ISSN: 2633-4828 Vol. 5 No.4, December, 2023

International Journal of Applied Engineering & Technology

SMART CONTRACT VULNERABILITIES AND DETECTION USING BLOCK CHAIN TECHNOLOGY

Sangeetha R[1] and Dr. Veena M N[2]
[1]Assistant Professor, CIST, ManasaGangortri, University of Mysore, Mysuru-570006
[2]Professor, Department of MCA, PES college of Engineering, Shankar Gowda road, Mandya-571401
[1]sangeethauom9@gmail.com and [2]veenadisha1@gmail.com
[1]Orcid Id: 0009-0004-1452-0549 and **[2]Orcid Id:** 0009-0004-5953-1065

ABSTRACT
Smart contracts play an important role in block-chain technology, but flaws in programming may result in significant security threats. As block-chain technology advances, smart contracts, an integral component, have attracted significant attention. However, security worries within smart contracts have become more apparent. Although algorithms based on machine learning demonstrate potential in the field of smart-contract security detection, there's a yet a scarcity of extensive research. To plug this research gap, this work suggests a unique analysis of smart-contract vulnerability identification using machine learning using block chain techniques. Developers rely on high-level languages for developing intricate company logic in smart contracts. A block-chain-based smart contract is crucial for decentralized applications, however it is prone to attacks. So, the presence of hazards must be tackled as a priority. To ensure financial stability, smart contracts must be checked for deficiencies before being implemented and connected to applications. The current research explores the use of deep learning for building safe, bug-free smart contracts. The purpose of the research paper is to determine three varieties of distinct kinds of vulnerabilities: reentrancy, timestamp, and infinite loop. A deep learning model for detecting smart contract vulnerabilities has been built via graph neural networks.

I. INTRODUCTION
Smart contracts make use of block-chain technology to execute computer programs self-sufficiently. This kind of contract, written as code on the block-chain, executes and enforces terms based on established standards and conditions. Smart contracts, also known as contract accounts, have balances and may facilitate transactions. Smart contracts, unlike human operations, are set up on the network and run as programs; they cannot be managed by humans. A smart contract's code is unchangeable once it is officially deployed, hence programming errors might result in permanent damages. Block-chain is a novel innovation that lets users connect without relying on others. It transforms how commercial organizations interact with one another without requiring a reliable third party[1]. The digital ledger technology known as block-chain is the foundation of cryptocurrencies such as Bitcoin. Basically, it's a distributed and decentralized ledger that keeps track of transactions across multiple systems. This assures that the record cannot be modified in the recent past without affecting all subsequently blocks and the network's consensus. Block-chain technology has evolved rapidly over the last ten years, not solely in financial services but also in a variety of other fields as well, including supply chain management, the Internet of Things (IoT), retail, healthcare, gaming, and communication, just to name a few. By 2020, block-chain technology had progressively driven global economic expansion, and by 2030, it is expected to boost global GDP by 1.76 trillion dollars by enhancing traceability and trust. Due to the intricacy of smart contracts and the decentralized nature of block-chain, the use of typical software security solutions in smart contracts is limited [1,2]. The block chain characteristics are shown in Fig 1.

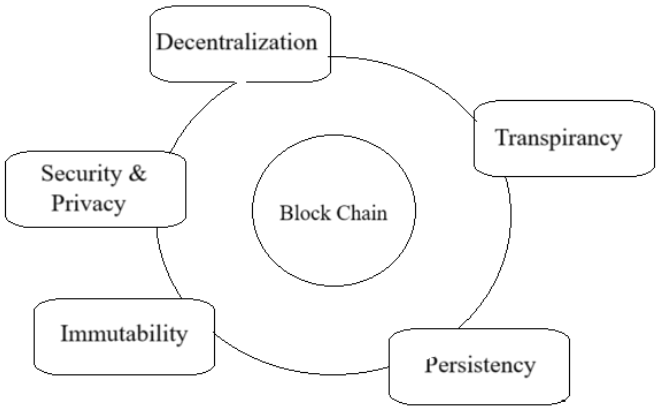


Fig 1. Block Chain Characterisitcs

As the consequence, smart contract vulnerability detection technology has attracted significant interest. Existing machine-learning-based smart contract security detection has surpassed a particular stage. However, in the existing literature, detailed studies on the use of machine learning in smart-contract security detection are scarce. To address this gap, develop a review paper outlining the present status, problems, and potential development trends of machine-learning algorithms in smart-contract security detection. This will serve as a complete and systematic reference for academics and practitioners worldwide.

