of hybrid consensus models that combine PoW with proof-of-stake (PoS) mechanisms can reduce the feasibility of these attacks by increasing the cost and complexity of gaining majority control [11]. Additionally, deploying checkpoint mechanisms, where trusted block hashes are hard coded into the software at specific intervals, can also prevent deep chain reorganizations [12].

To counter Sybil attacks, networks can implement more stringent node verification processes, such as requiring nodes to deposit a certain amount of native cryptocurrency as collateral (proof of stake) or using reputation systems that assess the reliability of nodes based on their activity [4]. These measures make it costly and challenging for malicious actors to gain significant influence over the network.

Educating users on the risks of phishing and the use of multi-factor authentication (MFA) can significantly reduce the success rate of phishing attacks. Blockchain wallets and services can integrate hardware security keys as part of the authentication process, providing an additional layer of protection against phishing [13].

Enhancing the security of the underlying internet infrastructure on which blockchain networks operate is crucial. Implementing secure routing protocols, such as BGPsec and RPKI, can reduce the risk of routing attacks. Additionally, the use of Transport Layer Security (TLS) for data transmitted between nodes ensures that even if data is intercepted, it remains encrypted and unintelligible to the attacker.

Research into alternative consensus mechanisms that offer improved security, scalability, and decentralization continues. Proof-of-stake (PoS) and its variations (Delegated PoS, Liquid PoS) provide mechanisms where the probability of verifying transactions and creating new blocks is proportional to the stake in the network, reducing the potential for centralization and attacks [14].

The analysis indicates that machine learning methods have been scarcely utilized for identifying anomalous and fraudulent transactions. This highlights an area for potential development in enhancing blockchain security through advanced analytical techniques.

3. Methodology for using machine learning to identify anomalous transactions

The integration of Machine Learning (ML) into the security domain, particularly within innovative technologies like blockchain, provides a robust toolkit for enhancing protection, predicting vulnerabilities, and optimizing security protocols. To understand how ML contributes to these aspects, it is crucial to comprehend the fundamentals of ML models and their applicability to security challenges.

Machine learning, a subset of artificial intelligence (AI), enables systems to learn from data, identify patterns, and make decisions with minimal human intervention. ML models are trained on large datasets, which allows them to improve their accuracy over time by processing more information.

Integrating ML into security strategies offers several benefits. By automating the detection of anomalies and potential vulnerabilities, ML models can significantly reduce the time required to identify and respond to security threats. This rapid response capability is crucial for limiting the damage caused by cyber-attacks.

ML models are also capable of processing the vast volumes of data generated by modern networks and systems, sifting through logs and real-time traffic to detect signs of fraud that human analysts might miss. Furthermore, as cyber threats evolve, ML models can adapt to new types of attacks, providing a dynamic defense mechanism that evolves alongside the increasing threat landscape.

However, the effectiveness of ML in security also depends on the quality of the data used for training, the specificity of the models for security tasks, and continuous updating of the models to adapt to new threats. Additionally, there is a challenge in securing the ML models themselves and preventing malicious actors from manipulating them to bypass security measures.

Figure 2 illustrates the process and methodology for using machine learning to identify anomalous transactions.

The first step involves collecting a large amount of transactional data. This data must include both normal and anomalous transactions. It is also crucial to ensure that the data is cleaned of errors, duplicates are removed, and missing values are processed.

During the feature engineering stage, those parameters of the input data are selected which will best assist the machine learning algorithm in performing the assigned task. For the task of detecting anomalous transactions, such parameters might include:

- Transaction amount: large sums may indicate potential fraud.
- Transaction time: transactions occurring at unusual times, for example, during the night.
- Transaction frequency: if transactions occur too frequently or involve too large sums over a short period.
- Geographical location: transactions taking place in regions with high levels of fraud or in unusual locations for the user.

Often, existing data can be transformed or combined to create new features that provide a deeper understanding of user behavior. For example, changes in frequency or volume of transactions compared to the average for a given user may indicate fraudulent transactions.

