# AI-Driven Digital Twin Framework for Predictive Risk Assessment of Upgradeable Smart Contracts in Blockchain-Based Metaverse Applications

Salma Al-Shammari[1,*]

[1]Business Administration Department, Amman Arab University, Jordan

## ABSTRACT

The increasing adoption of blockchain technology in metaverse applications has accelerated the use of upgradeable smart contracts to support maintainability and functional evolution without redeployment. However, the proxy mechanisms enabling upgradeability also introduce additional security risks related to delegation, administrative control, and storage management. Existing studies primarily focus on proxy smart contract detection and provide limited support for predictive security assessment under different operational conditions. This study proposes an AI-Driven Digital Twin framework for predictive smart contract security analysis that integrates leakage-safe machine learning with scenario-based Digital Twin simulation. A dataset consisting of 377 Ethereum smart contracts, including 17 proxy and 360 non-proxy contracts, was analyzed using static structural features extracted from Solidity source code. Logistic Regression and Random Forest classifiers were evaluated using leakage-safe grouped cross-validation to prevent structurally similar contracts from appearing simultaneously in the training and testing sets. The Digital Twin representation was subsequently used to estimate heuristic security risks under four operational scenarios, namely Baseline, Upgrade, Adversarial Caller, and High Load. The experimental results show that Logistic Regression achieved the best overall performance with 97.61% accuracy, 98.75% balanced accuracy, perfect recall, an F1-score of 0.7907, a Matthews Correlation Coefficient of 0.7984, a ROC-AUC of 0.9980, and a PR-AUC of 0.9647. Feature importance analysis identified EIP-1967-related indicators as the strongest predictors of proxy contracts, while the Digital Twin simulation consistently assigned higher risk scores to proxy contracts across all evaluated scenarios, with the Upgrade scenario producing the highest estimated risk. These findings demonstrate that the proposed framework extends conventional proxy detection by providing leakage-safe classification together with scenario-dependent predictive security assessment, supporting pre-deployment decision making for secure blockchain-based metaverse applications.

**Keywords** Digital Twin, Smart Contract Security, Blockchain, Metaverse, Predictive Security Analysis

Submitted: 18 June 2026
Accepted: 10 August 2026
Published: 2 November 2026

Corresponding author
Salma Al-Shammari,
s.alshammari@aau.edu.jo

Additional Information and Declarations can be found on page 285

DOI: 10.47738/ijrm.v3i4.86

Copyright
2026 Al-Shammari

Distributed under
Creative Commons CC-BY 4.0

## INTRODUCTION

The rapid growth of metaverse applications has accelerated the adoption of blockchain technology as the underlying infrastructure for managing digital assets, decentralized identities, virtual property ownership, and trusted economic transactions [1]. Smart contracts play a fundamental role in these ecosystems by enabling autonomous execution of predefined agreements without relying on centralized authorities. As metaverse platforms continue to evolve, smart contracts increasingly require functional upgrades to address security vulnerabilities, improve business logic, and accommodate new application requirements while preserving existing contract addresses and

**How to cite this article:** S. Al-Shammari, "AI-Driven Digital Twin Framework for Predictive Risk Assessment of Upgradeable Smart Contracts in Blockchain-Based Metaverse Applications," *Int. J. Res. Metav.*, vol. 3, no. 4, pp. 268-288, 2026.

stored data. To achieve this objective, upgradeable proxy architectures have become widely adopted because they separate contract logic from persistent storage, allowing implementation contracts to be replaced without disrupting deployed applications [2]. Although this architecture significantly improves maintainability and flexibility, it also introduces additional security challenges related to delegatecall execution, storage layout compatibility, administrative privileges, and upgrade authorization, making security assessment of proxy smart contracts an increasingly important research topic.

Recent studies have explored the application of artificial intelligence for smart contract security analysis, including vulnerability detection, malicious contract identification, and proxy contract classification. Machine learning and deep learning techniques have demonstrated promising performance by extracting structural and semantic information from Solidity source code to distinguish proxy contracts from conventional smart contracts [3]. In particular, transformer-based language models, graph neural networks, and traditional machine learning algorithms have been successfully applied to proxy identification using static code analysis. These approaches improve automated security analysis while avoiding the computational overhead associated with symbolic execution or runtime monitoring. However, most existing methods focus primarily on binary classification tasks that determine whether a contract is a proxy contract, providing little support for understanding how identified contracts may behave under different operational conditions. Consequently, current AI-based approaches offer limited assistance for predictive security assessment and decision making before smart contract deployment.

In parallel, Digital Twin technology has emerged as an effective paradigm for creating virtual representations of physical or digital systems to support monitoring, simulation, and predictive analysis throughout their operational lifecycle [4]. Originally developed for cyber-physical systems and industrial applications, Digital Twins have recently been extended to blockchain environments, where virtual representations can be used to evaluate smart contract behavior, analyze system evolution, and assess proposed modifications before deployment. Unlike conventional static analysis techniques, Digital Twins provide a simulation environment in which multiple operational scenarios can be evaluated without affecting the real blockchain. Nevertheless, existing blockchain-oriented Digital Twin studies have primarily focused on monitoring contract execution or evaluating blockchain system behavior after deployment. The integration of Digital Twin technology with artificial intelligence for predictive smart contract security analysis remains relatively unexplored, particularly for upgradeable proxy contracts used in metaverse applications.

Despite the progress achieved in both artificial intelligence and Digital Twin research, several important research gaps remain. First, existing proxy smart contract studies generally terminate after the classification stage and do not extend the analysis toward predictive security assessment. Second, current Digital Twin approaches for blockchain systems rarely incorporate machine learning models capable of automatically identifying upgradeable contracts before simulation. Third, limited research has investigated how static structural information extracted from Solidity source code can be transformed into scenario-dependent security risk estimates that support security-oriented decision making during the software development lifecycle. Finally, many

previous studies evaluate classification models using conventional random data splitting, which may introduce data leakage when contracts derived from the same source code appear in both training and testing datasets, resulting in overly optimistic performance estimates [5]. These limitations highlight the need for a unified framework that combines reliable proxy detection with Digital Twin-based predictive security analysis under a leakage-safe evaluation protocol.

To address these research gaps, this study proposes an AI-Driven Digital Twin framework for predictive smart contract security analysis in blockchain-based metaverse applications. The proposed framework first constructs a static Digital Twin representation by extracting structural and upgradeability-related features from Solidity source code. A leakage-safe machine learning model is then employed to identify proxy smart contracts using grouped cross-validation that prevents structurally similar contracts from appearing simultaneously in the training and testing datasets. Subsequently, the Digital Twin performs scenario-based simulations under Baseline, Upgrade, Adversarial Caller, and High Load operating conditions to estimate heuristic security risks prior to deployment. This approach extends conventional proxy detection by transforming static code characteristics into scenario-dependent predictive security assessments that support developers in evaluating smart contract risks before deployment.

