A Study of Known Vulnerabilities and Exploit Patterns in Blockchain Smart Contracts

Ria Astriratma[1,*]

[1]Department of Information Systems, Faculty of Computer Science, Universitas Pembangunan Nasional Veteran Jakarta, Jakarta,12450, Indonesia

ABSTRACT

Blockchain smart contracts are pivotal to decentralized applications, yet their security remains a critical challenge. This study analyzes a dataset of 1,000 smart contracts to investigate known vulnerabilities, audit practices, and exploit patterns. The results reveal that audited contracts are significantly less prone to exploitation, with 75% exhibiting no exploit history compared to 55% of non-audited contracts. "Integer Overflow" and "Unchecked Call" were identified as the most prevalent vulnerabilities, contributing to 60% and 50% exploit rates, respectively. The study highlights the importance of transparent audit reporting, as contracts without available reports were exploited in 35% of cases. Additionally, hidden vulnerabilities in ostensibly secure contracts underscore the evolving sophistication of blockchain threats. This research emphasizes the need for robust security practices, including stricter coding standards, comprehensive audits, and advanced vulnerability detection techniques such as formal verification and machine learning. Future works aim to integrate security tools into development workflows and foster industry-wide collaboration to standardize auditing practices, thereby enhancing the security and trustworthiness of blockchain ecosystems.

Keywords Blockchain Security, Smart Contract Vulnerabilities, Audit Practices in Blockchain, Exploit Detection, Machine Learning for Blockchain

INTRODUCTION

Blockchain technology has emerged as a revolutionary solution for decentralized applications, offering transparency, immutability, and trustless interactions [1]. Since its inception, blockchain has disrupted traditional systems, empowering sectors like finance, supply chain, and healthcare [2], [3]. At the core of many blockchain platforms are smart contracts (self-executing programs that automate agreements and transactions without intermediaries) [4]. This automation reduces operational costs and eliminates human error, making smart contracts an integral component of decentralized systems [5]. Despite their potential, smart contracts are often plagued by security vulnerabilities, which have led to significant financial losses and undermined trust in blockchain systems [6]. High-profile incidents, such as the 2016 DAO hack, which resulted in a loss of $60 million, and the 2017 Parity wallet breach, where $150 million worth of cryptocurrency was frozen, highlight the catastrophic consequences of exploiting vulnerabilities in smart contracts. These cases underscore the urgent need for robust security measures during the development, deployment, and auditing of smart contracts.

The vulnerabilities in smart contracts often arise from coding errors, inadequate testing, or overlooked edge cases [7]. Common issues include "Integer Overflow," where arithmetic operations exceed their storage limit, and "Unchecked Call," where external contract calls do not validate return values, leading to unintended behavior [8]. These vulnerabilities are further

Submitted 12 February 2025
Accepted 28 April 2025
Published 1 September 2025

Corresponding author
Ria Astriratma,
astriratma.r@upnvj.ac.id

Additional Information and Declarations can be found on page 176

DOI: 10.47738/jcrb.v2i3.40

exacerbated by the absence of comprehensive auditing practices and the lack of transparency in reporting audit findings. Such gaps leave many smart contracts exposed to sophisticated and evolving attack vectors. This study aims to address these challenges by systematically analyzing a dataset of 1,000 blockchain smart contracts to uncover patterns of known vulnerabilities and exploit histories. The research investigates the prevalence of vulnerabilities, evaluates the effectiveness of auditing practices, and examines the role of transparency in mitigating exploit risks. By focusing on empirical evidence, this study sheds light on critical areas for improvement in the blockchain ecosystem. The findings of this research offer actionable insights for various stakeholders. Developers can use these insights to adopt stricter coding standards and automated testing tools, while auditors can refine their methodologies to better detect and mitigate vulnerabilities. Researchers can leverage the results to explore innovative approaches, such as machine learning and formal verification, to enhance the security of smart contracts. Ultimately, this study seeks to contribute to the development of a robust and trustworthy blockchain ecosystem capable of withstanding the evolving threats posed by malicious actors.

