Indonesian Journal of Electrical Engineering and Computer Science
Vol. 38, No. 3, June 2025, pp. 1793~1803
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v38.i3.pp1793-1803

Vulnerability detection in smart contact using chaos optimization-based DL model

Srinivas A. Vaddadi[1], Sanjaikanth E. Vadakkethil Somanathan Pillai[2], Rohith Vallabhaneni[1], Santosh Reddy Addula[1], Bhuvanesh Ananthan[3]
[1]Department of Information Technology, University of the Cumberlands, Williamsburg, USA
[2]School of Electrical Engineering and Computer Science, University of North Dakota, Grand Forks, USA
[3]Department of Electrical and Electronics Engineering, PSN College of Engineering and Technology, Tirunelveli, India

Article Info

Article history:

Received Mar 18, 2024
Revised Nov 19, 2024
Accepted Nov 30, 2024

Keywords:

Blockchain
Deep learning
Loss function optimization
Smart contract
Vulnerability detection

ABSTRACT

This research article introduces a deep learning (DL) for identifying vulnerabilities in the smart contracts, leveraging an optimized DL method. The proposed method, termed LogT BiLSTM, combines bidirectional long short-term memory (BiLSTM) with logistic chaos Tasmanian devil optimization (LogT) for enhancing detection of vulnerability. The evaluation of the suggested approach is conducted using publicly available datasets. Initially, preprocessing steps involve removing duplicate data and imputing missing data. Subsequently, the vulnerability detection process utilizes BiLSTM, with the optimization of the loss function achieved through LogT. Results indicate promising performance in identifying vulnerabilities in SC, highlighting the efficacy of the LogT_BiLSTM approach.

Corresponding Author:

Srinivas A. Vaddadi
Department of Information Technology, University of the Cumberlands
6178 College Station Drive, Williamsburg, KY 40769, USA
Email: Vsad93@gmail.com

1. INTRODUCTION

Due to the accelerated growth of cryptocurrencies and the gradual maturation of blockchain (BC) technology, which together provide decentralised operations, transparent transaction processes, and tamper-evident properties, the concept of BC and digital currency has drawn a lot of attention in recent years. Thanks to these new technologies, computer protocols designed to be distributed, validated, or carried out in an informative way have been developed into smart contracts [1]. These contracts may be used as the basis for a variety of services and applications. The fast rise of smart contracts is causing them to become more complicated, and this is creating more important security concerns owing to the frequent incidence of smart contract vulnerabilities [2], [3].

Smart contracts play a pivotal role in securing information within decentralized systems, particularly in BC technology. These self-executing contracts are encoded with predefined rules and conditions, facilitating automatic execution and enforcement without the need for intermediaries [4]. By leveraging cryptographic techniques, SC provides the integrity and immutability of data, making them opposed for tampering and unauthorized access. They establish trust among parties by providing transparent and auditable transactions, as every action is recorded on the BC [5]. Furthermore, smart contracts enable permissioned access to sensitive information, allowing only authorized parties to interact with the data based on predefined permissions. Through their automated and transparent nature, smart contracts enhance the security of

information by reducing the risks associated with human error, fraud, and third-party interference. Thus, they serve as a robust mechanism for securing data in decentralized ecosystems [6].

Smart contracts face several challenges in guarding against malicious activity, including vulnerabilities in the code, exploits, and attacks aimed at manipulating contract execution or stealing sensitive information. Traditional security measures may not always suffice to address these threats effectively. However, deep learning (DL)-based malicious detection mechanisms offer a promising solution by leveraging advanced algorithms to identify and mitigate risks [7]. These mechanisms analyze large datasets of smart contract code and transaction histories to detect patterns indicative of malicious behavior. By learning from past incidents and continuously updating their models, DL algorithms can adapt to evolving threats and identify previously unseen attack vectors [8]. Additionally, they can detect anomalies in real-time, enabling proactive responses to potential security breaches. Furthermore, DL-based detection mechanisms can complement traditional security measures by providing a more robust defense against sophisticated attacks [9]. Through their ability to analyze complex data structures and detect subtle patterns, these mechanisms enhance the resilience of smart contracts against malicious activity, thereby strengthening the overall security of decentralized systems [10]. Thus, an optimized DL mechanism is introduced in this research. The major contribution is:

- Design of LogT algorithm: the logistic tasmanian devil algorithm (LogT) is designed by integrating the logistic chaos with the conventional Tasmanian devil to enhance the performance in terms of convergence rate.
- Design of LogT_BiLSTM based vulnerability detection: the vulnerability in the smart contract is detected using the proposed LogT_BiLSTM, wherein the loss function optimization is devised using the LogT algorithm.

The organization of the research is: Section 2 details the related works with problem statement and section 3 details the proposed malicious detection mechanism. The experimental results are presented in section 4 and conclusion in section 5.

2. RELATED WORKS

The existing methods concerning the malicious node detection in the BC scenario is detailed in this section. Aiming to address the security issues [11] designed a novel approach that integrates both technologies for enhanced anomaly detection. The technique hinges on deep autoencoder neural networks (DANNs) embedded within smart contracts deployed on a BC. DANNs excel at identifying deviations from normal patterns, making them ideal for anomaly detection. By integrating them into smart contracts, the system leverages the BC ‘s inherent security and transparency, ensuring data immutability and verifiable execution. The introduced methodology involves training DANNs on historical sensor data to establish a baseline of normal system behavior. Real-time sensor data is then fed into the smart contracts, where the DANNs analyze for anomalies. If detected, pre-defined actions are automatically triggered, such as sending alerts or initiating remediation procedures. This distributed, autonomous approach eliminates the need for a central authority, enhancing security and resilience. The failure in optimizing the parameters of the architecture limits the accuracy of detection.