The main contributions of this study are fourfold. First, an AI-Driven Digital Twin framework is proposed to integrate proxy smart contract detection with scenario-based predictive security analysis for blockchain-based metaverse applications. Second, a leakage-safe evaluation methodology based on grouped cross-validation is introduced to provide a more reliable assessment of machine learning performance by mitigating source-code leakage. Third, a static Digital Twin representation is developed to transform structural smart contract characteristics into heuristic security risk estimates across multiple operational scenarios. Finally, the experimental results demonstrate that the proposed framework effectively distinguishes proxy and non-proxy smart contracts while providing meaningful scenario-dependent risk assessments that can support secure smart contract development and pre-deployment security decision making in metaverse ecosystems.

## Literature Review and Related Works

### Smart Contract Security Analysis

Smart contracts have become fundamental components of blockchain-based applications because they enable autonomous execution of predefined agreements without relying on centralized authorities. Despite their advantages, smart contracts remain vulnerable to various security threats, including reentrancy attacks, integer overflow, access-control violations, delegatecall misuse, and storage-related vulnerabilities. Since deployed smart contracts cannot be easily modified, security flaws may result in irreversible financial losses and service disruption. Consequently, numerous studies have focused on improving smart contract security through static analysis, symbolic execution, formal verification, and machine learning techniques [6], [7].

Traditional security analysis methods primarily rely on static code analysis and symbolic execution to identify known vulnerability patterns before deployment. Although these approaches provide high interpretability, they often suffer from scalability limitations due to path explosion and increasing computational complexity when analyzing large smart contracts [8], [9]. Formal verification

techniques can mathematically prove contract correctness under predefined specifications, but they require significant expertise and are difficult to apply in practical software development environments [10]. These limitations have encouraged the adoption of Artificial Intelligence techniques capable of automatically learning structural characteristics from smart contract source code.

## Artificial Intelligence for Smart Contract Classification

Recent advances in Artificial Intelligence have significantly improved automated smart contract analysis. Machine learning algorithms have been applied to detect vulnerable contracts, classify malicious behavior, and identify upgradeable proxy contracts using structural and semantic information extracted from Solidity source code [11], [12]. Conventional machine learning models remain attractive because they provide competitive predictive performance while maintaining relatively low computational complexity and greater interpretability compared with deep learning approaches.

More recently, deep learning architectures have demonstrated superior capability in capturing semantic relationships within smart contract source code. Transformer-based language models and Graph Neural Networks have been successfully employed to represent source code through contextual embeddings and graph structures, allowing more accurate identification of proxy contracts and software vulnerabilities [13], [14]. These approaches exploit richer program semantics than traditional handcrafted features, leading to improved classification performance. Nevertheless, most existing studies formulate smart contract analysis as a binary classification problem that terminates after identifying whether a contract belongs to a specific category. Consequently, these approaches provide limited support for understanding how detected contracts may behave under different operational conditions or estimating their potential security risks before deployment.

## Upgradeable Proxy Smart Contracts

Upgradeable proxy architectures have become the preferred solution for maintaining blockchain applications because they allow implementation logic to evolve without changing deployed contract addresses or persistent storage. Proxy contracts separate execution logic from storage by forwarding function calls through delegation mechanisms, thereby enabling software upgrades while preserving application state [15]. Although this architecture improves maintainability and flexibility, it also introduces additional security challenges related to delegatecall execution, storage-layout compatibility, administrative privileges, and upgrade authorization [16].

Several studies have investigated automatic proxy contract identification using source-code analysis and machine learning techniques. These approaches generally extract lexical, syntactic, or semantic characteristics associated with upgradeability patterns and employ supervised learning algorithms to distinguish proxy and non-proxy contracts [17]. While the reported classification performance is encouraging, existing methods generally conclude their analysis after proxy identification and do not further evaluate how identified proxy contracts may respond under different operational scenarios. As a result, developers receive limited information regarding the relative security implications of upgradeability beyond the classification outcome itself.

### Digital Twin for Blockchain Systems

Digital Twin technology has emerged as an effective paradigm for constructing virtual representations of physical or digital systems to support monitoring, simulation, optimization, and predictive analysis throughout the system lifecycle [18]. Originally developed for cyber-physical systems, Digital Twins have recently been extended to blockchain environments to facilitate virtual evaluation of distributed applications without affecting operational systems. By maintaining a synchronized virtual representation, Digital Twins enable developers to analyze alternative configurations and estimate system behavior before implementing changes in real environments.

In blockchain applications, Digital Twins have been explored for monitoring transaction behavior, evaluating system performance, and assessing proposed modifications to smart contracts before deployment [19]. Compared with conventional static analysis, Digital Twins provide additional capabilities by enabling scenario-based simulation under multiple operational conditions. However, existing blockchain-oriented Digital Twin studies primarily focus on system monitoring or post-deployment evaluation rather than predictive security assessment during software development. Furthermore, the integration of Digital Twin technology with Artificial Intelligence for automated smart contract security analysis remains relatively limited.

### Research Gap

Although previous studies have demonstrated significant progress in smart contract security analysis, several important research gaps remain. First, most Artificial Intelligence approaches focus exclusively on proxy contract classification without extending the analysis toward predictive security assessment under different operational conditions. Second, Digital Twin studies have mainly emphasized blockchain monitoring and behavioral evaluation after deployment rather than integrating intelligent contract identification with virtual security simulation. Third, existing research rarely transforms static structural characteristics extracted from Solidity source code into scenario-dependent security risk estimates that support pre-deployment decision making. Finally, many previous studies employ conventional random train-test splitting, which may introduce data leakage when contracts derived from identical or highly similar source code appear in both training and testing datasets, thereby producing overly optimistic performance estimates [20]. To address these limitations, this study proposes an AI-Driven Digital Twin framework that combines leakage-safe machine learning with scenario-based Digital Twin simulation to perform predictive security analysis for upgradeable smart contracts in blockchain-based metaverse applications.

## Methodology

### Research Framework

This study proposes an AI-Driven Digital Twin framework for predictive smart contract security analysis in blockchain-based metaverse applications. The overall research workflow is illustrated in figure 1, which consists of five sequential stages: (1) dataset preparation and preprocessing, (2) static Digital Twin construction through feature extraction, (3) leakage-safe proxy smart contract classification using machine learning, (4) scenario-based Digital Twin simulation, and (5) performance evaluation. The proposed framework

transforms Solidity source code into a virtual Digital Twin representation capable of supporting both proxy smart contract identification and scenario-dependent security risk estimation before deployment. The workflow begins with preprocessing and normalization of Solidity source code, followed by the extraction of structural and upgradeability-related features that collectively form the Digital Twin representation. These features are subsequently used to train supervised machine learning models under a leakage-safe grouped cross-validation strategy to classify proxy and non-proxy smart contracts. Finally, the constructed Digital Twin is evaluated under multiple operational scenarios to estimate heuristic security risks, while the predictive performance of the proposed framework is assessed using several complementary classification metrics.