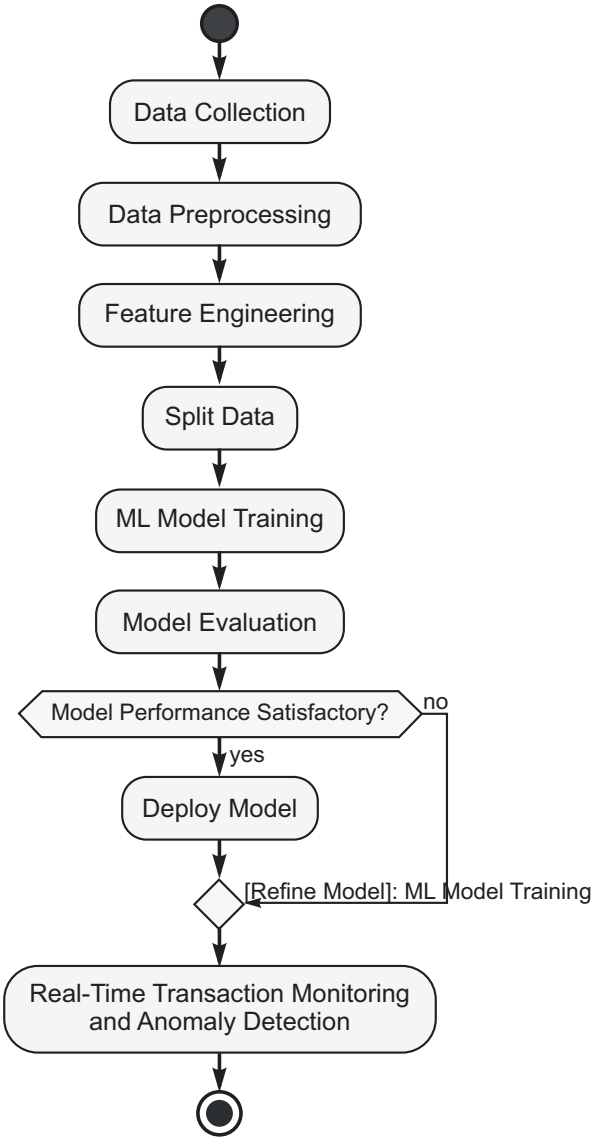

Fig. 2. ML training process.

After feature development, it is necessary to evaluate their effectiveness on a validation dataset to ensure that they indeed enhance the model. This process is often iterative, where the analyst may return to the feature creation step to refine or replace them.

Each machine learning model has a set of parameters that are not learned directly from the data but are defined before training the model. These parameters are called hyperparameters, and their proper tuning can significantly improve model performance.

After selecting and configuring the hyperparameters, the model is trained on the training dataset. The training process (Figure 2) includes the following steps:

1. Data splitting: the data are divided into training, validation, and test sets. The validation set is used to evaluate the model during training, while the test set is used for the final assessment of performance.
2. Model training: the model learns to adjust its parameters (e.g., weights in a neural network) according to the training data, minimizing the prediction error.
3. Cross-validation: applying cross-validation techniques to ensure that the model performs effectively across different data subsets. This helps to avoid overfitting and ensure overall reliability.
4. Model evaluation: using metrics such as accuracy, recall, precision, F1-score, and AUC-ROC to assess the performance of the model.

After initial training, further iterations may be possible to enhance performance by adjusting hyperparameters, adding or removing features, or using different models. This cyclical process helps to adapt the model to changing conditions and new types of data [15].

ML models, especially those specialized in anomaly detection, can identify unusual transaction patterns that deviate from the norm, indicating potential fraud. Common methods used for this purpose include clustering (e.g., K-means) and neural networks (e.g., autoencoders).

Pattern recognition: supervised learning models such as decision trees, support vector machines, and neural networks can be trained on historical transaction data labeled as fraudulent or legitimate. These models learn to recognize patterns associated with fraud, enabling them to accurately classify new transactions [16].

Sequential analysis: some ML models specialize in analyzing sequences of transactions to detect fraudulent patterns over time. Techniques such as recurrent neural networks and long short-term memory networks are well-suited for processing time-series data, making them effective for identifying complex fraud schemes that occur over extended periods.

4. Choosing a machine learning model

The experimental workflow consists of several stages, including data preprocessing, feature development, model selection, training, evaluation, and validation. The dataset is divided into training and testing sets using stratified splitting to ensure a balanced distribution of anomalous and normal transactions. Models are trained on the training data and evaluated on the test data using selected performance metrics. Hyperparameter tuning is conducted using methods such as grid search or randomized search to optimize model performance. Finally, based on the evaluation results, the best model is selected and deployed for real-world use.