A DL technique adept at analyzing graph structures like smart contracts was designed by [12]. GNNs extract features from the contract’s code, capturing complex relationships and dependencies. Phase two incorporates expert rules, encoded as decision trees, to identify specific vulnerability patterns. This combined approach leverages the generalizability of DL and the precision of expert knowledge, aiming for better accuracy and wider vulnerability coverage. Here, the risk assessment feature streamlines mitigation efforts by focusing on critical vulnerabilities. Incorporating and updating expert rules necessitates continuous involvement from security experts, posing potential scalability concerns.

A novel approach to identify the threat was designed by [13] by synergizing the power of DL and multimodal fusion (MMDF). The core technique revolves around exploiting diverse informational facets of smart contracts. It extracts not only the source code itself, but also the compiled bytecode and control flow graph, representing each modality through distinct DL models. These models, tailored to each data type, capture comprehensive vulnerability signatures embedded within. The key innovation lies in the multimodal fusion module, which intelligently combines the outputs from these individual models, leveraging their complementary strengths to achieve a more robust and accurate vulnerability detection.

A DL-based approach for accurate vulnerability detection was designed by [14] concerning the Smart contract. The technique hinges on two key elements: the power of bidirectional long short-term memory (BiLSTM) networks and the metric learning prowess of Triplet Loss. BiLSTM networks excel at capturing temporal dependencies within code sequences, making them adept at understanding the complex logic of smart contracts. Triplet Loss, on the other hand, pushes similar contracts closer together in the

feature space while driving dissimilar ones further apart, enhancing the model's ability to distinguish vulnerable contracts from secure ones. The combination of BiLSTM and Triplet Loss surpasses traditional methods, achieving high accuracy in detecting various vulnerability types. The training process, especially with Triplet Loss, can be computationally demanding, requiring powerful computing resources.

A system combining machine learning and BC technology was designed by [15] using verifiable byzantine fault tolerance (VBFT) approach. The technique leverages two key components like Machine learning for decentralized anomaly detection and BC for secure and tamper-proof data storage. At each sensor node, a lightweight machine learning model, like an Isolation Forest, analyzes local sensor data and network interactions. This model identifies deviations from normal behavior, potentially indicating malicious activity. These anomaly reports are then cryptographically signed and broadcasted to the network. Distributed detection and tamper-proof data storage make the system more resilient to attacks and manipulation. Maintaining a scalable and efficient BC infrastructure for large-scale WSNs requires further research. Besides, balancing anomaly detection with preserving node privacy necessitates careful design of data sharing mechanisms.

Hackers have also become interested in smart contract security [16], as they have been using smart contract weaknesses to steal illicit assets. A re-entrant vulnerability in TheDAO's crowdfunding contract was exploited by attackers in 2016, causing losses of roughly USD 60 million [17]. A Parity wallet [18] contract vulnerability in 2017 caused user losses of USD 31 million. The market value of the BEC tokens produced by the United States Chain firm dropped to nearly nothing in 2018 as a result of hackers taking use of an integer overflow vulnerability in the Ethereum ERC-20 smart contract [19]. As of 2023, attacks against Ethereum smart contracts had resulted in cumulative damages above USD 3.1 billion, as per data from the SlowMist website [20]. The vulnerability issue of smart contracts has drawn the attention of more academics due to the frequency of BC security incidents and the growing amount of assets involved in smart contracts [21]. When security issues arise with smart contracts, it might be challenging to retrieve stolen assets because of its tamper-evident and automatically executing features. Therefore, in order to guarantee the security of smart contracts and the protection of each participant's asset interests, it is imperative to identify potential security flaws prior to deployment. The majority of current smart contract vulnerability detection techniques [22] are based on conventional program vulnerability detection techniques (such C++ and Java), which typically use static analysis [23] and dynamic execution techniques [24] to find issues. This approach includes flaws including limited automation, low efficiency, and inadequate flexibility and frequently depends on the experience of specialists. The application of DL technology for smart contract vulnerability detection has gained popularity in recent years due to its rapid development [25].

Problem statement: The current strategies for detecting vulnerabilities in SC are primarily based on the common methods used in conventional programmes like Java and C++. In addition to having issues with limited automation, low efficiency, and inadequate adaptability, this approach frequently depends on the knowledge of experts. The application of DL technology for smart contract vulnerability detection has been a research hotspot in recent years due to the technology's fast growth. The shortcomings of traditional detection approaches, such as their limited automation, low efficiency, and reliance on specialised expertise, can be compensated for by DL-based detection technologies. In this research, an optimised DL model is developed to address the problems.

3. PROPOSED VULNERABILITY DETECTION AND MITIGATION MECHANISM

The proposed vulnerability detection of the smart contract is designed based on optimized DL method. Initially, the pre-processing is devised based on removing the duplicate data and missing data imputation. Finally, the vulnerability detection is devised using the proposed logistic chaos Tasmanian devil optimization based bidirectional long short term memory (LogT_BiLSTM). In this, the vulnerability detection is devised using the BiLSTM, wherein the loss function optimization is devised using the LogT. The structure of proposed vulnerability detection and mitigation mechanism for smart contract is portrayed in Figure 1.