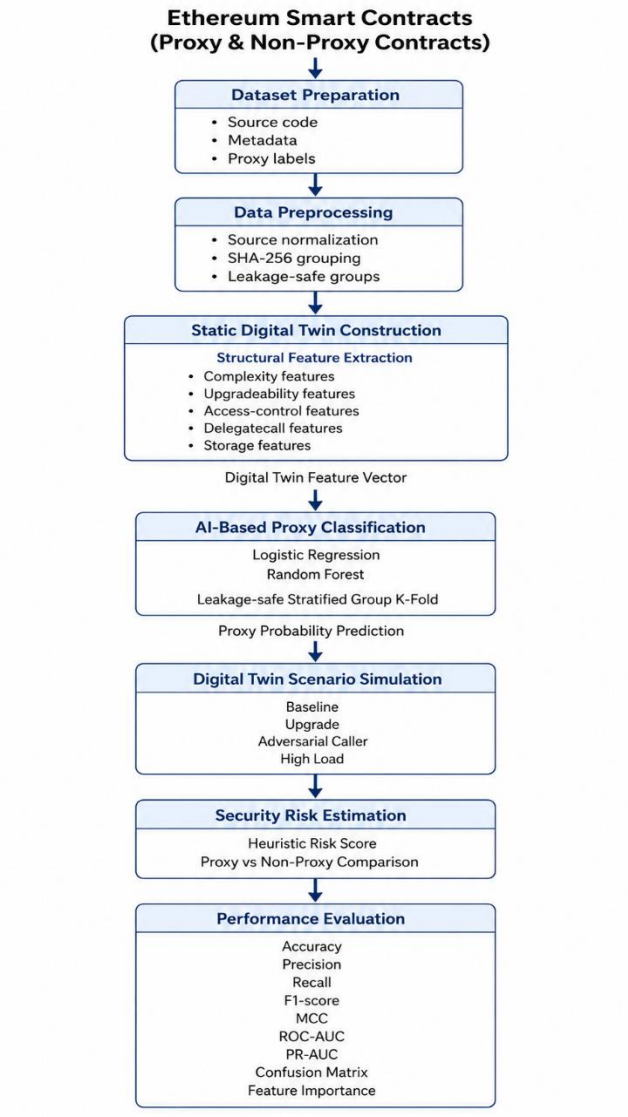



**Figure 1 Research Framework**

## Dataset Preparation

The experiments were conducted using a dataset of Ethereum smart contracts written in Solidity. Each dataset entry contains the smart contract address,

contract name, compiler version, license type, Solidity source code, normalized source code, and the corresponding proxy label. The target variable is binary, where proxy contracts are labeled as one and conventional smart contracts are labeled as zero.

Prior to feature extraction, the Solidity source code was normalized to remove formatting inconsistencies while preserving the semantic structure of the program. This preprocessing step ensured that contracts with identical logic produced identical feature representations regardless of formatting differences. To eliminate the possibility of source-code leakage during model evaluation, each normalized Solidity source code was converted into a unique source-code group identifier using the SHA-256 hashing algorithm. The grouping process is defined as:

$$G_i = \mathrm{SHA256}(C_i) \quad (1)$$

$C_i$denotes the normalized Solidity source code of the $i$-th contract and $G_i$represents its corresponding source-code group. Contracts sharing the same normalized source code therefore receive identical group identifiers and are assigned to the same cross-validation fold during model evaluation.

## Static Digital Twin Construction

The proposed Digital Twin represents each smart contract through a collection of static structural characteristics extracted directly from the Solidity source code. Unlike runtime monitoring approaches, the proposed Digital Twin relies exclusively on static program analysis to construct a virtual representation capable of supporting predictive security assessment before blockchain deployment.

Feature extraction includes general software metrics, execution characteristics, storage descriptors, upgradeability indicators, and access-control information. General software metrics include the numbers of source-code lines, functions, public functions, private functions, external functions, internal functions, payable functions, modifiers, mappings, structures, events, and inheritance relationships. Execution-related characteristics include occurrences of delegatecall, call, staticcall, callcode, inline assembly, fallback functions, receive functions, selfdestruct, require, and assert statements. Since the primary objective of this study is proxy smart contract analysis, several upgradeability-specific features are additionally extracted, including proxy-related keywords, implementation references, upgrade operations, administrator identifiers, ownership expressions, storage-slot indicators, EIP-1967 references, and access-control terms. Aggregate descriptors, including cyclomatic complexity, maximum brace depth, storage surface, upgradeability surface, external interaction surface, and access-control surface, are subsequently computed to characterize the architectural properties of each contract.

The extracted features are combined into a Digital Twin feature vector defined as:

$$DT_i = [f_1, f_2, \ldots, f_n] \quad (2)$$

$DT_i$denotes the Digital Twin representation of the $i$-th smart contract and

$f_j$represents the $j$-th extracted structural feature. This Digital Twin serves as the virtual representation used throughout both the machine learning classification process and the subsequent scenario-based security simulations.

## Leakage-Safe Proxy Smart Contract Classification

The constructed Digital Twin feature vectors are used as input for supervised machine learning to distinguish proxy and non-proxy smart contracts. Two classification algorithms were evaluated, namely Logistic Regression and Random Forest. Missing numerical values were replaced using median imputation. Feature standardization was applied to Logistic Regression using z-score normalization, whereas Random Forest was trained using the original feature values because tree-based models are inherently insensitive to feature scaling.

For Logistic Regression, the probability that a smart contract belongs to the proxy class is estimated using the sigmoid function:

$$P(y = 1 \mid x) = \frac{1}{1 + e^{-(\beta_0 + \beta^T x)}} \quad (3)$$

$x$denotes the Digital Twin feature vector, $\beta$represents the learned regression coefficients, and $P(y = 1 \mid x)$corresponds to the probability that the contract is classified as a proxy smart contract.

To obtain reliable performance estimates, this study employs leakage-safe grouped cross-validation using the Stratified Group K-Fold strategy. Unlike conventional random train-test splitting, this approach simultaneously preserves class distribution and ensures that contracts sharing identical normalized source code remain within the same validation fold. The grouping constraint is expressed as:

$$\bigcup_{k=1}^{K} G_k = G, G_i \cap G_j = \emptyset, i \neq j \quad (4)$$

$G$denotes the complete set of source-code groups and $G_k$represents the groups assigned to fold $k$. This strategy prevents structurally identical contracts from appearing simultaneously in the training and testing datasets, thereby reducing optimistic performance estimates caused by source-code leakage.

## Digital Twin Scenario Simulation

Following proxy contract classification, the constructed Digital Twin is employed to estimate scenario-dependent security risks. Rather than executing smart contracts on the Ethereum blockchain, the Digital Twin simulates alternative operational conditions by modifying the contribution of structural characteristics associated with upgradeability, delegation mechanisms, storage management, administrative control, external interaction, and software complexity.

Four simulation scenarios are considered in this study, namely Baseline, Upgrade, Adversarial Caller, and High Load. The Baseline scenario represents normal operating conditions without additional risk amplification. The Upgrade scenario increases the contribution of upgrade-related structural characteristics, including delegatecall usage, storage-slot indicators, implementation references, and administrative privileges. The Adversarial Caller scenario emphasizes privileged external interaction and access-control characteristics to

approximate malicious transaction behavior. The High Load scenario increases the contribution of computational complexity and external communication features to simulate intensive operational workloads.