Various tools and frameworks can be used to facilitate the implementation, training, evaluation, and visualization of a machine learning (ML) model aimed at enhancing blockchain security. Below are key tools and frameworks used in the development process:

- **Python programming language**: the simplicity, versatility, and extensive libraries of Python make it well-suited for developing and experimenting with ML models.
- **Pandas and NumPy**: these Python libraries are used for data manipulation, preprocessing, and feature development. Pandas provides data structures like DataFrames, essential for processing and analyzing structured data, while NumPy offers support for numerical operations and array processing.
- **Scikit-learn (sklearn)**: this is a popular Python machine learning library that provides a wide range of tools and algorithms for ML tasks such as classification, regression, clustering, and dimensionality reduction.
- **Matplotlib and Seaborn**: these Python libraries are used for data visualization. Matplotlib offers a broad range of plotting functions and customization options, while Seaborn provides high-level interfaces for creating informative and visually attractive statistical graphics. These libraries are utilized for visualizing data distributions, model performance metrics, and decision boundaries.

The random forest model has been employed to build a model for detecting fraudulent transactions due to its significant advantages for this class of tasks. This model's robustness and versatility in handling various complexities associated with transaction data make it particularly suitable for identifying discrepancies indicative of fraud. Advantages of using the random forest model for detecting fraudulent transactions:

1. High accuracy: the random forest method often demonstrates high accuracy in various classification tasks through the use of multiple decision trees to make a final prediction. The ensemble method of voting for classification or averaging for regression ensures a balanced decision, reducing the likelihood of overfitting on a single tree.
2. Handling large data sets: random forest can efficiently process datasets with a large number of records and features, making it ideal for complex tasks where other models may struggle.

3. Capability to handle missing data: the model can effectively manage cases with missing data. During the construction of decision trees, it can use the average values of variables to substitute for missing data, thus avoiding the loss of important information.
4. Minimization of overfitting: compared to individual decision trees, random forest generally does a better job of avoiding overfitting on the training dataset. Since the final decision is made based on a large number of trees, the effect of overfitting is minimized, making the model more generalizable.
5. Feature importance: random forest can provide an assessment of the importance of features used for classification, allowing a deeper understanding of which factors influence the predicted outcome. This is particularly useful for model interpretation and identifying key features in the data.
6. Flexibility: the model can be used for both classification and regression tasks, making it a versatile tool for various types of data and applications.
7. Ease of use: thanks to high-quality implementations available in popular machine learning libraries like scikit-learn, the random forest model is easy to integrate and use in projects.
8. These attributes make Random Forest an excellent choice for developing a robust model for detecting fraudulent transactions, enhancing the security of blockchain systems, and other similar applications requiring reliable and interpretable results.

At its core, random forest utilizes multiple decision trees. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g., whether a coin toss results in heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (the decision made after computing all attributes). Paths from the root to the leaves represent classification rules.

The fundamental principle underlying Random Forest involves combining several decision trees to form a more reliable and accurate predictive model. Ensemble predictions are more accurate and stable than any single decision tree, thanks to the averaging process which reduces both variance and bias. This methodology enhances the overall decision-making capability by leveraging the strengths of multiple trees to offset the weaknesses of individual components.

5. Random forest algorithm for predictive modeling

Consider a dataset D consisting of n independent observations $D = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\}$, where X_i represents the feature vector of the i-th observation, and Y_i represents its target output. For each of k trees, a sample D_k is selected from the dataset D with replacement, adding an additional level of randomness to the training of each tree.

Each tree T_k is constructed using the dataset D_k. At each node, a subset of m features is randomly selected from the complete set of features. The best split based on these m features is used to divide the node. The criteria for the "best" split can vary [17]. The process continues until the maximum depth of the tree is reached.

After training, the model is used for classification and prediction. For an input vector X, each tree T_k provides a prediction $Y_k(X)$. The final prediction is the mode of the predictions obtained from all the trees for classification tasks:

$$Y(X) = \text{mode}\left\{Y_1(X), Y_2(X), \ldots, Y_k(X)\right\}.$$

For regression tasks, the final prediction is the arithmetic mean of all values obtained from each tree,

$$Y(X) = \frac{1}{k}\sum_{i=1}^{k} Y_i(X).$$

Random forest also provides insights into the importance of each feature in prediction. This is often calculated by measuring how much each feature decreases the weighted impurity in the tree, averaged across all trees. The mathematical elegance of the random forest method lies in its simplicity and the powerful concept of creating a "forest" of random decision trees to achieve a highly efficient model. The randomness introduced through bootstrapping and feature selection ensures that the ensemble model is more robust and less prone to overfitting compared to a single decision tree.

Ultimately, the selection of random forest for building a model for classifying fraudulent transactions is justified by its versatility, high accuracy, and ability to efficiently handle complex datasets while minimizing the risk of overfitting.