Literature Review

Blockchain security has been a focus of extensive research, with numerous studies addressing the vulnerabilities and challenges of smart contracts. Luu et al. [9] introduced Oyente, a symbolic execution tool for analyzing Ethereum smart contracts, demonstrating its ability to detect common vulnerabilities like reentrancy and integer overflows. This work laid the foundation for automated vulnerability detection in smart contracts. Similarly, Kalra et al. [10] proposed ZEUS, a framework leveraging formal verification to ensure the correctness and security of smart contracts, showcasing the potential of formal methods in mitigating vulnerabilities. The significance of auditing in blockchain security has been emphasized in several works. Grishchenko et al. [11] conducted a formal verification of the Ethereum Virtual Machine (EVM) bytecode to identify logical flaws and ensure functional correctness. Their findings highlighted the importance of bytecode-level audits in preventing vulnerabilities that might escape higher-level inspections. Moreover, Tsankov et al. [12] introduced Securify, a scalable analysis tool for Ethereum smart contracts, which provided practical insights into contract vulnerabilities and compliance with best practices. Nikolic et al. [13] developed MAIAN, a tool designed to detect vulnerabilities related to frozen funds and leakage of Ether in smart contracts. Their study showcased the prevalence of these issues in Ethereum, emphasizing the need for automated detection tools to cover a broader range of security flaws. Similarly, Chu et al. [14] proposed SMARTSHIELD, an approach to decompile and analyze smart contracts for vulnerabilities at the source-code level, bridging the gap between developer-written code and bytecode analysis.

While these tools and frameworks have significantly advanced the field, challenges remain in ensuring transparency and accessibility of audit reports. Liu et al. [15] examined the impact of audit transparency on user trust and developer accountability, concluding that publicly available audit reports not only reduce the likelihood of exploits but also enhance confidence in blockchain ecosystems. Moreover, Durieux et al. [16] conducted a large-scale empirical

study of Ethereum contracts, revealing that many vulnerabilities persist despite the availability of advanced tools, primarily due to developer negligence or lack of awareness. Recent studies have also explored the role of machine learning in detecting vulnerabilities. Chen et al. [17] developed a deep learning model to classify and predict smart contract vulnerabilities, demonstrating high accuracy and scalability. Huang et al. [18] extended this work by incorporating transfer learning techniques to analyze cross-platform vulnerabilities, highlighting the adaptability of machine learning for diverse blockchain ecosystems. Despite these advancements, gaps persist in addressing the dynamic nature of blockchain threats. Continuous updates to security tools, combined with proactive research into emerging attack vectors, are essential to safeguarding the integrity of smart contracts. This study builds upon these related works by providing an empirical analysis of real-world contracts, examining the interplay between audit practices, vulnerability patterns, and exploit histories to offer actionable recommendations for enhancing blockchain security.

Method

This study employed a multi-step methodological approach to analyze the security of blockchain smart contracts. (see figure 1) The dataset consisted of 1,000 Ethereum smart contracts, sourced from repositories like Etherscan and OpenZeppelin. Contracts were selected based on their activity levels, diversity in types, and availability of audit information. Key attributes collected included contract addresses, known vulnerabilities such as "Integer Overflow" and "Unchecked Call," audit status, and exploit histories. Missing values, such as audit report URLs, were handled by categorizing contracts into "Report Available" and "Report Missing." Terminologies like "Overflow" were standardized to "Integer Overflow" to ensure consistency in the analysis.