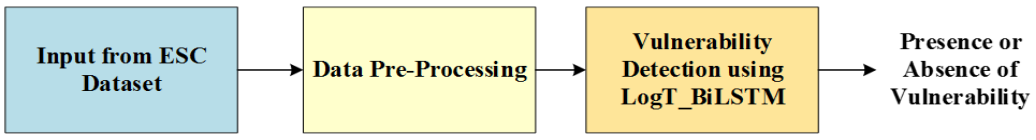


Figure 1. Proposed vulnerability detection and mitigation mechanism for smart contract

3.1. Data acquisition

The input data for processing the proposed vulnerability detection in the smart contact is acquired from ethernet smart contracts (ESC) dataset.

3.2. Data pre-processing

The duplication removal and missing value imputation are devised in the data pre-processing step for enhancing the performance. Here, duplicate data removal ensures that each data point in your dataset is unique, which can prevent biases and inaccuracies caused by redundant information.

3.3. Vulnerability detection using LogT-BiLSTM

The vulnerability detection is devised using the proposed LogT-BiLSTM, wherein the loss functions of the BiLSTM is optimized using the LogT algorithm. The structure of proposed LogT_BiLSTM method is portrayed in Figure 2.

Structure of BiLSTM: the BiLSTM is applicable for time-series data analysis, including vulnerability detection in smart contracts. Input layer: the input to the BiLSTM model consists of sequences of data points. In the context of vulnerability detection in smart contracts, these data points could represent various features extracted from the smart contract code. BiLSTM layer: the essential structure of the BiLSTM model comprises two LSTM layers: one processes the input sequence in the forward direction, while the other processes it in the backward direction. This bidirectional approach enables the model to grasp dependencies from both the past and future, enhancing its ability to understand contextual information within the data. The equations governing the operations within an LSTM cell are as follows:

- Forget gate: $k_l = \beta(A_k \cdot [g_{l-1}, m_l] + d_k)$ (1)
- Input gate: $p_l = \beta(A_p \cdot [g_{l-1}, m_l] + d_p)$ (2)
- Candidate gate: $\tilde{B}_l = tanh(A_B \cdot [g_{l-1}, m_l] + d_B)$ (3)
- Cell state update: $B_l = k_l \cdot B_{l-1} + p_l \cdot \tilde{B}_l$ (4)
- Output gate: $q_l = \beta(A_q \cdot [g_{l-1}, m_l] + d_q)$ (5)
- Hidden state update: $g_l = q_l \cdot tanh(B_l)$ (6)

where, the input at time step l is denoted as m_l, the hidden state of the LSTM cell from the previous time step is signified as g_{l-1}, and the forget, input, candidate, and output gate is denoted as k_l, p_l, $\tilde{B}_l$, and q_l respectively. The cell state at time step$l$is defined as B_l, and sigmoid activation function is signified as β. Hyperbolic tangent activation function is denoted as $tanh$, and weight matrices for the forget gate, input, candidate, and output is denoted as d_k, d_p, d_B, and d_q respectively. Besides, the bias concerning the layers is signified as A_k, A_p, A_B, and A_q.

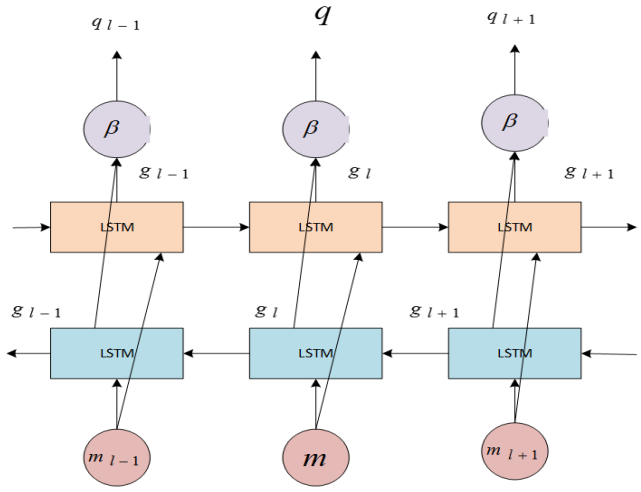


Figure 2. Structure of BiLSTM

Output layer: by leveraging the bidirectional nature of the BiLSTM model, it can effectively obtain complex patterns and dependencies in the input sequences, making it suitable for vulnerability detection in smart contracts where understanding both past and future code context is crucial.

Loss function optimization using LogT algorithm: the loss function optimization is devised using the LogT algorithm, wherein the logistic chaos is integrated in the Tasmanian devil algorithm for obtaining the global best solution. Here, the Logistic chaotic mapping introduces randomness into the search process, helping the algorithm escape local optima and explore a wider range of solutions. This can be particularly useful for complex optimization problems with many local minima. The initialization of the algorithm is represented as (7).