Feature selection is a critical step in model development. For this experiment, the transaction amount and the transaction type were considered as primary features for detecting fraud. The transaction type is encoded distinctly to represent categorical variables in digital form. Additional feature engineering methods can be applied to derive new features or improve model performance.

The effectiveness of ML models is evaluated using various metrics adapted to fraud detection tasks. These metrics include accuracy, precision, recall, F1 score, the receiver operating characteristic (ROC) curve, and the area under the ROC curve (AUC) [18]. The choice of evaluation metrics depends on the specific requirements and objectives of the fraud detection system.

6. Construction and analysis of the software model

The program, written in Python, utilizes the pandas library for data loading and processing, sklearn for constructing and evaluating the machine learning model, and matplotlib for visualizing the results. It is designed to analyze transactions within a blockchain to identify anomalous transactions that may indicate fraudulent activities or vulnerabilities in security. This comprehensive approach integrates data manipulation, predictive modeling, and result visualization to enhance the detection capabilities within blockchain transaction networks.

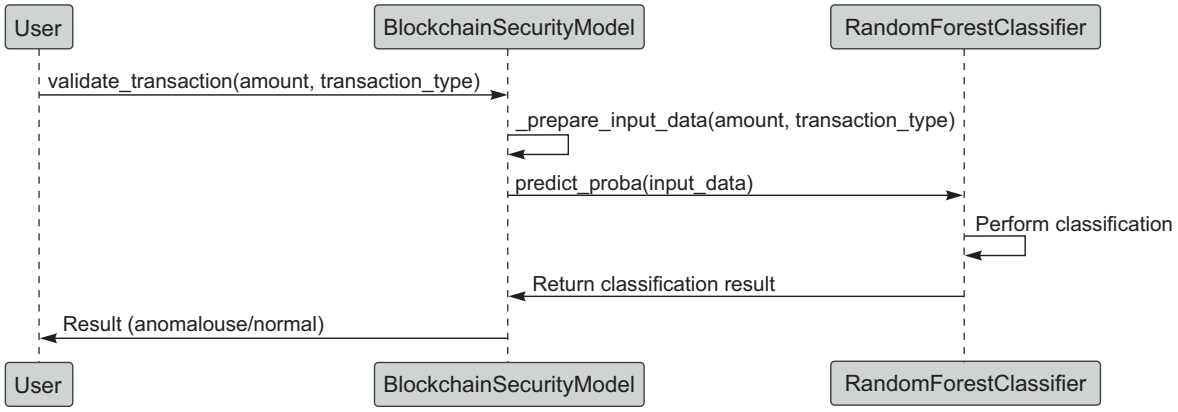

Fig. 3. Application diagram.

For training the model, a test dataset consisting of 100 000 records was utilized, which includes the following fields:

- *transaction_*id: a unique identifier for each transaction.
- *amount*: the transaction amount, generated from a gamma distribution.
- *transaction_type*: the type of transaction, randomly selected from three possible values.
- *timestamp*: the time stamp of the transaction.
- *from_account* and *to_account*: identifiers for the sender's and recipient's accounts.
- *transaction_fee*: the fee associated with the transaction.
- *is_anomalous*: a flag indicating whether the transaction is anomalous.

Data preparation involved selecting the appropriate features (columns) and the target variable. One-Hot Encoding was used to convert the categorical variable transaction_type into a numerical format suitable for machine learning models.

The RandomForestClassifier was employed to train the model on the training set. This stage also included evaluating the model's performance on the test dataset and generating a classification report and ROC-AUC score.

The output of the program includes a classification report (Figure 4) and a ROC curve (Receiver Operating Characteristic curve) [19] — a graphical representation of the classification model's performance at all possible classification thresholds (Figure 5).

Classification Report:

	precision	recall	f1-score	support
0	0.98	0.97	0.97	18667
0	0.60	0.71	0.65	1333
accuracy			0.95	20000
macro avg	0.79	0.84	0.81	20000
weighted avg	0.95	0.95	0.95	20000

Fig. 4. Model classification report.