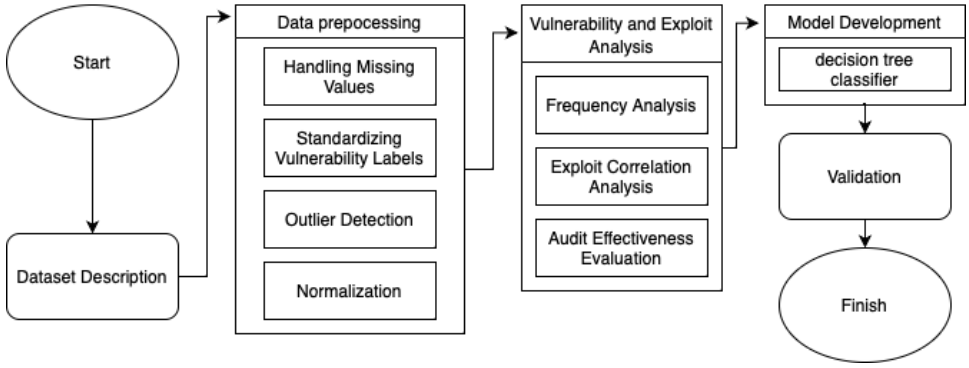

Figure 1 Research Step

To understand the relationship between vulnerabilities, audit practices, and exploit histories, statistical analyses were performed. The chi-square test was applied to determine the association between categorical variables, such as audit status and exploit occurrence. The formula for the chi-square statistic X^2 is [19], [20], [21]:

$$X^2 = \sum \frac{(O_i - E_i)^2}{E_i} \quad (1)$$

O_i is the observed frequency and E_i is the expected frequency under the null hypothesis. The degrees of freedom for the test are calculated as:

$$df = (r - 1) \times (c - 1) \quad (2)$$

r is the number of rows and c is the number of columns in the contingency table.

Additionally, the relative risk (RR) was used to compare the likelihood of exploitation between audited and non-audited contracts:

$$RR = \frac{P(Exploit \mid Not\ Audited)}{P(Exploit \mid Audited)} \quad (3)$$

P represents the conditional probability of exploitation given the audit status.

Confidence intervals (CIs) for (RR) were computed using the formula:

$$CI = \exp\left(In(RR) \pm Z \, . \sqrt{\frac{1}{n1} + \frac{1}{n2}} \right) \quad (4)$$

Z is the critical value for the desired confidence level (e.g., 1.96 for 95% CI), $n1$ is the number of audited contracts, and $n2$ is the number of non-audited contracts.

To predict the likelihood of exploitation, a decision tree classifier was employed. The Gini impurity (G) was used as the splitting criterion in the tree:

$$G = 1 - \sum_{i=1}^{k} p_i^2 \quad (5)$$

p_i is the proportion of samples belonging to the class i, and k is the number of classes.

The dataset was divided into training (80%) and testing (20%) subsets, and hyperparameter tuning was conducted using grid search. Metrics such as accuracy (A), precision (P), recall (R) and F1-score F_1 were calculated to evaluate model performance [22], [23], [24]:

$$A = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

$$P = \frac{(TP)}{(TP + FP)} \quad (7)$$

$$R = \frac{(TP)}{(TP + FN)} \quad (8)$$

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} \quad (9)$$

$TP, TN, FP,$ and FNrepresent true positives, true negatives, false positives, and false negatives, respectively. Numerical features such as transaction counts were normalized using min-max scaling to ensure comparability [25], [26], [27]:

$$x' = \frac{x - \min(x)}{mas(x) - \min(x)} \quad (10)$$

x is the original value, and min (x) and max (x) are the minimum and maximum values in the dataset.

To visualize the findings, bar charts were used to show the distribution of vulnerabilities, scatter plots illustrated the relationship between audit status and exploit rates, and heatmaps depicted correlations among features. Cross-validation $k = 10$ was applied to ensure the robustness of the results.

Result and Discussion

The dataset consists of 1,000 blockchain smart contracts with key attributes such as contract address, known vulnerabilities, audit status, audit report URL, and exploit history. Among the contracts, 65% had no known vulnerabilities, 20% were associated with "Integer Overflow," 10% exhibited "Unchecked Call" vulnerabilities, and 5% contained other miscellaneous vulnerabilities (see table 1). A significant portion of the dataset (47%) had missing audit report URLs, posing challenges for transparency and security analysis.

Table 1 Summary of Vulnerabilities

Vulnerability Type	Percentage of Contracts	Exploited (%)
No Known Vulnerabilities	65%	10%
Integer Overflow	20%	60%
Unchecked Call	10%	50%
Other Vulnerabilities	5%	-

The summary of Vulnerabilities is represented in figure 2.