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_r \\ \vdots \\ F_P \end{bmatrix}_{P\times G} = \begin{bmatrix} f_{1,1} & \cdots & f_{1,v} & \cdots & f_{1,G} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ f_{r,1} & \cdots & f_{r,v} & \cdots & f_{r,G} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ f_{P,1} & \cdots & f_{P,v} & \cdots & f_{P,G} \end{bmatrix}_{P\times G} \tag{7}$$

Fitness estimation: the fitness of a solution candidate is typically inversely proportional to its MSE value. In other words, lower MSE values indicate better performance, so solutions with lower MSE values are considered to have higher fitness. The optimization algorithm aims to minimize this MSE value by iteratively adjusting the parameters of the model or the solution candidates until a satisfactory solution is found. This process involves generating new candidate solutions, evaluating their fitness using the MSE, and updating the parameters based on the fitness scores. It is expressed as:

$$Fit = \frac{1}{d}\sum_{x=1}^{d}(act_x - pre_x)^2 \tag{8}$$

where, the number of data points in the dataset is signified as d, the actual target value for the x^{th} data point is defined as act_x, and the predicted value for the x^{th} data point is referred as pre_x. The fitness function is defined as Fit.

Exploration phase: the exploration phase mimics the hunting behavior of Tasmanian devils. Besides, the exploration phase plays a crucial role in diversifying the population and helping the algorithm avoid getting stuck in local optima. The detection of prey is evaluated initially and is formulated as:

$$A_r = F_m, r = 1,2,\ldots.P, m \in \{1,2,\ldots.P | m \neq r\} \tag{9}$$

where, the search agent r chooses the prey A_r. After identifying the prey, the individuals update their position and are defined as:

$$f_{r,v}^{n,D1} = \begin{cases} f_{r,v} + g \cdot \left(s_{v,w} - M \cdot f_{v,w}\right), Fit_{A_r} < Fit_r \\ f_{r,v} + g \cdot \left(f_{v,w} - s_{v,w}\right), Otherwise \end{cases} \tag{10}$$

$$F_r = \begin{cases} F_r^{n,D1}, Fit_r^{n,D1} < Fit_r \\ F_r, Otherwise \end{cases} \tag{11}$$

where, g indicates the variable with $[0,1]$, $E_r^{n,D1}$ is the new location that the search agent has assessed, and the definitions of the fitness function and carrion's fitness are $Fit_r^{n,D1}$ and Fit_{A_r} respectively. M indicates the arbitrary variable with the value, it is shown as that which uses the values 2 or 1. Here, if the selection of prey and movement mechanisms lack randomness, the algorithm might become repetitive and get stuck in the same patterns, which is solved by including logistic chaos mapping in the exploration phase. The logistic chaos is outlined as:

$$f_{r,v}^{n,D1} = H \cdot f_{r,v}^{D1}\left(1 - f_{r,v}^{D1}\right) \tag{12}$$

where, the factor utilized for obtaining the enhanced randomization phase is defined as H that falls within the range of $[0,1]$. The solution obtained after the inclusion of logistic chaos is defined as:

$$\left(f_{r,v}^{n,D1}\right)_{LogT} = 0.5\left(f_{r,v}^{n,D1}\right)_{Tasmanian \rightleftarrows devil} + 0.5\left(f_{r,v}^{n,D1}\right)_{logisticchaos} \tag{13}$$

$$\left(f_{r,v}^{n,D1}\right)_{LogT} = 0.5\left(\begin{cases} f_{r,v} + g \cdot \left(s_{v,w} - M \cdot f_{v,w}\right), Fit_{A_r} < Fit_r \\ f_{r,v} + g \cdot \left(f_{v,w} - s_{v,w}\right), Otherwise \end{cases}\right) + 0.5\left(H \cdot f_{r,v}^{D1}\left(1 - f_{r,v}^{D1}\right)\right) \quad (14)$$

$$\left(f_{r,v}^{n,D1}\right)_{LogT} = \begin{cases} 0.5\left(f_{r,v} + g \cdot \left(s_{v,w} - M \cdot f_{r,v}\right)\right) + 0.5\left(H \cdot f_{r,v}^{D1}\left(1 - f_{r,v}^{D1}\right)\right), Fit_{A_r} < Fit_r \\ 0.5\left(f_{r,v} + g \cdot \left(f_{r,v} - s_{v,w}\right)\right) + 0.5\left(H \cdot f_{r,v}^{D1}\left(1 - f_{r,v}^{D1}\right)\right), Otherwise \end{cases} \quad (15)$$

using the proposed solution updation (15), the global best solution is obtained.

Local search: the algorithm prioritizes exploration in regions of the solution space that have shown potential for good solutions based on the information gathered in the exploration phase. In this phase, the search agent chases it by updating its position based on the chosen prey's location. This update involves a combination of movement towards the prey and random exploration within the local neighborhood. The prey is chosen through:

$$Q_r = F_m, r = 1,2,\ldots.P, m \in \{1,2,\ldots.P | m \neq r\} \quad (16)$$

where, Q_r signifies the prey chosen by r^{th} search agent. Then, the solution updated by the search agent is represented as.

$$\left(f_{r,v}^{n,D1}\right) = \begin{cases} f_{r,v} + g \cdot \left(d_{v,w} - M \cdot f_{r,v}\right), Fit_{Q_r} < Fit_r \\ f_{r,v} + g \cdot \left(f_{r,v} - q_{r,v}\right), Otherwise \end{cases} \quad (17)$$

$$F_r = \begin{cases} F_r^{n,D2}, Fit_r^{n,D2} < Fit_r \\ F_r, Otherwise \end{cases} \quad (18)$$