For each scenario, the Digital Twin computes a heuristic security risk score using a weighted combination of structural features,

$$R_s = \sum_{i=1}^{m} w_i^{(s)} f_i \tag{5}$$

$R_s$denotes the estimated risk score under scenario $s$, $f_i$represents the extracted Digital Twin feature, and $w_i^{(s)}$denotes the scenario-specific weighting coefficient. Different operational scenarios apply different weighting factors to emphasize the structural characteristics most relevant to the simulated operating condition. The resulting heuristic scores provide relative security estimates that enable comparative risk assessment before blockchain deployment.

## Performance Evaluation

The proposed framework is evaluated using several complementary performance metrics to provide a comprehensive assessment of classification capability. The evaluation includes Accuracy, Precision, Recall, F1-score, Balanced Accuracy, Matthews Correlation Coefficient (MCC), Receiver Operating Characteristic Area Under the Curve (ROC-AUC), and Precision-Recall Area Under the Curve (PR-AUC). Because the experimental dataset is highly imbalanced, particular emphasis is placed on Recall, Balanced Accuracy, MCC, and PR-AUC, which provide more informative performance estimates than overall accuracy alone.

Accuracy is computed as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

Precision is defined as:

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

Recall is calculated as:

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

The F1-score is given by:

$$F1 = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

The Matthews Correlation Coefficient is computed as:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

To interpret model behavior, aggregated confusion matrices and permutation feature importance are also analyzed. Permutation feature importance measures the reduction in PR-AUC after randomly permuting each feature,

$$I_j = PRAUC - PRAUC_j^{perm} \quad (11)$$

$I_j$denotes the importance of feature $j$, $PRAUC$represents the original Precision-Recall Area Under the Curve, and $PRAUC_j^{perm}$denotes the PR-AUC obtained after randomly permuting feature $j$. Larger values of $I_j$indicate greater influence of the corresponding feature on classification performance.

Finally, the Digital Twin simulation results are evaluated by comparing the mean heuristic risk scores of proxy and non-proxy smart contracts across the four operational scenarios. This comparative analysis provides insight into how the proposed Digital Twin transforms static structural characteristics into scenario-dependent security assessments and supports predictive decision making before smart contract deployment.

## Results

### Dataset Characteristics and Leakage-Safe Grouping

The dataset comprised 377 Ethereum smart contracts collected from publicly available Solidity source code repositories, consisting of 360 non-proxy contracts and 17 proxy contracts. Consequently, proxy contracts accounted for only 4.51% of the entire dataset, resulting in a highly imbalanced binary classification problem that required evaluation metrics beyond overall accuracy. To reduce the risk of information leakage arising from duplicated or closely related source code, contracts were further organized into unique source-code groups based on structural similarity. This grouping process produced 146 unique groups, including 139 non-proxy groups and 7 proxy groups, indicating that multiple contracts originated from identical or highly similar implementations. Rather than performing conventional random data partitioning, all experiments employed leakage-safe grouped cross-validation, ensuring that contracts belonging to the same source-code group were assigned exclusively to either the training or testing partitions. This evaluation strategy provides a more realistic assessment of model generalization because it prevents the classifier from exploiting repeated implementation patterns that may otherwise inflate predictive performance. The distribution of smart contracts before and after the grouping process is illustrated in figure 2, highlighting the reduction from individual contracts to unique source-code groups while preserving the original class imbalance.

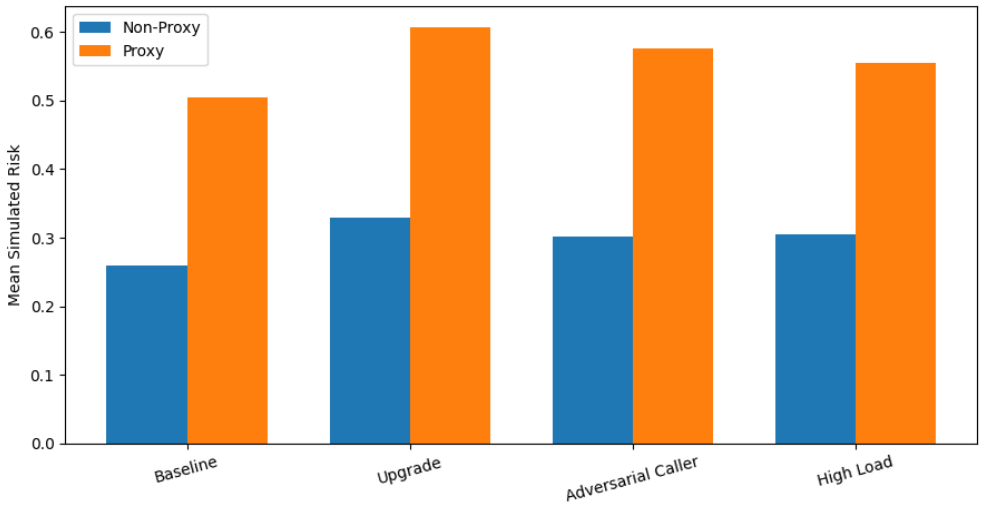



**Figure 2 Distribution of proxy and non-proxy smart contracts at the individual-contract and unique-source-code-group levels.**

## Proxy Smart Contract Detection Performance

Table 1 summarizes the classification performance of the proposed proxy smart contract detection models under the leakage-safe grouped cross-validation protocol. Both Logistic Regression and Random Forest achieved excellent overall classification performance, with accuracy values exceeding 97%. Random Forest obtained the highest overall accuracy of 0.9788, slightly outperforming Logistic Regression, which achieved an accuracy of 0.9761. However, because the dataset contained only 17 proxy contracts out of 377 smart contracts, overall accuracy alone cannot adequately represent model performance. In highly imbalanced classification problems, a classifier may achieve high accuracy by correctly predicting the majority class while failing to identify minority-class instances. Therefore, additional evaluation metrics, including Balanced Accuracy, Precision, Recall, F1-score, Matthews Correlation Coefficient (MCC), Receiver Operating Characteristic Area Under the Curve (ROC-AUC), and Precision-Recall Area Under the Curve (PR-AUC), were used to provide a more comprehensive assessment of classification performance.

As reported in table 1, Logistic Regression demonstrated superior performance across most imbalance-sensitive evaluation metrics. The model achieved a Balanced Accuracy of 0.9875, indicating excellent predictive performance across both proxy and non-proxy classes. It also obtained the highest F1-score (0.7907), MCC (0.7984), ROC-AUC (0.9980), and PR-AUC (0.9647), while achieving a Recall of 1.0000, which indicates that every proxy smart contract in the dataset was correctly identified. In contrast, Random Forest achieved a higher Precision of 0.7647, suggesting that it produced fewer false-positive predictions. However, its Recall decreased to 0.7647, meaning that several proxy contracts were incorrectly classified as non-proxy contracts. From a security perspective, false-negative predictions are generally considered more critical than false-positive predictions because undetected proxy contracts may bypass subsequent Digital Twin-based security assessment and risk simulation. Consequently, despite its slightly lower overall accuracy, Logistic Regression was selected as the primary classification model due to its superior ability to identify proxy smart contracts while maintaining excellent overall discrimination

performance.