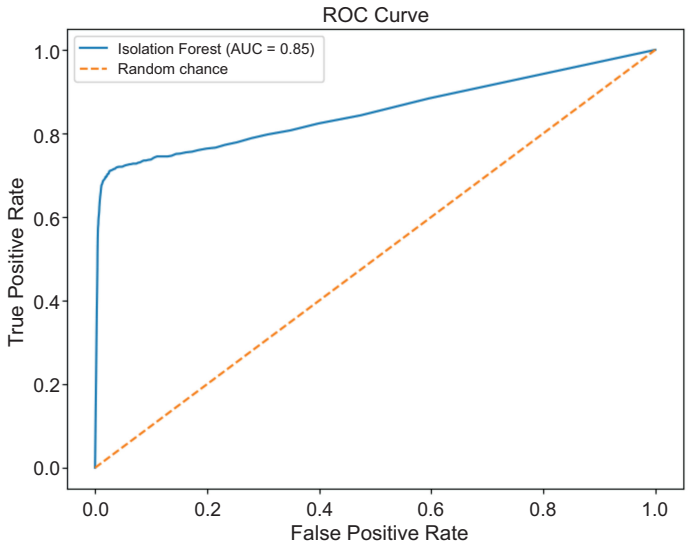

Fig. 5. Receiver Operating Characteristic curve.

7. Classification results interpretation

The classification report (Figure 5) illustrates the model's performance on a dataset comprising 20 000 transactions, of which 18 667 are normal (class 0) and 1 333 are anomalous (class 1). Here is a detailed analysis of the results:

Class 0 – Normal Transactions:

- Accuracy: the model demonstrates an overall accuracy of 95%, indicating a high capability to correctly classify transactions as normal or anomalous.
- Precision: at 98%, this metric signifies that among the transactions classified as normal, 98% are indeed normal.
- Recall: at 97%, this shows that the model successfully identified 97% of all normal transactions within the dataset.
- F1-Score: a balance between precision and recall for class 0 is 0.97, indicating excellent performance.

Class 1 – Anomalous Transactions:

- Precision: at 60%, this indicates that 60% of the transactions classified as anomalous are truly anomalous. Although lower than for class 0, this is still a reasonable rate for anomaly detection.
- Recall: at 71%, this indicates that the model detected 71% of all anomalous transactions, suggesting effective anomaly identification with room for improvement.
- F1-Score: the balance between precision and recall for class 1 is 0.65, which is quite good considering the complexity of detecting anomalies.

Overall Metrics:

— Macro Avg: average values for precision, recall, and F1-score are 0.79, 0.84, and 0.81, respectively. These metrics consider results for both classes, providing a balanced assessment of model performance.

— Weighted Avg: weighted averages for precision, recall, and F1-score, taking into account the support (number of examples) for each class, are 0.95, confirming high overall model performance.

The model demonstrates high effectiveness in classifying normal transactions and a reasonable ability to detect anomalous transactions. Given the challenges associated with anomaly detection, the results for class 1 are promising but indicate potential for further improvement, especially in enhancing the precision of anomalous transactions.

Figure 5 displays the ROC curve for the Isolation Forest algorithm with an AUC (Area Under the Curve) of 0.85. The ROC curve graphically represents the trade-off between sensitivity (True Positive Rate) and specificity (1 — False Positive Rate) at various classification thresholds.

- True Positive Rate (TPR), or sensitivity, shows the proportion of anomalous transactions correctly identified by the model. The curve rises swiftly at low values of False Positive Rate (FPR), indicating good performance.
- False Positive Rate (FPR) shows the proportion of normal transactions falsely identified as anomalous. Ideally, a low FPR at a high TPR is desired, represented by a curve that rises towards the upper left corner.
- Random Chance Line: the diagonal dashed line represents random choice. If the ROC curve of the model is above this line, it indicates that the model performs better than random choice.
- AUC Value: an AUC of 0.85 suggests that the model has a fairly high ability to distinguish between classes. AUC values range from 0 to 1, where 1 indicates a perfect ability of the model to correctly classify all instances.

Based on the ROC curve, it can be concluded that the employed Isolation Forest algorithm performs well in detecting anomalous transactions, significantly outperforming random selection and showing substantial potential for further optimization to improve performance.

This program serves as a robust example of applying machine learning to address a specific issue in blockchain transaction security, demonstrating the steps of data preparation, model training, evaluation, and real-world application.

8. Conclusions

This article presents a theoretical justification, development process, and implementation strategies for a machine learning model aimed at enhancing the security of blockchain technology. The starting point of the research is the analysis of vulnerabilities in blockchain systems, based on which a comprehensive methodological approach to using machine learning algorithms for detecting anomalies in blockchain transactions that may indicate security threats has been developed.

The architectural scheme of the model meticulously illustrates the integration of selected tools and technologies at each stage of the machine learning model lifecycle, including data collection, preprocessing, model training, performance evaluation, and real-time monitoring.