Where, the fitness of the prey is defined as Fit_{Q_r}, the fitness of solution is defined as $Fit_r^{n,D2}$, v^{th} variable's solution is defined as $f_{r,v}^{n,D1}$, and the solution estimated by the search agent $r$is defined as $F_r^{n,D2}$. The exploitation beyond the search area is limited by adjusting the radius and is formulated as (19).

$$K = 0.01\left(1 - \frac{l}{L}\right) \quad (19)$$

After adjusting the search radius, the position updation of the search agent is defined as (20).

$$f_{r,v}^{n} = f_{r,v} + (2K - 1) \cdot Q \cdot f_{r,v} \quad (20)$$

The updated solution is evaluated for its feasibility and is represented as (21).

$$F_r = \begin{cases} F_r^{n}, Fit_r^{n} < Fit_r \\ F_r, Otherwise \end{cases} \quad (21)$$

The upper bound of algorithm is defined as L, but the current iteration is signified as l.

Stoppage: the algorithm terminates after a specified number of iterations have been completed. This criterion ensures that the algorithm doesn't run for an excessively long time. Else, LogT may stop when the improvement in the best solution found falls below a certain threshold over a predefined number of iterations. This indicates that the algorithm has converged to a satisfactory solution. Thus, using the solution obtained through the LogT, the loss function optimization is devised for identifying the vulnerability in smart contract.

4. RESULT AND DISCUSSION

Besides, the existing smart contract based malicious detection methods were compared with suggested approach to illustrate the superiority of the proposed method. The conventional methods like BiLSTM [14], Deep AutoEncoder [11], MMDF [13], and VBFT [15] are utilized for comparison.

4.1. Comparative analysis

Figure 3 presents a comprehensive comparative analysis of the proposed model's performance against other methods, including Deep AutoEncoder, MMDT, BLSTM, and VBFT, across four key

evaluation metrics: accuracy, precision, recall, and F-score, for varying K-fold values. In terms of accuracy Figure 3(a), the proposed model consistently outperforms the competing techniques, demonstrating superior predictive accuracy across all K-fold values. Precision analysis (Figure 3(b)) further highlights the proposed method's ability to minimize false positives effectively, as it consistently achieves the highest precision scores. Similarly, in recall Figure 3(c), the proposed approach outperforms its counterparts by identifying relevant instances with minimal false negatives, indicating its reliability in recall-sensitive scenarios. Finally, the F-score analysis Figure 3(d), which balances precision and recall, shows that the proposed model achieves the highest scores across all K-fold values, underscoring its robust and balanced performance. Overall, these results confirm the superior efficacy of the proposed model compared to the baseline methods in all evaluated metrics.