**Table 1 Leakage-safe proxy smart contract detection performance**

| Model | Accuracy | Balanced Accuracy | Precision | Recall | F1-score | ROC-AUC | PR-AUC | MCC |
|---|---|---|---|---|---|---|---|---|
| Logistic Regression | 0.9761 | 0.9875 | 0.6538 | 1.0000 | 0.7907 | 0.9980 | 0.9647 | 0.7984 |
| Random Forest | 0.9788 | 0.8768 | 0.7647 | 0.7647 | 0.7647 | 0.9948 | 0.9146 | 0.7536 |

## Precision–Recall Analysis

Since proxy smart contracts accounted for only 4.51% of the dataset, the Precision-Recall (PR) curve provides a more appropriate evaluation than the Receiver Operating Characteristic (ROC) curve because it focuses on the classifier's ability to correctly identify the minority class while accounting for false-positive predictions. The leakage-safe Precision-Recall curves of both classification models are illustrated in figure 3. Logistic Regression achieved an average precision of 0.9647, outperforming Random Forest, which obtained an average precision of 0.9146. Both models substantially exceeded the positive-class prevalence baseline of approximately 0.045, indicating that their predictive performance was considerably better than random classification despite the severe class imbalance. Furthermore, the Precision-Recall curve of Logistic Regression maintained consistently high precision across nearly the entire recall range and successfully achieved a recall value of 1.0000, demonstrating that all proxy smart contracts were correctly detected without sacrificing overall precision. In comparison, the Precision-Recall curve of Random Forest exhibited a more noticeable decline in precision as recall increased, reflecting its lower capability to maintain stable performance when identifying proxy contracts. These results further support the selection of Logistic Regression as the primary proxy detection model for the proposed AI-Driven Digital Twin framework, particularly because maintaining high recall is essential for ensuring that proxy smart contracts are not excluded from subsequent Digital Twin-based risk simulation and security analysis.

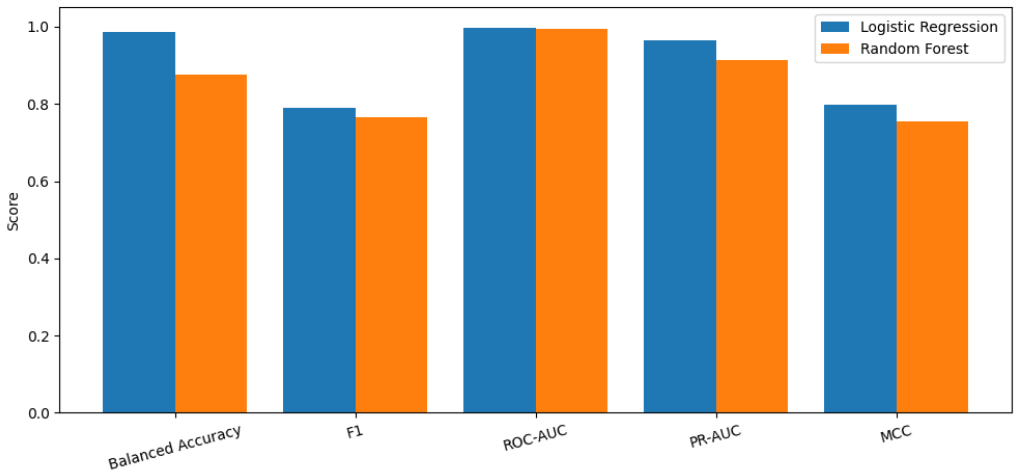



**Figure 3 Leakage-safe precision–recall curves for Logistic Regression and Random Forest. The dashed horizontal line indicates the proxy-class prevalence.**

## Confusion Matrix Analysis

The aggregated classification results of the selected Logistic Regression model under leakage-safe grouped cross-validation are summarized in figure 4 through the confusion matrix. As illustrated in the figure, the model correctly classified 351 of the 360 non-proxy smart contracts and successfully identified all 17 proxy smart contracts. Consequently, the resulting confusion matrix was $\begin{bmatrix} 351 & 9 \\ 0 & 17 \end{bmatrix}$, indicating that nine non-proxy contracts were incorrectly classified as proxy contracts, whereas no proxy contract was misclassified as a non-proxy contract. The absence of false-negative predictions is particularly important in the context of the proposed AI-Driven Digital Twin framework because every proxy smart contract can proceed to the subsequent scenario-based simulation and security assessment stages. In contrast, the presence of false-positive predictions only increases the number of contracts requiring additional analysis without excluding potentially critical proxy contracts from further evaluation. The nine false-positive classifications explain the relatively lower precision obtained by Logistic Regression compared with Random Forest. However, the model achieved perfect recall together with the highest Balanced Accuracy and Matthews Correlation Coefficient, demonstrating a better balance between sensitivity and overall classification performance. From a security perspective, maximizing the detection of proxy smart contracts is generally more desirable than minimizing false-positive predictions because overlooking an actual proxy contract may result in an incomplete Digital Twin representation and reduce the effectiveness of subsequent predictive risk analysis.

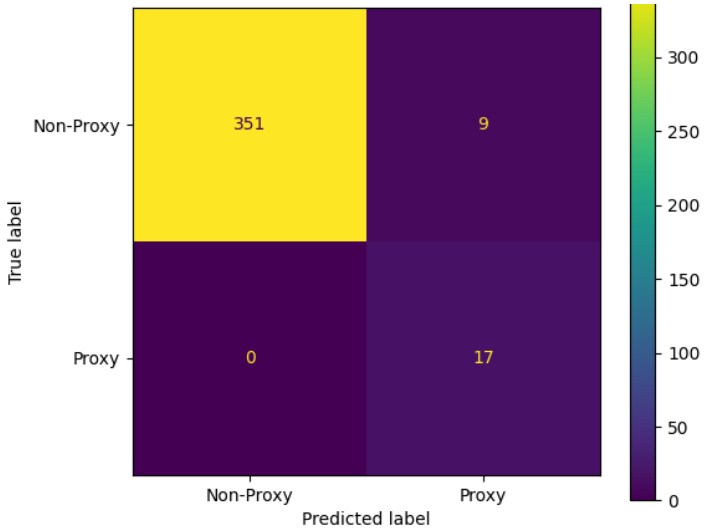



**Figure 4 Aggregated confusion matrix for Logistic Regression under leakage-safe grouped cross-validation.**

## Feature Importance

To better understand the contribution of each static Digital Twin feature to the proxy detection task, permutation feature importance was computed by measuring the decrease in Precision-Recall Area Under the Curve (PR-AUC) after randomly permuting the values of each feature while keeping the remaining features unchanged. The resulting feature importance ranking is illustrated in figure 5. Among all extracted features, n_eip1967_terms produced the largest reduction in PR-AUC after permutation, demonstrating that it was the most influential predictor for distinguishing proxy and non-proxy smart

contracts. This finding indicates that references related to the EIP-1967 proxy storage standard provide a strong discriminative signal because they are commonly associated with standardized upgradeable proxy implementations. The second most influential feature was upgradeability_intensity, although its contribution was considerably smaller than that of n_eip1967_terms. Additional upgradeability-related features, including n_admin_terms, n_upgrade_terms, and upgradeability_surface, also contributed to the classification process, but their individual effects on predictive performance were relatively modest. In contrast, general structural characteristics extracted from the Solidity source code, such as the number of functions, modifiers, source-code lines, public functions, external functions, mappings, events, and structs, produced only negligible reductions in PR-AUC when permuted, indicating that these generic code-complexity measures alone were insufficient for reliable proxy identification. Overall, the feature importance analysis suggests that explicit upgradeability indicators are substantially more informative than general structural metrics for detecting proxy smart contracts. Nevertheless, the overwhelming importance of n_eip1967_terms also indicates that the current classification model is strongly influenced by standardized proxy implementation patterns. Consequently, the ability of the proposed framework to generalize toward uncommon or customized proxy architectures that do not explicitly adopt the EIP-1967 standard remains an important direction for future investigation and broader validation.