II. RELATED WORK

Numerous researchers initiate their research contribution on detecting smart contract vulnerabilities. These state of art research may be classified into two distinct groups namely symbolic execution and classical static analysis to identify vulnerabilities. Next class of works, includes to explores deep learning models to deal with smart contracts. They brought up the idea of vulnerability candidate slicing, which has the potential to greatly enhance deep learning model performance [4]. [5] suggested an entirely novel technology named Peculiar that enhances detection through the use of data flow graphs.

In Paper [6] author developed CBGRU, a hybrid learning model which improves vulnerability detection accuracy through integrating the benefits of many algorithms. These techniques lack the ability to encode significant advice from experts. Since deep learning is a black-box, and the most of them are not very understandable.

In paper [7] author proposes Eth2Vec is a static analysis tool that uses machine learning to find vulnerabilities in smart contracts. Eth2Vec remains resilient to code rewrites, so it can identify flaws in rebuilt code as well. The attributes that analysts manually construct are needed as inputs for machine-learning-based static analysis tools. Eth2Vec, on the other hand, automatically learns the characteristics of contracts that are susceptible by using a neural network for language processing.

In paper [8]. a DBC-MulBiLSTM framework for smart contract vulnerability identification was presented. The system initially extracts contextual characteristics from smart contracts using the lightweight pre-trained model DistilBERT, and then it uses Convolutional Neural Networks (CNN) to find local features. To enhance the model's capacity to identify intricate vulnerability patterns, feature fusion creates a multi-dimensional feature representation. The MulBiLSTM training framework was established by integrating a multi-head self-attention mechanism into the BiLSTM architecture. This approach makes it possible to simultaneously capture long-range connections throughout the whole dataset, improving the model's capacity to accurately depict complex dependencies and contextual information.

An Attention-based Wide and Deep Neural Network (AWDNN) for Ethereum smart contract reentrancy vulnerability detection is presented in this research paper [9]. AWDNN improves its accuracy in spotting intricate vulnerability patterns by highlighting important smart contract characteristics. Code optimization, vectorization,

and vulnerability identification are the three stages of our methodology. We simplify smart contract programming by eliminating unnecessary elements and identifying important snippets. In order to identify vulnerabilities, these pieces are converted into vectors that encapsulate the semantic properties of the smart contract. They are then sent through a deep and extensive neural network. According to experimental data, our model outperforms current tools. In order to increase efficiency, future research will try to identify more vulnerabilities and apply sophisticated vectorization techniques.

[10]. paper suggests, the use of deep and wide neural networks to find vulnerabilities in smart contracts is the aim of this article. Author provide WIDENNET, a deep neural network-based technique for identifying smart contract vulnerabilities related to timestamp reliance and reentrancy. With this method, the contracts' bytecodes are extracted, converted to Operational Codes (OPCODES), and then converted into unique vector representations. In order to discover vulnerabilities, these vectors are then input into a neural network, which extracts both basic and complex patterns.

The paper [11, 12] The author suggested an novel based BiGAS-based smart contract reentrancy vulnerability detection approach. It detects reentrancy vulnerabilities in smart contracts with an accuracy and F1-score of more than 93%. Softmax has been switched out for the SVM classifier in the model to confirm that the selection of SVM is one of the factors contributing to our method's improved performance. With Softmax in lieu of the classifier, the model's accuracy was 89.78%, and its F1-score was 89.83%. The author also contrasted various deep learning-based vulnerability detection techniques with sophisticated automated audit systems. In contrast to the current state-of-the-art techniques, the precision and

III. METHODOLOGY AND IMPLEMENTATION

3.1 Model Design

In order to address the security issues, a novel approach has been introduced in smart contracts, wherein the source code encapsulating the data and control relationships between the executable scripts is transformed to create a contract graph. Nodes in the graph indicate critical function invocations or variables, and edges store the temporal execution traces of those nodes. Since most graphical neural networks have flat response propagation function. An elimination step has been designed to normalize the graph. The degree of freedom in Graphical Convolution networks has been enhanced in order to control normalized graphs. Furthermore, a unique temporal message propagation network (TMP) has been suggested after taking into account the diverse roles and temporal relationships of numerous program elements.