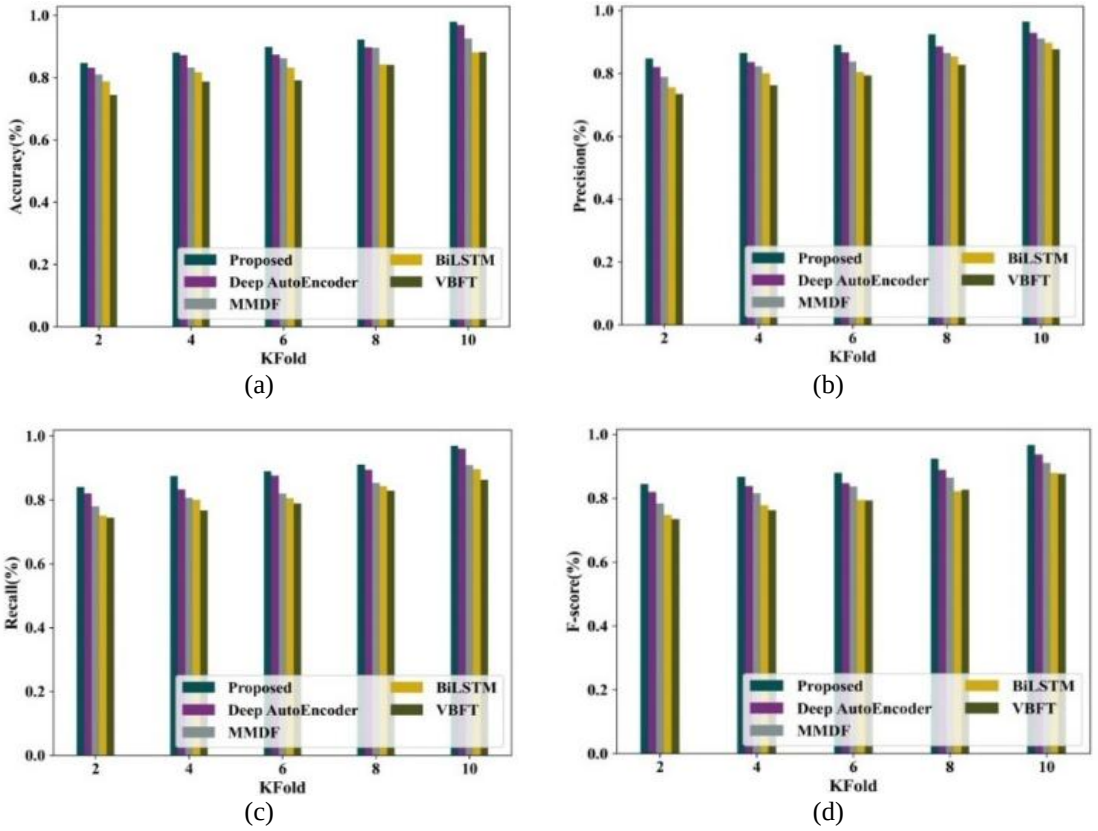


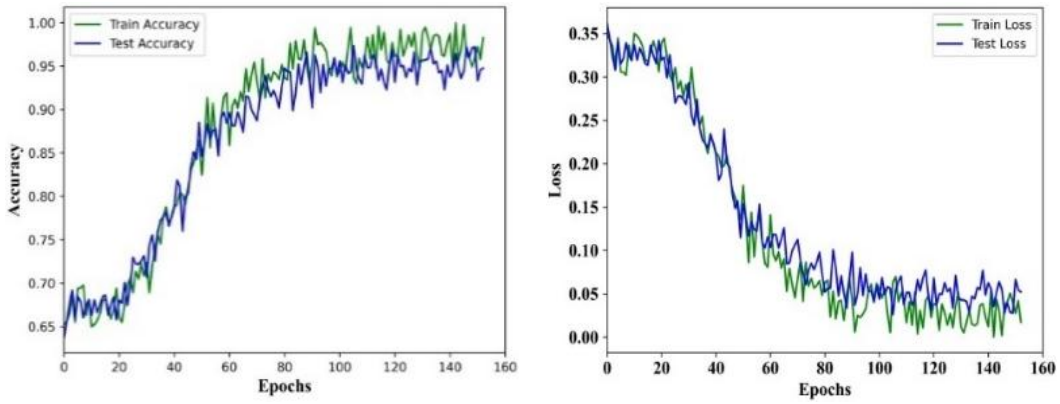
Figure 3. Comparative analysis: (a) accuracy, (b) precision, (c) recall, and (d) F-score

The analysis is devised by varying the K-fold value of the methods and the outcome is presented. Here, the better outcome is evaluated with higher value of K-fold and the proposed LogT_BiLSTM. Here, with a higher K-fold value, the dataset is divided into more subsets, allowing for a more comprehensive evaluation of the model's performance across different data samples. This reduces the variance in performance estimation, providing a stable and reliable assessment of the model's effectiveness. Besides, by training and testing the model on multiple different subsets of the data, increasing the K-fold value enables the model to generalize better to unseen data.

4.2. Accuracy-loss

Accuracy refers to the proportion of correctly classified instances (both intrusions and normal activities) by the model among all instances. Loss, on the other hand, represents the error or discrepancy between the predicted and actual values. Here, the closeness of testing and training accuracy and loss depicts the enhanced generalization capability of the model, which provides better outcome while evaluating the unknown data in Figure 4. Figure 4 illustrates the accuracy-loss analysis for both training and testing phases over multiple epochs, focusing on (a) accuracy and (b) loss. In Figure 4(a), the training and testing accuracy show a consistent upward trend as the epochs progress, with the model achieving high accuracy values near convergence. This indicates effective learning and generalization, as the test accuracy closely aligns with the training accuracy without significant overfitting. In Figure 4(b), both the training and testing loss exhibit a

steady decline with increasing epochs, demonstrating that the model effectively minimizes error during the learning process. The near-convergence of training and testing loss values reflects the model's robust performance and ability to generalize well to unseen data. These observations collectively validate the model's capability to learn efficiently and perform reliably on both training and testing datasets.



(a) (b)

Figure 4. Accuracy-loss analysis (a) accuracy and (b) loss

4.3. AUC analysis

The AUC analysis is portrayed in Figure 5. Here, the analysis based on various measures illustrates the suggested approach's superiority. The reason behind the superiority is the weight optimization of the BiLSTM with the novel LogT algorithm. The inclusion of logistic mapping based chaotic mapping within the Tasmanian Devil algorithm assist to enhance the detection accuracy.

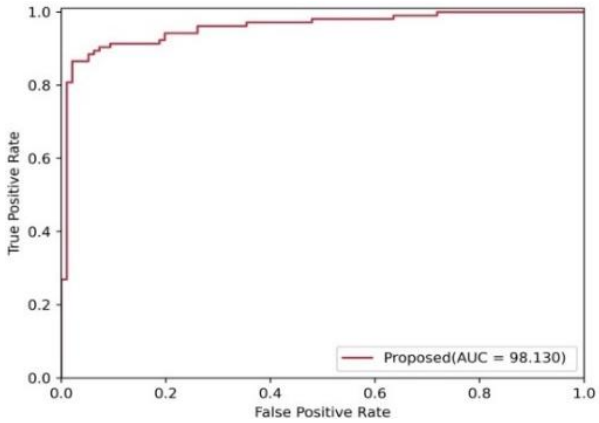


Figure 5. AUC analysis

5. CONCLUSION

In this study, DL model is introduced for detecting vulnerabilities in SC is which is crucial for ensuring their security and reliability in BC applications. By combining BiLSTM with logistic chaos Tasmanian devil optimization (LogT), we have developed an effective method, termed LogT_BiLSTM, for enhancing vulnerability detection. Through evaluation on publicly available datasets, it is demonstrated the efficacy of the proposed approach in accurately identifying vulnerabilities. The preprocessing steps of removing duplicate data and imputing missing data contribute to the robustness of our method. Overall, our findings suggest that LogT_BiLSTM holds promise for improving the security of smart contracts in BC ecosystems. Future research directions may involve further optimization techniques and real-world deployment of the proposed approach. Future research on vulnerability detection in smart contracts using a

chaos optimization-based DL model can focus on enhancing feature engineering to better capture vulnerabilities and developing hybrid models that combine chaos optimization with other techniques for improved detection accuracy. There is potential to create scalable, real-time detection systems and explore transfer learning for cross-platform vulnerability detection. Additional areas include incorporating explainable AI to make the detection process more transparent, integrating dynamic and static analysis, and establishing benchmarks for standardized evaluation. Further research could also focus on automating the patching of vulnerabilities, reducing manual intervention in smart contract security.