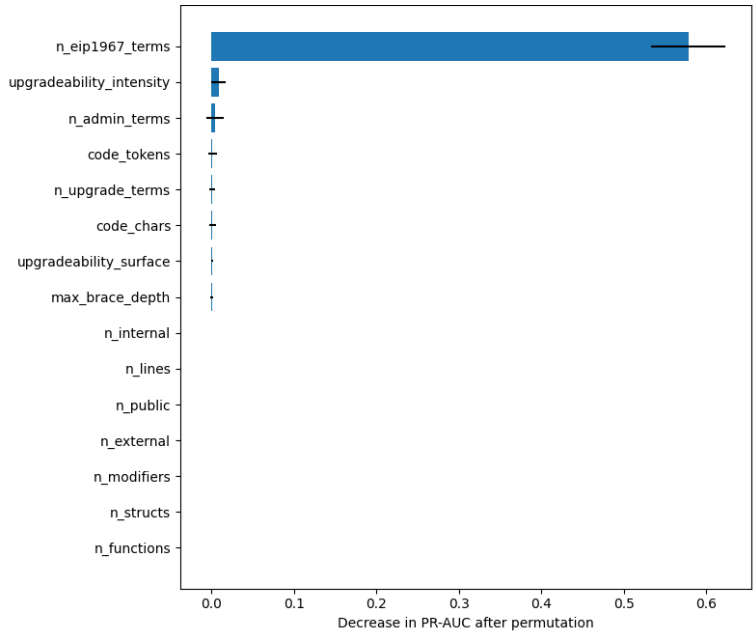



**Figure 5 Permutation feature importance measured by the decrease in PR-AUC after feature permutation.**

## Digital Twin Scenario Simulation

The proposed static Digital Twin representation was further evaluated by simulating four operational scenarios that reflect different execution conditions commonly encountered by upgradeable smart contracts, namely Baseline, Upgrade, Adversarial Caller, and High Load. Rather than executing contracts on the Ethereum network, each scenario modified the contribution of selected structural features extracted from the Solidity source code to estimate the

corresponding heuristic security risk. The resulting mean risk scores for proxy and non-proxy smart contracts are summarized in table 2, while the comparative visualization of these scenario-based risk estimates is presented in figure 6. Across all simulation scenarios, proxy smart contracts consistently exhibited higher mean risk scores than non-proxy contracts, indicating that the proposed Digital Twin representation successfully differentiated contracts with upgradeability characteristics from conventional smart contracts. Under the Baseline scenario, the average heuristic risk score was 0.50 for proxy contracts compared with 0.26 for non-proxy contracts, demonstrating that proxy-related structural characteristics alone resulted in a noticeably higher estimated security risk even before additional operational conditions were introduced.

The simulated Upgrade scenario produced the highest risk estimates for both contract classes, with mean scores increasing to 0.61 for proxy contracts and 0.33 for non-proxy contracts, as shown in table 2 and figure 6. This increase reflects the stronger influence assigned to upgrade-related indicators, administrative privileges, and proxy storage patterns within the Digital Twin simulation model. Similarly, the Adversarial Caller scenario generated a mean risk score of 0.58 for proxy contracts and 0.30 for non-proxy contracts, suggesting that contracts containing upgradeability mechanisms became more susceptible to simulated malicious external interactions. Under the High Load scenario, proxy contracts obtained an average risk score of 0.55, whereas non-proxy contracts maintained a comparatively lower mean score of 0.30. Although the High Load scenario produced lower risk values than the Upgrade and Adversarial Caller scenarios, the separation between proxy and non-proxy contracts remained consistent throughout all simulations. For proxy contracts, the overall scenario ranking was Upgrade, followed by Adversarial Caller, High Load, and Baseline, indicating that simulated upgrade operations contributed the largest increase in the estimated security risk within the proposed framework.

The scenario-based results demonstrate that the proposed Digital Twin is capable of transforming static structural characteristics extracted from Solidity source code into scenario-dependent security risk estimates without requiring blockchain deployment or runtime execution. The consistent increase in risk scores across progressively more demanding simulation scenarios suggests that the Digital Twin framework captures structural properties associated with upgradeability, administrative control, and external interaction that may influence the security posture of smart contracts. Furthermore, the clear separation between proxy and non-proxy contracts across all scenarios indicates that the extracted structural representation provides a meaningful basis for predictive security assessment. Nevertheless, the reported risk values should be interpreted as heuristic relative risk scores rather than empirically validated probabilities of vulnerabilities or successful attacks. Since the current Digital Twin simulation relies exclusively on static source-code features and predefined scenario transformations, the estimated risks have not yet been calibrated using execution traces, confirmed vulnerability datasets, or formal verification results. Therefore, while these findings demonstrate the feasibility of integrating Digital Twin simulation with AI-based proxy detection for pre-deployment security assessment, further validation using dynamic execution data and real-world vulnerability labels will be required to establish the predictive accuracy of the proposed risk estimation framework.

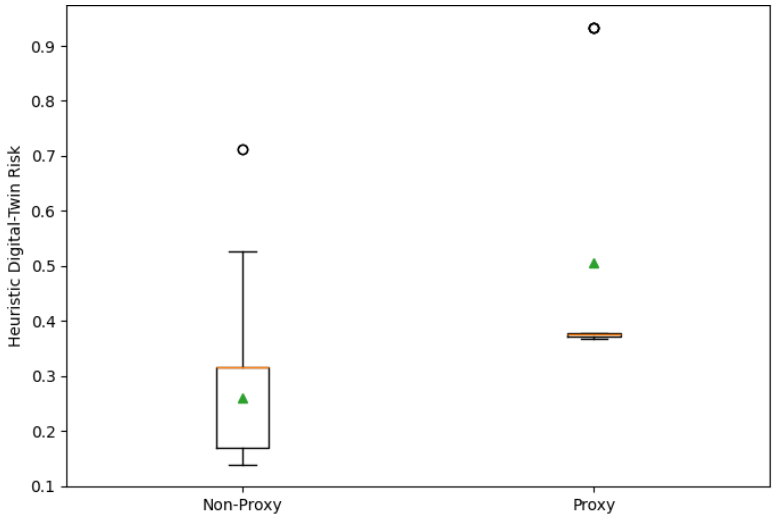



**Figure 6** **Mean heuristic Digital Twin risk scores for proxy and non-proxy contracts across the Baseline, Upgrade, Adversarial Caller, and High Load scenarios**