The tasks have been divided into subtasks while implementing the proposed model, which are listed below

A) Design Of Graph Neural Network

Graphical neural networks (GNNs) are deep learning algorithms focused on manipulating graphical topology. GNNs remain the most commonly used method for graph analysis because to their outstanding efficiency and effortless interpretability. Traditional CNNs and Recurrent Neural networks could be envisioned as being extended to graph-based data by GNNs. Applications such as object recognition, picture categorization, and image identification frequently employ CNNs. To enhance and improve the security of block chain-based systems, this suggested approach uses GNNs as an approach to determine smart contract vulnerabilities in block chain systems. A GNN would potentially be trained in this scenario to predict the occurrence of a particular kind of vulnerability, such as an endless loop, timestamp, or reentrancy issue. The graph representation of the susceptible smart contract that we wish to test would be the input to the GNN, and the output would be a binary prediction indicating whether or not the contract differs from a certain kind of vulnerability.

B) Graph Convolutional Network Design

In the spatial domain, node formations from their surrounding neighborhoods are combined in a process known as graphical convolution. Graphical convolutional networks have been established as a result of this kind of algorithm. This kind of neural network design uses the information about the nodes from an adjacent node in a

convolutional way using graph structure. The degree of freedom in Graphical Convolutional Networks can learn models of graphs and perform better in a range of tasks and applications because of their strong expressive capabilities. One of the most fundamental GNNs for implementing convolution operations into graph designs is degree free GCN. Graph convolution networks may be classified as either spatial or spectral domains based on feature extraction techniques. It results from processing the graph signal, and a filter is used to accomplish graph convolution. It is comparable to taking an input signal and filtering out the noise to get the classification result.

C) Multi-Method Privacy Framework

The extensive research provides an unprecedented privacy-preservation architecture which integrates innovative algorithms from deep learning with the decentralized capabilities of block-chain to enhance security and privacy. Fundamentally, the architecture avoids the dangers associated with centralized platforms by distributing data control to users via block-chain. To ensure transparency and compliance to privacy standards, it uses smart contracts to govern data access guidelines.

D) Intrusion Detection Systems

Through the monitoring and detection of actions that contradict established safety rules, intrusion detection systems perform a crucial role in protecting computer networks. Anomaly based, misuse-based, and hybrid approaches are the three classes into whose intrusion detection system technologies descent. Outliers are identified via statistical models, and anomaly-based intrusion detection systems are made for recognizing strange behaviors that deviate from standard practices [14]. By comparing observed actions to known attack patterns, misuse-based intrusion detection systems utilize signature-based techniques to identify known threats. Hybrid intrusion detection systems integrate both techniques, improving detection capabilities by utilizing each method's advantages to cover a wider range of threats.

E) Threat Model

The threat model focused point-to-point vulnerabilities and emphasizes on several attack routes for block-chain-based systems. The strategy, that addresses both internal and external threats, is based on the assumption that nodes within the network may not be entirely trustworthy. Although protecting user data within the network, the block-chain has restricts due to threats that attempt to establish false node identities in order to influence a segment of the network (Eclipse attacks) or the entire network (Sybil attacks). In the recommended research strategy, the block-chain network's node vulnerability has been taken into account.

F) Skip-Gram Model

The target word has been determined by the skip-gram model shown Fig. 2. By deriving a context word representation. The input has been organized into training terms and tokenized. The amount of weight that is liable for the output is the matrix of interest. By considering an assortment of training words into account, the model's main objective is to enhance the average logarithmic probability [17].