ACKNOWLEDGEMENTS

The Author with a deep sense of gratitude would thank the supervisor for his guidance and constant support rendered during this research.

FUNDING INFORMATION

No funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Srinivas A. Vaddadi	✓		✓		✓	✓		✓		✓			✓	
Sanjaikanth E. Vadakkethil Somanathan Pillai		✓	✓	✓	✓	✓		✓	✓			✓	✓	
Rohith Vallabhaneni	✓		✓	✓	✓	✓	✓			✓	✓		✓	
Santosh Reddy Addula	✓	✓	✓	✓			✓	✓	✓	✓				
Bhuvanesh Ananthan			✓	✓		✓			✓		✓	✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, [S.A.V], upon reasonable request.

REFERENCES

[1] I. A. A. El-Moghith and S. M. Darwish, "Towards designing a trusted routing scheme in wireless sensor networks: a new deep blockchain approach," *IEEE Access*, vol. 9, pp. 103822-103834, 2021, doi: 10.1109/ACCESS.2021.3098933.
[2] L. K. Ramasamy, F. K. KP, A. L. Imoize, J. O. Ogbebor, S. Kadry, and S. Rho, "Blockchain-based wireless sensor networks for malicious node detection: A survey," *IEEE Access*, vol. 9, pp. 128765-128785, 2021, doi: 10.1109/ACCESS.2021.3111923.
[3] R. K. Sharma and R. S. Pippal, "Malicious attack and intrusion prevention in IoT network using blockchain based security analysis," *In 2020 12th International Conference on Computational Intelligence and Communication Networks (CICN), IEEE*, pp. 380-385, 2020, doi: 10.1109/CICN49253.2020.9242610.
[4] A. Diro, N. Chilamkurti, V. D. Nguyen, and W. Heyne, "A comprehensive study of anomaly detection schemes in IoT networks using machine learning algorithms," *Sensors*, vol. 21, no. 24, pp. 8320, 2021, doi: 10.3390/s21248320.
[5] R. Kumar, P. Kumar, R. Tripathi, G. P. S. Garg, and M. M. Hassan, "A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network," *Journal of Parallel and Distributed Computing*, vol. 164, pp. 55-68, 2022, doi: 10.1016/j.jpdc.2022.01.030.
[6] P. Jisna, T. Jarin, and P. N. Praveen, "Advanced intrusion detection using deep learning-LSTM network on cloud environment," *In 2021 Fourth International Conference on Microelectronics, Signals and Systems (ICMSS), IEEE*, pp. 1-6, 2021, doi: 10.1109/ICMSS53060.2021.9673607.