**Table 2** **Mean heuristic Digital Twin risk across simulation scenarios**

| Scenario | Non-Proxy | Proxy |
|---|---|---|
| Baseline | 0.26 | 0.50 |
| Upgrade | 0.33 | 0.61 |
| Adversarial Caller | 0.30 | 0.58 |
| High Load | 0.30 | 0.55 |

## Discussion

The findings of this study demonstrate that integrating leakage-safe machine learning with a Digital Twin representation provides an effective framework for predictive security analysis of upgradeable smart contracts. Although both Logistic Regression and Random Forest achieved high overall classification performance, the superiority of Logistic Regression on imbalance-sensitive metrics indicates that the proposed feature representation is capable of capturing structural characteristics associated with proxy smart contracts. The perfect recall achieved by Logistic Regression is particularly significant because every proxy contract was successfully identified, thereby ensuring that no upgradeable contract was excluded from subsequent Digital Twin analysis. From a security perspective, this characteristic is more desirable than simply maximizing overall accuracy because false-negative predictions may allow potentially critical contracts to bypass further inspection. Moreover, the adoption of leakage-safe grouped cross-validation contributes to the reliability of the reported results by preventing contracts with identical or highly similar source-code structures from being simultaneously assigned to the training and testing sets. This evaluation strategy reduces the possibility that the classifier learns duplicated implementation patterns instead of generalizable proxy characteristics, resulting in a more realistic estimation of model performance under practical deployment conditions [21].

The feature importance analysis further explains the behavior of the proposed classification model and provides insight into the structural properties of

upgradeable smart contracts. The dominance of the n_eip1967_terms feature indicates that references to the EIP-1967 storage-slot standard remain the strongest indicator for identifying proxy implementations in the current dataset. This observation is consistent with the widespread adoption of EIP-1967 in modern Ethereum upgradeable smart contracts because the standard provides a well-defined mechanism for separating proxy storage from implementation logic [22]. In contrast, general structural characteristics such as the numbers of functions, modifiers, public methods, external methods, and source-code lines contributed only marginally to classification performance. These findings suggest that upgradeability-related semantic features contain considerably more discriminative information than conventional software complexity metrics. From the perspective of Digital Twin construction, these structural indicators provide a meaningful virtual representation of contract behavior because they describe architectural properties directly related to upgradeability, administrative control, and storage management [23]. Nevertheless, the strong influence of EIP-1967-related features also indicates that the current framework may perform better for standardized proxy implementations than for customized proxy architectures employing alternative upgrade mechanisms. Therefore, future studies should extend the Digital Twin representation by incorporating additional semantic relationships, control-flow information, inheritance structures, and storage-layout characteristics to improve generalization across a wider range of smart contract designs.

The scenario-based Digital Twin simulation demonstrates that static structural representations can be transformed into predictive security assessments without requiring blockchain deployment or runtime execution. Across all simulated operating conditions, proxy smart contracts consistently exhibited higher heuristic risk scores than non-proxy contracts, while the Upgrade scenario produced the highest estimated risk among the four evaluated scenarios. This behavior is reasonable because upgrade operations introduce additional administrative privileges, delegation mechanisms, and storage-management requirements that increase the structural complexity of smart contracts and may enlarge the potential attack surface. Likewise, the Adversarial Caller scenario generated higher estimated risks by emphasizing external interactions and privileged access patterns that are commonly associated with upgradeable contracts. These observations indicate that the proposed Digital Twin is capable of simulating multiple operational conditions using only static source-code information and can therefore serve as a pre-deployment decision-support tool for developers of blockchain-based metaverse applications [24]. However, the generated risk scores should be interpreted as heuristic relative indicators rather than validated measurements of vulnerability or exploitability because the current simulation relies on predefined scenario transformations instead of execution traces or externally verified vulnerability labels. Consequently, although the proposed framework demonstrates the feasibility of integrating Artificial Intelligence with Digital Twin technology for predictive smart contract security analysis, future work should incorporate dynamic execution information, transaction histories, vulnerability datasets, symbolic execution results, or formal verification outcomes to calibrate the Digital Twin model and further improve the accuracy and practical applicability of the estimated security risk [25].

## Conclusion

This study proposed an AI-Driven Digital Twin framework for predictive smart contract security analysis in blockchain-based metaverse applications. The framework integrates leakage-safe machine learning with a static Digital Twin representation to identify upgradeable proxy smart contracts and estimate their relative security risks under multiple simulated operational scenarios. Unlike previous studies that primarily focused on binary proxy classification, the proposed approach extends smart contract analysis by incorporating scenario-based risk simulation, enabling predictive security assessment before deployment. To improve the reliability of model evaluation, leakage-safe grouped cross-validation was employed to prevent structurally similar contracts from appearing simultaneously in the training and testing datasets, thereby providing a more realistic assessment of model generalization.

The experimental results demonstrate that the proposed feature representation effectively distinguishes proxy and non-proxy smart contracts using static structural characteristics extracted from Solidity source code. Among the evaluated classifiers, Logistic Regression achieved the most suitable performance for security-oriented screening by obtaining perfect recall together with superior Balanced Accuracy, PR-AUC, and Matthews Correlation Coefficient under leakage-safe evaluation. The feature importance analysis further revealed that upgradeability-related indicators, particularly those associated with the EIP-1967 storage standard, contributed most significantly to proxy detection, while general structural metrics showed comparatively limited discriminative capability. Furthermore, the Digital Twin simulation consistently assigned higher heuristic risk scores to proxy contracts across all evaluated scenarios, with the Upgrade scenario producing the highest estimated risk. These findings indicate that the proposed Digital Twin representation is capable of transforming static source-code characteristics into meaningful scenario-dependent security assessments without requiring blockchain deployment or runtime execution.

Despite these promising results, the current framework has several limitations. The Digital Twin simulations rely on heuristic transformations of static structural features, meaning that the generated risk scores should be interpreted as relative indicators rather than empirically validated probabilities of vulnerabilities or successful attacks. In addition, the experimental dataset contains a limited number of proxy contracts and focuses exclusively on Ethereum smart contracts, which may restrict the generalizability of the proposed approach to other blockchain ecosystems and emerging proxy architectures. Future research should incorporate dynamic execution traces, transaction histories, storage-state evolution, and externally validated vulnerability datasets to improve the fidelity of the Digital Twin representation. Integrating richer semantic representations and advanced learning architectures, such as Graph Neural Networks or Transformer-based models, may further enhance predictive performance and enable more comprehensive Digital Twin-based security analysis for next-generation blockchain and metaverse applications.

## Declarations

### Author Contributions

Conceptualization: S.A.S.; Methodology: S.A.S.; Software: S.A.S.; Validation: S.A.S.; Formal Analysis: S.A.S.; Investigation: S.A.S.; Resources: S.A.S.; Data Curation: S.A.S.; Writing Original Draft Preparation: S.A.S.; Writing Review and

Editing: S.A.S.; Visualization: S.A.S.; All authors have read and agreed to the published version of the manuscript.

### Data Availability Statement

The data presented in this study are available on request from the corresponding author.


### Funding

The authors received no financial support for the research, authorship, and/or publication of this article.


### Institutional Review Board Statement

Not applicable.

### Informed Consent Statement

Not applicable.

### Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


[1] T. Huynh-The, T. R. Gadekallu, W. Wang, G. Yenduri, P. Ranaweera, Q.-V. Pham, D. B. da Costa, and M. Liyanage, "Blockchain for the metaverse: A review," *Future Generation Computer Systems*, vol. 143, no. June, pp. 401–419, 2023, doi: 10.1016/j.future.2023.02.008.

[2] P. van Vulpen, H. Heijnen, S. Mens, T. Kroon, and S. Jansen, "Upgradeable diamond smart contracts in decentralized autonomous organizations," *Frontiers in Blockchain*, vol. 7, no. December, pp. 1-13, 2024, doi: 10.3389/fbloc.2024.1481914.

[3] M. Krichen, "Strengthening the security of smart contracts through the power of artificial intelligence," *Computers*, vol. 12, no. 5, pp. 1-18, 2023, doi: 10.3390/computers12050107.

[4] F. Tao, Q. Qi, L. Wang, and A. Y. C. Nee, "Digital twins and cyber–physical systems toward smart manufacturing and Industry 4.0: Correlation and comparison," *Engineering*, vol. 5, no. 4, pp. 653–661, 2019, doi: 10.1016/j.eng.2019.01.014.

[5] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, p. 100804, 2023, doi: 10.1016/j.patter.2023.100804.

[6] H. Chu, P. Zhang, H. Dong, Y. Xiao, S. Ji, and W. Li, "A survey on smart contract vulnerabilities: Data sources, detection and repair," *Information and Software Technology*, vol. 159, no. July, p. 107221, 2023, doi: 10.1016/j.infsof.2023.107221.

[7] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on Ethereum smart contracts (SoK)," in *Principles of Security and Trust*, ser. *Lecture Notes in Computer Science*, vol. 10204. Cham, Switzerland: Springer, 2017, vol. 2017, no. March, pp. 164–186, doi: 10.1007/978-3-662-54455-6_8.

[8] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, “Making smart contracts smarter,” *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS ’16)*, Vienna, Austria, 2016, vol. 2016, no. October, pp. 254–269, doi: 10.1145/2976749.2978309.

[9] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Bünzli, and M. Vechev, “Securify: Practical security analysis of smart contracts,” *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS ’18)*, Toronto, ON, Canada, 2018, vol. 2018, no. October, pp. 67–82, doi: 10.1145/3243734.3243780.

[10] S. Kalra, S. Goel, M. Dhawan, and S. Sharma, “ZEUS: Analyzing safety of smart contracts,” *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, Feb. 2018, vol. 2018, no. February, pp. 1-15, doi: 10.14722/ndss.2018.23082.

[11] W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su, “ContractWard: Automated vulnerability detection models for Ethereum smart contracts,” *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1133–1144, Apr.–Jun. 2021, doi: 10.1109/TNSE.2020.2968505.

[12] J. Crisóstomo, F. Bação, and V. Lobo, “Machine learning methods for detecting smart contracts vulnerabilities within Ethereum blockchain: A review,” *Expert Systems with Applications*, vol. 268, no. April, p. 126353, 2025, doi: 10.1016/j.eswa.2024.126353.

[13] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, and M. Zhou, “CodeBERT: A pre-trained model for programming and natural languages,” *Findings of the Association for Computational Linguistics: EMNLP 2020*, vol. 2020, no. November, pp. 1536–1547, doi: 10.18653/v1/2020.findings-emnlp.139.

[14] Y. Zhuang, Z. Liu, P. Qian, Q. Liu, X. Wang, and Q. He, “Smart contract vulnerability detection using graph neural network,” *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, 2020, pp. 3283–3290, doi: 10.24963/ijcai.2020/454.

[15] M. Salehi, J. Clark, and M. Mannan, “Not so immutable: Upgradeability of smart contracts on Ethereum,” in *Financial Cryptography and Data Security*, ser. *Lecture Notes in Computer Science*, vol. 13412. Cham, Switzerland: Springer, 2023, vol. 2023, no. July, pp. 539–554, doi: 10.1007/978-3-031-32415-4_33.

[16] X. Li, J. Yang, J. Chen, Y. Tang, and X. Gao, “Characterizing Ethereum upgradable smart contracts and their security implications,” *Proceedings of the ACM Web Conference 2024 (WWW ’24)*, Singapore, 2024, vol. 2024, no. May, pp. 1847–1858, doi: 10.1145/3589334.3645640.

[17] A. M. Ebrahimi, B. Adams, G. A. Oliva, and A. E. Hassan, “A large-scale exploratory study on the proxy pattern in Ethereum,” *Empirical Software Engineering*, vol. 29, no. 4, p. 81, 2024, doi: 10.1007/s10664-024-10485-1.

[18] A. Onwubiko, R. Singh, S. M. Awan, Z. Pervez, and N. Ramzan, “Enabling trust and security in digital twin management: A blockchain-based approach with Ethereum and IPFS,” *Sensors*, vol. 23, no. 14, p. 6641, 2023, doi: 10.3390/s23146641.

[19] F. Corradini, A. Marcelletti, A. Morichetta, B. Re, and L. Ruschioni, “A digital twin approach for blockchain smart contracts,” *Proceedings of the 2024 IEEE International Conference on Software Analysis, Evolution and Reengineering*

*Companion (SANER-C)*, Rovaniemi, Finland, 2024, pp. 1–11, doi: 10.1109/SANER-C62648.2024.00007.

[20] M. Allamanis, "The adverse effects of code duplication in machine learning models of code," *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward! 2019)*, Athens, Greece, 2019, vol. 2019, no. October, pp. 143–153, doi: 10.1145/3359591.3359735.

[21] M. A. Lones, "Avoiding common machine learning pitfalls," *Patterns*, vol. 5, no. 10, p. 101046, 2024, doi: 10.1016/j.patter.2024.101046.

[22] H. Rameder, M. di Angelo, and G. Salzer, "Review of automated vulnerability analysis of smart contracts on Ethereum," *Frontiers in Blockchain*, vol. 5, no. March, pp. 1-20, 2022, doi: 10.3389/fbloc.2022.814977.

[23] R. Sahal, S. H. Alsamhi, K. N. Brown, D. O'Shea, C. McCarthy, and M. Guizani, "Blockchain-empowered digital twins collaboration: Smart transportation use case," *Machines*, vol. 9, no. 9, p. 193, 2021, doi: 10.3390/machines9090193.

[24] S. Boschert and R. Rosen, "Digital twin—The simulation aspect," in *Mechatronic Futures: Challenges and Solutions for Mechatronic Systems and Their Designers*, P. Hehenberger and D. Bradley, Eds. Cham, Switzerland: Springer, 2016, vol. 2016, no. June, pp. 59–74, doi: 10.1007/978-3-319-32156-1_5.

[25] Y. Huang, X. Wu, Q. Wang, Z. Qian, X. Chen, M. Tang, and Z. Zheng, "The sword of Damocles: Upgradeable smart contract in Ethereum," *Proceedings of the 2024 IEEE/ACM 32nd International Conference on Program Comprehension (ICPC)*, Lisbon, Portugal, 2024, vol. 2024, no. June, pp. 333–345, doi: 10.1145/3643916.3644426.