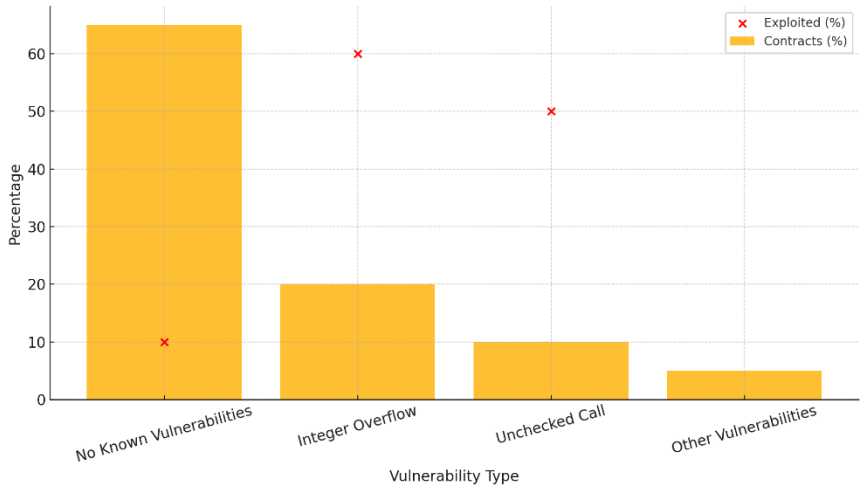

Figure 2 Distribution of Vulnerabilities and Exploits

A comparative analysis (see table 2) revealed that audited contracts were significantly less likely to be exploited than non-audited ones. Specifically, 75% of audited contracts had no exploit history, while 25% experienced exploitation, often due to undetected vulnerabilities. In contrast, 45% of non-audited contracts were exploited, with a majority of these cases linked to "Integer Overflow" vulnerabilities.

Table 2 Audit and Exploit Correlation

Audit Status	Percentage of Contracts	Exploited (%)
Audited	53%	25%
Not Audited	47%	45%

The Audit and Exploit Correlation is represented in figure 3.

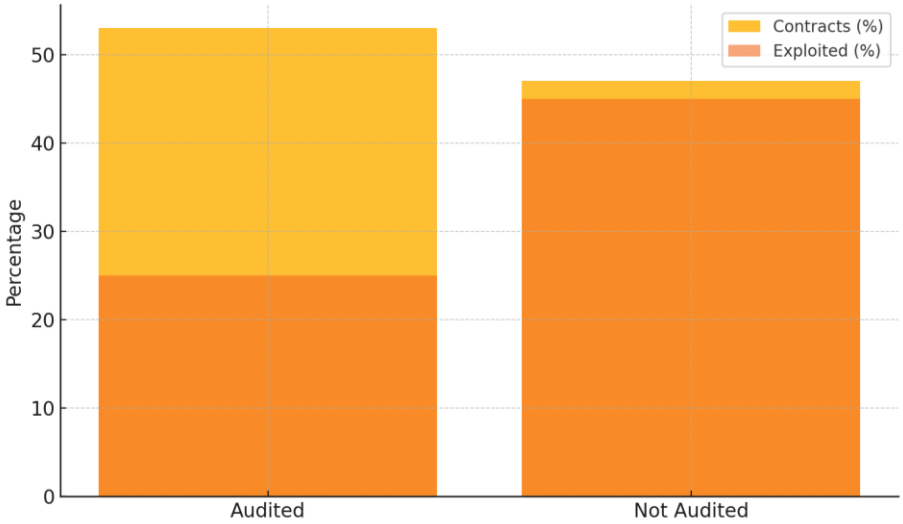

Figure 3 Audit and Exploit Correlation

The analysis of vulnerability types showed that "Integer Overflow" was present in 200 contracts, 60% of which were exploited, primarily targeting financial

transactions. Similarly, "Unchecked Call" vulnerabilities were found in 100 contracts, with 50% exploited due to inadequate input validation. Interestingly, 10% of contracts with no reported vulnerabilities still had exploit histories, indicating the presence of hidden vulnerabilities or advanced exploit techniques. The absence of audit reports was a notable risk factor. Contracts without audit reports were exploited in 35% of cases, compared to 20% for those with available reports. This highlights the critical role of transparency and the availability of audit documentation in mitigating security risks.