[7] B. Jiang, Y. Liu, and W. K. Chan, "Contractfuzzer: fuzzing smart contracts for vulnerability detection," *In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pp. 259-269, 2018, doi: 10.1145/3238147.3238177.
[8] P. Qian, Z. Liu, Q. He, B. Huang, D. Tian, and X. Wang, "Smart contract vulnerability detection technique: a survey," *arXiv preprint arXiv:2209.05872*, 2022, doi: 10.48550/arXiv.2209.05872.
[9] M. Yu, J. Zhuge, M. Cao, Z. Shi, and L. Jiang, "A survey of security vulnerability analysis, discovery, detection, and mitigation on IoT devices," *Future Internet*, vol. 12, no. 2, pp. 27, 2020, doi: 10.3390/fi12020027.
[10] R. Agarwal, T. Thapliyal, and S. K. Shukla, "Vulnerability and transaction behavior based detection of malicious smart contracts," *In Cyberspace Safety and Security: 13th International Symposium, CSS 2021, Virtual Event, November 9–11, 2021, Proceedings 13, Springer International Publishing*, pp. 79-96, 2022, doi: 10.1007/978-3-030-94029-4_6.
[11] K. Demertzis, L. Iliadis, N. Tziritas, and P. Kikiras, "Anomaly detection via block chained deep learning smart contracts in industry 4.0," *Neural Computing and Applications*, vol. 32, pp. 17361-17378, 2020, doi: 10.1007/s00521-020-05189-8.
[12] Z. Liu, M. Jiang, S. Zhang, J. Zhang, and Y. Liu, "A smart contract vulnerability detection mechanism based on deep learning and expert rules," *IEEE Access*, 2023, doi: 10.1109/ACCESS.2023.3298048.
[13] W. Deng, H. Wei, T. Huang, C. Cao, Y. Peng, and X. Hu, "Smart contract vulnerability detection based on deep learning and multimodal decision fusion," *Sensors*, vol. 23, no. 16, pp. 7246, 2023, doi: 10.3390/s23167246.
[14] M. Wang, Z. Xie, X. Wen, J. Li, and K. Zhou, "Ethereum smart contract vulnerability detection model based on triplet loss and BiLSTM," *Electronics*, vol. 12, no. 10, pp. 2327, 2023, doi: 10.3390/electronics12102327.
[15] M. Nouman, U. Qasim, H. Nasir, A. Almasoud, M. Imran, and N. Javaid, "Malicious node detection using machine learning and distributed data storage using blockchain in WSNs," *IEEE Access*, vol. 11, pp. 6106-6121, 2023, doi: 10.1109/ACCESS.2023.3236983.
[16] S. Lin, L. Zhang, J. Li, L. Ji, and Y. Sun, "A survey of application research based on blockchain smart contract," *Wireless Networks*, vol. 28, pp. 635–690, 2022, doi: 10.1007/s11276-021-02874-x.
[17] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on ethereum smart contracts," *Cryptology ePrint Archive*, Paper 2016/1007, 2016. [Online]. Available: https://eprint.iacr.org/2016/1007
[18] *The Parity Wallet Hack Explained*, 2023. [Online]. Available: https://blog.openzeppelin.com/on-the-parity-wallet-multisig-hack-405a8c12e8f7
[19] K. Peng, M. Li, H. Huang, C. Wang, S. Wan, and K. K. R. Choo, "Security challenges and opportunities for smart contracts in internet of things: a survey," *IEEE Internet of Things Journal*, vol. 8, pp. 12004–12020, 2021, doi: 10.1109/JIOT.2021.3074544.
[20] *Slowmist*, 2023. [Online]. Available: https://hacked.slowmist.io/
[21] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Buenzli, and M. Vechev, "Securify: Practical security analysis of smart contracts," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, Toronto, ON, Canada, pp. 67–82, Oct. 2018, doi: 10.1145/3243734.3243780.
[22] S. S. Kushwaha, S. Joshi, D. Singh, M. Kaur, and H. N. Lee, "Systematic review of security vulnerabilities in ethereum blockchain smart contract," *IEEE Access*, vol. 10, pp. 6605–6621, 2022, doi: 10.1109/ACCESS.2021.3140091.
[23] A. Ghaleb and K. Pattabiraman, "How effective are smart contract analysis tools? Evaluating smart contract static analysis tools using bug injection," in *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, Los Angeles, CA, USA, Jul. 2020, pp. 415–427, doi: 10.1145/3395363.3397385.
[24] H. Wang, Y. Liu, Y. Li, S. W. Lin, C. Artho, L. Ma, and Y. Liu, "Oracle-supported dynamic exploit generation for smart contracts," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, pp. 1795–1809, 2020, doi: 10.1109/TDSC.2020.3037332.
[25] P. Qian, Z. Liu, Q. He, R. Zimmermann, and X. Wang, "Towards automated reentrancy detection for smart contracts based on sequential models," *IEEE Access*, vol. 8, pp. 19685–19695, 2020, doi: 10.1109/ACCESS.2020.2969429.

BIOGRAPHIES OF AUTHORS

Srinivas A. Vaddadi is a dynamic and forward-thinking professional in the field of Cloud and DevSecOps. With a solid educational foundation in computer science, Srinivas embarked on a journey of continuous learning and professional growth. Their relentless pursuit of knowledge and commitment to staying at the forefront of industry advancements has earned them recognition as a thought leader in the Cloud and DevSecOps space. He can be contacted at email: Vsad93@gmail.com.

Sanjaikanth E. Vadakkethil Somanathan Pillai (Senior Member, IEEE) holds an MS in Software Engineering from The University of Texas at Austin, Texas, USA, and a BE from the University of Calicut, Kerala, India. Currently pursuing a PhD in Computer Science at the University of North Dakota, Grand Forks, North Dakota, USA, his research spans diverse areas such as mobile networks, network security, privacy, location-based services, and misinformation detection. He is a proud member of Sigma Xi, The Scientific Research Honor Society, underlining his commitment to advancing scientific knowledge and research excellence. He can be contacted at email: s.evadakkethil@und.edu.

Rohith Vallabhaneni is a dedicated worker with a strong work ethic in leading teams to solve organizational issues. He is capable of learning all aspects of information within a company and using the technical knowledge and business background to effectively analyze security measures to determine their effectiveness in order to strengthen the overall security posture. He has great work ethic and outstanding team leadership skills and seek to accomplish organizational goals, while growing in knowledge and experience. He can be contacted at email: rohit.vallabhaneni.2222@gmail.com.

Santosh Reddy Addula a Senior Member of IEEE, is a research scholar at the University of the Cumberlands. His educational qualifications include a Ph.D. and a Master of Science in Information Technology. With extensive experience in the IT industry, he has demonstrated expertise across multiple domains. Santosh is an innovator who has made significant contributions to academic research through his articles as an author and co-author. Additionally, he serves as a reviewer for esteemed journals, demonstrating his commitment to advancing knowledge and upholding high standards in scholarly publications within his field. He can be contacted at email: santoshaddulait@gmail.com.

Bhuvanesh Ananthan received the B.E. degree in Electrical and Electronics Engineering from Anna University in 2012, M.Tech. in Power System Engineering from Kalasalingam University in 2014 and Ph.D. degree from Faculty of Electrical Engineering of Anna University in 2019. He has published more than papers in reputed 100 international journals, 75 papers in international conferences and 20 books. He is a life time member of International Society for Research and Development, International Association of Engineers. He can be contacted at email: bhuvanesh.ananthan@gmail.com.