Let the sequence training words be $u_1,u_2,u_3....u_R$

The total number of training corpus T be

$$\frac{1}{T} = \sum_{t=1}^{T} \sum_{-k<j<k, j\neq 0} log(u_{t+1}|u_t) \quad \text{--(1)}$$

k is the size of the training context word w_t. The increased value of k is a consequence of the inclusion of a larger number of training instances, leading to enhanced accuracy throughout the training phase [17,18]. t denotes the index of the current target word in the training corpus, and j

is the relative position within the context window, specifying the offset from the current target word u_T.

fundamental equation for the definition of the skip-gram structure is the probability of observing $P(w_{t+j} + w_t)$ is given by softmax funtion. The Softmax function is a mathematical function often used in ML, particularly in the

context of neural networks and classification tasks. The Soft-max function takes a vector z and applies the following formula to each element z_i in Eq. 2

$$softmax(z_i) = \rho\left(\frac{z_i}{z_j}\right) = \frac{e^{z_i}}{\sum_j e^{z_j}} \quad \text{-- (2)}$$

It converts a vector of numbers into a probability distribution, where each number represents the likelihood of the corresponding class.

G) Privacy-Based Blockchain System

In order to improve security and privacy, the algorithm offers a structured method that methodically combines block-chain technology with a dual-layer privacy-preservation mechanism is shown in Fig 2. By using cutting-edge block-chain and machine learning technologies, this algorithm strengthens security by carefully processing and safeguarding user data via a sequence of steps from pre-processing to secure block-chain transactions, guaranteeing strong data integrity and privacy.

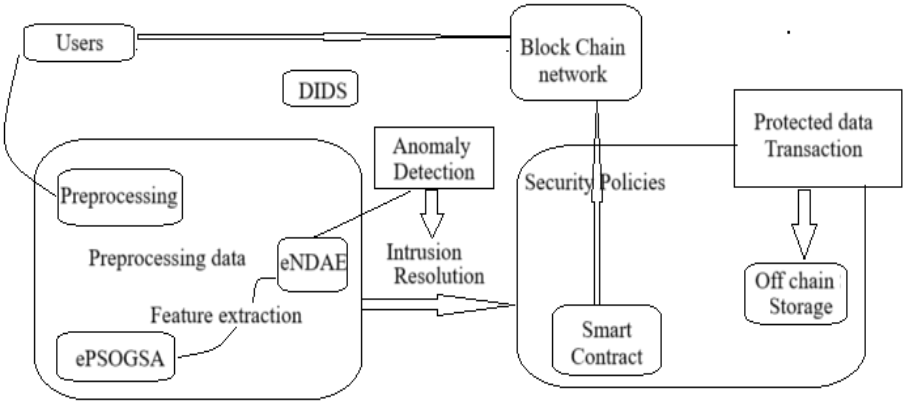


Fig. 3: privacy-preservation and block chain integration model

H) Dataset Architecture & Construction

There are three phases that make up our method's overall architecture as shown in Fig. 4.

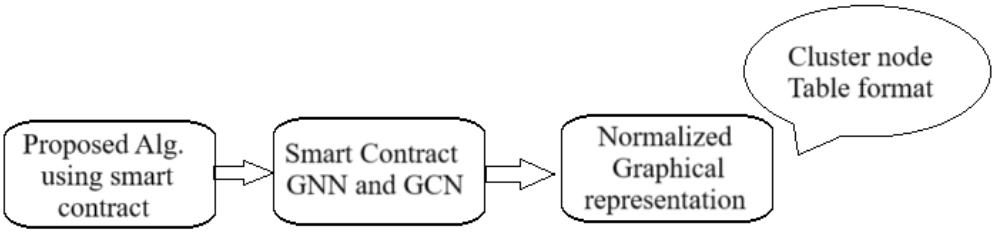
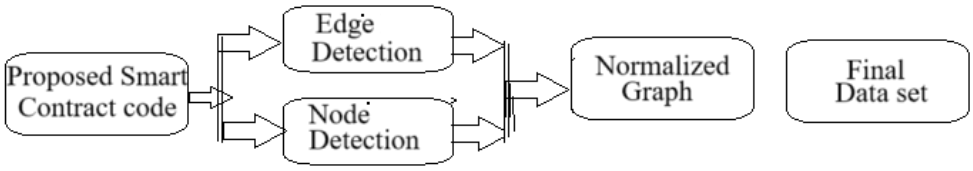


Fig 4: Proposed Dataset construction

The data and control flow semantics of the smart contract code are extracted during the first graphical generation phase. Proposed code program is converted into symbolic graph format that can retain the semantic connections between code elements. Furthermore, an explicit model of a fallback mechanism is included. In the second step, which includes the functions from the k-partite graph, the graphs are normalized. Figure 4 illustrates how the major nodes have been used to characterize each of the most significant functions, which can be expressed by M_1, M_2,M_3...,M_n. The secondary nodes are used to simulate the crucial variables, which are identified by S_1,S_2,S_3...,S_n. Throughout model execution, the secondary nodes were used to model important variables like user balance and bonus flag. The last stage, the third stage, involves building networks for message propagation with the aim to identify and simulate smart contract vulnerabilities. The temporal number of these edges indicates their order inside the contract function, and they clarify potential paths that the function may follow.