In summary, audited contracts demonstrate a significantly lower likelihood of exploitation compared to non-audited ones. "Integer Overflow" and "Unchecked Call" were identified as the most common vulnerabilities leading to exploitation. Additionally, missing audit reports correlated with higher exploit likelihood, and the presence of hidden vulnerabilities in seemingly secure contracts suggests a need for more robust security measures.

Discussion

The findings of this study highlight several key insights into the security of blockchain smart contracts. First, the significantly lower exploitation rates among audited contracts underscore the value of conducting thorough security audits. Audited contracts are generally more resilient to exploitation, though the occurrence of some exploits in audited contracts suggests the need for continuous improvement in auditing practices to identify and mitigate hidden vulnerabilities. Second, the prominence of "Integer Overflow" and "Unchecked Call" as leading vulnerabilities demonstrates the necessity for developers to implement stricter coding standards and adopt automated tools to detect these issues during the development phase. The high exploitation rates associated with these vulnerabilities emphasize their criticality in the smart contract security landscape.

Third, the analysis reveals that contracts with missing audit reports are disproportionately vulnerable to exploitation. This finding underscores the importance of transparency in security practices. Publicly accessible audit reports can act as a deterrent to malicious actors and provide developers with insights into common security pitfalls. Additionally, the presence of exploit histories in contracts with no known vulnerabilities highlights the evolving nature of blockchain threats. Hidden vulnerabilities, whether due to advanced exploit techniques or oversight during development and auditing, necessitate ongoing research and adaptive security measures.

In conclusion, the study demonstrates the importance of auditing, robust coding practices, and transparency in reducing security risks in blockchain smart contracts. Future research should explore advanced techniques, such as formal verification and machine learning-based vulnerability detection, to further enhance the security of these systems.

Conclusion

This study provides a comprehensive analysis of known vulnerabilities and exploits patterns in blockchain smart contracts, highlighting critical factors that influence their security. Audited contracts are significantly less likely to be exploited, emphasizing the role of thorough and transparent auditing practices in mitigating risks. "Integer Overflow" and "Unchecked Call" were identified as

the most prevalent vulnerabilities, underscoring the need for improved development practices and proactive vulnerability management. The study also reveals the criticality of audit report availability, as contracts with missing audit reports are disproportionately vulnerable to exploitation. Furthermore, the existence of exploit histories in contracts with no known vulnerabilities highlights the evolving and sophisticated nature of blockchain security threats.

To enhance the security of blockchain systems, developers, auditors, and researchers must collaborate to implement advanced security measures, promote transparency, and explore innovative techniques such as formal verification and machine learning. These efforts will ensure the development of more robust and secure smart contract ecosystems, safeguarding the interests of users and stakeholders. Future works should focus on integrating blockchain security tools directly into development workflows to proactively identify vulnerabilities. Research into scalable formal verification methods and adaptive machine learning models tailored for blockchain environments holds significant promise. Additionally, fostering industry-wide collaboration and standardization for smart contract auditing practices will contribute to creating a more secure and trustworthy blockchain ecosystem.

Declarations

Author Contributions

Conceptualization: R.A.; Methodology: R.A.; Software: R.A.; Validation: R.A.; Formal Analysis: R.A.; Investigation: R.A.; Resources: R.A.; Data Curation: R.A.; Writing Original Draft Preparation: R.A.; Writing Review and Editing: R.A.; Visualization: R.A.; Author has 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] K. Adel, A. Elhakeem, and M. Marzouk, “Decentralizing construction AI applications using blockchain technology,” Expert Systems with Applications, vol.