The proposed machine learning model demonstrates significant potential for applying artificial intelligence to strengthen the protection of blockchain technologies, considering both current and potential future threats. The detailed description of the processes of development and implementation of security systems based on machine learning for blockchain applications is provided through the analysis of the components and procedures discussed in the article.

Overall, the article offers an in-depth analysis of the prospects for using machine learning in the context of blockchain security, highlighting key methodological approaches, tools, and frameworks necessary for creating effective solutions in this area. The machine learning model discussed not only contributes to enhancing the security of blockchain networks but also plays a crucial role in forming the foundation for integrating artificial intelligence technologies to address complex cybersecurity challenges.

[1] Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System (2008).

[2] King S., Nadal S. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake (2012).

[3] Charts B. Bitcoin network. `http://bitcoincharts.com/bitcoin/` (2024).

[4] Douceur J. R. The Sybil Attack. Peer-to-Peer Systems. 251–260 (2002).

[5] Eyal I., Sirer E. G. Majority is not enough: Bitcoin mining is vulnerable. Communications of the ACM. **61** (7), 95–102 (2018).

[6] Mt. Gox Hack – The Most Iconic Exchange Hack, Blockonomi. `https://blockonomi.com/mt-gox-hack/` (2024).

[7] Weichbroth P., Wereszko K., Anacka H., Kowal J. Security of Cryptocurrencies: A View on the State-of-the-Art Research and Current Developments. Sensors. **23** (6), 3155 (2023).

[8] Apostolaki M., Zohar A., Vanbever L. Hijacking Bitcoin: Routing Attacks on Cryptocurrencies. 2017 IEEE Symposium on Security and Privacy. 375–392 (2017).

[9] Eskandari S., Barrera D., Stobert E., Clark J. A First Look at the Usability of Bitcoin Key Management. Workshop on Usable Security (2015).

[10] Atzei N., Bartoletti M., Cimoli T. A Survey of Attacks on Ethereum Smart Contracts (SoK). Principles of Security and Trust. 164–186 (2017).

[11] Buterin V. A Next-Generation Smart Contract and Decentralized Application Platform. `https://github.com/ethereum/wiki/wiki/White-Paper` (2017).

[12] Zohar A. Bitcoin: under the hood. Communications of the ACM. **58** (9), 104–113 (2015).

[13] Unger N., Dechand S., Bonneau J., Fahl S., Perl H., Goldberg I., Smith M. SoK: Secure Messaging. 2015 IEEE Symposium on Security and Privacy. 232–249 (2015).

[14] Kiayias A., Russell A., David B., Oliynykov R. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. Advances in Cryptology – CRYPTO 2017. 357–388 (2017).

[15] Kuhn M., Johnson K. Nonlinear classification models. In: M. Kuhn, K. Johnson (eds). Applied Predictive Modeling. New York, Springer (2013).

[16] Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques. Book, Third Edition (2012).

[17] Chen B., Liu Y., Zhang C., Wang Z. Time Series Data for Equipment Reliability Analysis With Deep Learning. IEEE Access. **8**, 105484–105493 (2020).

[18] Zhang X., Li X., Feng Y., Liu Z. The use of ROC and AUC in the validation of objective image fusion metrics. Signal Processing. **115**, 38–48 (2015).

[19] Fawcett T. An introduction to ROC analysis. Pattern Recognition Letters. **27** (8), 861–874 (2006).

Застосування алгоритмів машинного навчання для підвищення безпеки мережі блокчейн

Соломка І. Р., Любінський Б. Б., Торшин В. В.

Національний університет "Львівська політехніка", вул. С. Бандери, 12, 79013, Львів, Україна

У статті проведено детальне дослідження проблем безпеки, з якими стикаються мережі блокчейн, включаючи аномальні транзакції, повторні витрати та атаки типу "51%". Вона розглядає останні досягнення в галузі машинного навчання та пропонує нову методологію для виявлення аномалій в режимі реального часу, прогнозування загроз і адаптивних протоколів безпеки, які використовують дані для зміцнення блокчейна як від відомих, так і від нових загроз. Аналізуючи дослідження випадків та емпіричні дані, стаття демонструє ефективність методів машинного навчання у підвищенні стійкості та цілісності систем блокчейна. В статті також досліджено вплив цих інновацій на майбутні використання блокчейна, пропонуючи рамки для інтеграції машинного навчання в стратегії безпеки блокчейна.

Keywords: *блокчейн; машинне навчання; безпека; механізми консенсусу.*