The proposed flow of the dataset construction is shown in Fig 5.



A tuple including the beginning and ending nodes *(V_s and V_e)*, their sequence in time *(t)*, and the edge type*(o)* determines each edge's a trait. Four distinct kinds of edges are created in order to establish the semantic link between the nodes: control flow, data, fallback, and forward edges.

IV. EVALUATION RESULTS

The detailed proposed parametric includes the characteristics performance of GNN and degree free GCN models are shown in Table 1.

Table 1: Parametric of the models

SL no.	Parameters	Proposed GNN model	Proposed GCN model
1	Learning Rate	0.019	0.001
2	Dropout values	0.18	0.1
3	Epochs	50	50
4	Size for each batch	32	32
5	Filter	64	64
5	Folds	5	5
6	Swapped node	yes	NA

The models are evaluated based on the metrics listed as

a) Precision

$$Precision = \frac{Total\ number\ of\ Positive\ instance}{Total\ number\ of\ positive\ instance + False\ Positive\ instance}$$

b) Recall

$$Recall = \frac{Total\ number\ of\ Positive\ instance}{Total\ number\ of\ positive\ instance + False\ negaticve\ instance}$$

c) F1 Score

$$F1\ score = \frac{2(precision\ X\ Recall)}{Precision + Recall}$$

d) Accuracy

$$Accuracy = \frac{Sum\ (Positive\ instance + Negative\ instance)}{Sum(Positive\ and\ Negative\ instance) + Sum\ (Faslse\ Positive\ nad\ Negative\ instance)}$$

The Evaluation metrics performance is shown in Table 2.

Sl no.	Evaluation Metrics	Ex. GNN	Ex GCN	Prop. GNN	Prop. GCN
1	Precision	0.8787	0.916	0.876	0.923
2	Recall	0.8161	0.867	0.82	0.904
3	F1 Score	0.8531	0.887	0.864	0.967
4	Accuracy	0.9114	0.9292	0.927	0.932

It can be observed from the proposed evaluation metrics that GNN has done far better than GCN to identify the vulnerabilities. The purpose for this is that even though graph neural networks (GNNs) are generalized variants of GCNs that can operate with unordered and variable node counts, GCNs are specifically designed to function with organized input.

CONCLUSION & FUTURE SCOPES

However, an abundance of incidents on smart contract vulnerabilities cause clients to lose considerable funds and raise issues about how trustworthy blockchain systems are. A completely automated vulnerability analysis of smart contracts has been proposed in the current study. In contrast to previous techniques, the fallback mechanism for smart contracts has been explained, intricate links among program components are considered, and the possibility of using cutting-edge graph neural networks for vulnerability discovery is proposed.The model accomplished the goal of finding multiple vulnerabilities in smart contracts. The outcomes show that the model has a 93% accuracy rate to recognize vulnerabilities. An improvised deep learning model and security feature extraction may be used to find new vulnerabilities in the smart contract. This will improve the model's accuracy and highlight the right issues.

REFERENCES

[1]. Gad, A. G., Mosa, D. T., Abualigah, L., & Abohany, A. A. (2022). Emerging trends in blockchain technology and applications: A review and outlook. Journal of King Saud University-Computer and Information Sciences, 34(9), 6719-6742.

https://doi.org/10.1016/j.jksuci.2022.03.007

[2]. Zhang, L., Wang, J., Wang, W., Jin, Z., Su, Y., & Chen, H. (2022). Smart contract vulnerability detection combined with multi-objective detection. Computer Networks, 217, 109289.

https://doi.org/10.1016/j.comnet.2022.109289

[3]. Yu, X., Zhao, H., Hou, B., Ying, Z., & Wu, B. (2021, July). Deescvhunter: A deep learning-based framework for smart contract vulnerability detection. In 2021 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

https://doi.org/10.1109/IJCNN52387.2021.9534324

[4]. X. Yu, H. Zhao, B. Hou, et al., Deescvhunter: a deep learning-based framework for smart contract vulnerability detection, in: Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–8, https://doi.org/10.1109/IJCNN52387.2021.9534324

[5]. L. Zhang, W. Chen, W. Wang, et al., Cbgru: a detection method of smart contract vulnerability based on a hybrid model, Sensors 22 (9) (2022) 3577, https://doi.org/10.3390/s22093577.