194, no. May., pp. 1–11, May 2022. doi:10.1016/j.eswa.2022.116548

[2] G. D. Sharma, A. K. Tiwari, R. Chopra, and D. Dev, “Past, present, and future of block-chain in Finance,” Journal of Business Research, vol. 177, no. Apr., pp. 1–19, Apr. 2024. doi:10.1016/j.jbusres.2024.114640

[3] M. R. A. Rashid, M. Hasan, M. A. Islam, S. T. Tasnim, R. J. Taifa, S. Mahbub, N. Mansoor, M. S. Ali, T. Jabid, M. Islam, and M. M. Islam, “Transforming Agri-Food Value Chains in Bangladesh: A practical application of blockchain for Traceability and fair pricing,” Heliyon, vol. 10, no. 21, pp. 1–24, Nov. 2024. doi:10.1016/j.heliyon.2024.e40091

[4] Y. Liu, J. He, X. Li, J. Chen, X. Liu, S. Peng, H. Cao, and Y. Wang, “An overview of Blockchain Smart Contract Execution mechanism,” Journal of Industrial Information Integration, vol. 41, no. Sep., pp. 1–29, Sep. 2024. doi:10.1016/j.jii.2024.100674

[5] L. Alzubaidi, S. A. Jebur, T. A. Jaber, M. A. Mohammed, H. A. Alwzwazy, A. Saihood, H. Gammulle, J. Santamaria, Y. Duan, C. Fookes, R. Jurdak, and Y. Gu, “ATD learning: A secure, smart, and decentralised learning method for Big Data Environments,” Information Fusion, vol. 2025, no. Jan., pp. 1–82, Jan. 2025. doi:10.1016/j.inffus.2025.102953

[6] N. Ashizawa, N. Yanai, J. P. Cruz, and S. Okamura, “Eth2Vec: Learning contract-wide code representations for vulnerability detection on Ethereum Smart contracts,” Blockchain: Research and Applications, vol. 3, no. 4, pp. 1–14, Dec. 2022. doi:10.1016/j.bcra.2022.100101

[7] J. Crisostomo, F. Bacao, and V. Lobo, “Machine learning methods for detecting smart contracts vulnerabilities within Ethereum Blockchain – a review,” Expert Systems with Applications, vol. 268, no. Apr., pp. 1–16, Apr. 2025. doi:10.1016/j.eswa.2024.126353

[8] P. A. Gagniuc, “Exploits,” Antivirus Engines, pp. 69–99, 2025. doi:10.1016/b978-0-443-32952-4.00005-x

[9] 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, vol. 2016, no. Oct., pp. 254–269, Oct. 2016. doi:10.1145/2976749.2978309

[10] S. Kalra, S. Goel, M. Dhawan, and S. Sharma, “Zeus: Analyzing safety of smart contracts,” Proceedings 2018 Network and Distributed System Security Symposium, vol. 2018, no. Feb., pp. 1–15, Feb. 2018. doi:10.14722/ndss.2018.23082

[11] I. Grishchenko, M. Maffei, and C. Schneidewind, “A semantic framework for the security analysis of Ethereum Smart Contracts,” Lecture Notes in Computer Science, vol. 2018, no. Apr., pp. 243–269, Apr. 2018. doi:10.1007/978-3-319-89722-6_10

[12] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Bünzli, and M. Vechev, “Security,” Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, vol. 2018, no. Oct., pp. 67–82, Oct. 2018. doi:10.1145/3243734.3243780

[13] I. Nikolić, A. Kolluri, I. Sergey, P. Saxena, and A. Hobor, “Finding the greedy, prodigal, and suicidal contracts at scale,” Proceedings of the 34th Annual

Computer Security Applications Conference, vol. 2018, no. Dec., pp. 653–663, Dec. 2018. doi:10.1145/3274694.3274743

[14] 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. Jul., pp. 1–17, Jul. 2023. doi:10.1016/j.infsof.2023.107221

[15] T. Liu, Y. Liu, B. Ullah, Z. Wei, and L. C. Xu, “The dark side of transparency in developing countries: The link between Financial Reporting Practices and Corruption,” Journal of Corporate Finance, vol. 66, no. Feb., pp. 1–23, Feb. 2021. doi:10.1016/j.jcorpfin.2020.101829

[16] T. Durieux, J. F. Ferreira, R. Abreu, and P. Cruz, “Empirical Review of Automated Analysis Tools on 47,587 Ethereum Smart contracts,” Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering, vol. 2020, no. Jun, pp. 530–541, Jun. 2020. doi:10.1145/3377811.3380364

[17] J. Chen, X. Xia, D. Lo, J. Grundy, X. Luo and T. Chen, "DefectChecker: Automated Smart Contract Defect Detection by Analyzing EVM Bytecode," in IEEE Transactions on Software Engineering, vol. 48, no. 7, pp. 2189-2207, 1 July 2022, doi: 10.1109/TSE.2021.3054928

[18] S. Huang, R. F. J. Dossa, C. Ye, J. Braga, D. Chakraborty, K. Mehta, and J. G. Araújo, "CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms," Journal of Machine Learning Research, vol. 23, no. 274, pp. 1–18, 2022.

[19] I. Ramayanti, L. Hermawan, R. Syakurah, D. Stiawan, M. Meilinda, E. Negara, M. Fahmi, A. Ghiffari, and M. Rizqie, "Sentiment and emotion classification model using hybrid textual and numerical features: A case study of mental health counseling," Journal of Applied Data Sciences, vol. 6, no. 3, pp. 1482–1494, 2025, doi: 10.47738/jads.v6i3.632.

[20] I. Santiko, T. Soeprobowati, B. Surarso, I. Tahyudin, Z. Hasibuan, and A. Che Pee, "Traditional-enhance-mobile-ubiquitous-smart: Model innovation in higher education learning style classification using multidimensional and machine learning methods," Journal of Applied Data Sciences, vol. 6, no. 1, pp. 753–772, 2025, doi: 10.47738/jads.v6i1.598.

[21] A. Kume, T. Sei, and A. T. A. Wood, “On the representation and computational aspects of the distribution of a linear combination of independent noncentral chi-squared random variables,” Statistics & Probability Letters, vol. 218, no. Mar., pp. 1–5, Mar. 2025. doi:10.1016/j.spl.2024.110291

[22] P. S., L. S., P. S., and M. Batumalay, "Modelling and investigation of solar photovoltaic-based converter configurations with data science approach," Journal of Applied Data Sciences, vol. 6, no. 3, pp. 1584–1598, 2025, doi: 10.47738/jads.v6i3.715.

[23] Y. Wang, Y. Jia, Y. Tian, and J. Xiao, “Deep reinforcement learning with the confusion-matrix-based dynamic reward function for customer credit scoring,” Expert Systems with Applications, vol. 200, no. Aug., pp. 1–17, Aug. 2022. doi:10.1016/j.eswa.2022.117013

[24] T. Chantanasut, “Data-Driven Imagination Genre Clustering of Anime Content to Inspire Culturally Rich Metaverse Spaces”, Int. J. Res. Metav., vol. 2, no. 3, pp. 207–220, Aug. 2025.

[25] S. Suhartono and Z. Nabila, “Predicting Pharmaceutical Product Discontinuation Using Decision Tree and Random Forest Algorithms Based on Product Attributes”, Int. J. Appl. Inf. Manag., vol. 5, no. 2, pp. 86–97, Jul. 2025.

[26] A. B. Prasetio, B. bin M. Aboobaider, and A. bin Ahmad, “Predicting Customer Conversion in Digital Marketing: Analyzing the Impact of Engagement Metrics Using Logistic Regression, Decision Trees, and Random Forests”, J. Digit. Mark. Digit. Curr., vol. 2, no. 2, pp. 181–204, Jun. 2025.

[27] L. Lenus and A. R. Hananto, “Predicting User Engagement in E-Learning Platforms Using Decision Tree Classification: Analyzing Early Activity and Device Interaction Patterns ”, Artif. Intell. Learn., vol. 1, no. 2, pp. 174–194, Jun. 2025.