[6]. H. Wu, Z. Zhang, S. Wang, et al., Peculiar: smart contract vulnerability detection based on crucial data flow graph and pre-training techniques, in: Proceedings of the 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE), IEEE, 2021, pp.378–389,

https://doi.org/10.1109/ISSRE52982.2021.00047.

[7]. Ashizawa, N., Yanai, N., Cruz, J. P., & Okamura, S. (2021, May). Eth2vec: learning contract-wide code representations for vulnerability detection on ethereum smart contracts. In Proceedings of the 3rd ACM international symposium on blockchain and secure critical infrastructure (pp. 47-59). https://doi.org/10.1145/3457337.3457841

[8]. Xu, S., He, H., Mihaljević, M. J., Zhang, S., Shao, W., & Wang, Q. (2025). DBC-MulBiLSTM: A DistilBERT-CNN Feature Fusion Framework enhanced by multi-head self-attention and BiLSTM for smart contract vulnerability detection. Computers and Electrical Engineering, 123, 110096.

https://doi.org/10.1016/j.compeleceng.2025.110096

[9]. Osei, S. B., Huang, R., & Ma, Z. (2025). An Attention-based Wide and Deep Neural Network for Reentrancy Vulnerability Detection in smart contracts. Journal of Systems and Software, 112361.

https://doi.org/10.1016/j.scico.2024.103172

[10] Osei, S. B., Ma, Z., & Huang, R. (2024). Smart contract vulnerability detection using wide and deep neural network. Science of Computer Programming, 238, 103172. https://doi.org/10.1016/j.jss.2025.112361

[11]. Zhang, L., Li, Y., Guo, R., Wang, G., Qiu, J., Su, S., ... & Tian, Z. (2024). A novel smart contract reentrancy vulnerability detection model based on BiGAS. Journal of Signal Processing Systems, 96(3), 215-237. https://doi.org/10.1007/s11265-023-01859-7

[12]. Guo, R., Chen, W., Zhang, L., Wang, G., & Chen, H. (2022). Smart contract vulnerability detection model based on siamese network (SCVSN): a case study of reentrancy vulnerability. Energies, 15(24), 9642.

https://doi.org/10.3390/en15249642

[13]. Vidal, F. R., Ivaki, N., & Laranjeiro, N. (2024). Vulnerability detection techniques for smart contracts: A systematic literature review. Journal of Systems and Software, 112160. : https://doi.org/10.5281/zenodo.8109651

[14]. Frimpong, S. A., Han, M., Effah, E. K., Adjei, J. K., Hanson, I., & Brown, P. (2024). A deep decentralized privacy-preservation framework for online social networks. Blockchain: Research and Applications, 5(4), 100233.

https://doi.org/10.1016/j.bcra.2024.100233

[15]. Gao, T., & Li, F. (2022, May). Machine learning-based online social network privacy preservation. In Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security (pp. 467-478).

[16]. D. Singh, B. Singh, Feature wise normalization: an effective way of normalizing data, Pattern Recogn. 122 (2022)108307. https://doi.org/10.1016/j.patcog.2021.108307.

[17]. Ma, C., Wang, T., Zhang, L., Cao, Z., Huang, Y., & Ding, X. (2023). Distributed representation learning with skip-gram model for trained random forests. Neurocomputing, 551, 126434.

https://doi.org/10.1016/j.neucom.2023.126434

[18] Gupta, N. A., Bansal, M., Sharma, S., Mehrotra, D., & Kakkar, M. (2024). Detection of vulnerabilities in blockchain smart contracts using deep learning. Wireless Networks, 1-17. https://doi.org/10.1007/s11276-024-03755-9