Semi-Supervised Learning and Blockchain Integration for Transparent Smart Contract Security

Muhammad Sannan Khaliq [1], Love Allen Chijioke Ahakonye [2], Jae Min Lee [1], Dong-Seong Kim [1] *
[1] IT-Convergence Engineering, *Kumoh National Institute of Technology*, Gumi, South Korea
* NSLab Co. Ltd., Gumi, South Korea, *Kumoh National Institute of Technology*, Gumi, South Korea
[2] ICT Convergence Research Center, *Kumoh National of Technology*, Gumi, South Korea
(Sannan, loveahakonye, ljmpaul, dskim)@kumoh.ac.kr

Abstract—**With the rapid proliferation of blockchain technology and smart contracts in consumer IoT systems, ensuring digital trust and security remains a persistent challenge due to vulnerability diversity, data scarcity, and limited on-chain auditability. To address these issues, this paper presents PureChain, a scalable trust framework that integrates semi-supervised deep learning for smart contract vulnerability detection with blockchain-based audit logging for transparent security assurance. The framework employs a lightweight Conv1D neural architecture trained via iterative pseudo-labeling and a Mean Teacher strategy, effectively leveraging tens of thousands of unlabeled contracts to minimize annotation requirements. Experimental evaluation shows that PureChain achieves macro F1-scores exceeding 99% and significant improvements in recall for rare vulnerabilities, outperforming both supervised and state-of-the-art baselines. The integration of blockchain ensures all detection events are immutably recorded, enabling accountable and verifiable device operation. These results demonstrate that adaptive semi-supervised learning, combined with on-chain transparency, provides a robust and efficient foundation for secure smart contract monitoring across IoT and edge environments, with future work targeting more adaptive pseudo-labeling, semi supervised learning (SSL) techniques and cross-chain generalization.**

Index Terms—**CNN, Deep Learning, Iterative Pseudo Labeling, Mean Teacher, PureChain**

I. Introduction

The increasing integration of blockchain technology and smart contracts into consumer electronics and edge computing devices is revolutionizing digital trust, automation, and asset management [1]–[3]. As billions of IoT devices connect to decentralized networks, embedded smart contracts drive critical consumer services and automated device interactions [3]. However, code vulnerabilities such as reentrancy, delegatecall exploits, integer overflow/underflow, and timestamp dependency remain persistent threats [4], [5]. These commonly exploited vulnerabilities compromise device reliability, financial security, and user privacy [6], [7].

Traditional smart contract auditing relies heavily on static analysis tools or manual code review, which are limited by the need for large labeled datasets and substantial expert labor constraints that do not scale for constantly evolving consumer environments [8]–[11]. This challenge is further compounded as device manufacturers and operators seek to deploy blockchain-based solutions at scale, often without the resources for exhaustive annotation and rapid model adaptation [12].

Recent advances in deep learning have enabled multi-label vulnerability detection with higher accuracy and automation [10]. In particular, semi-supervised learning: leveraging both labeled and abundant unlabeled contract data, shows promise for bridging the annotation gap and improving detection in real-world deployments [13]. Yet, few frameworks effectively integrate these advances with auditable, tamper-resistant blockchain trust mechanisms, essential to consumer applications demanding transparency and accountability.

In this work, we propose a blockchain-enabled trust framework that integrates semi-supervised neural models with real-time smart contract auditing for consumer IoT and edge systems. The approach employs a lightweight multi-kernel Conv1D network trained with iterative pseudo-labeling and Mean Teacher learning to achieve scalable and label-efficient vulnerability detection and device authentication. By immutably anchoring model outputs, confidence scores, and system events on-chain, PureChain ensures transparent auditing, verifiable trust management, and reproducible benchmarking across heterogeneous environments.

- A multi-kernel Conv1D model trained via pseudo-labeling and Mean Teacher learning for multi-label smart contract vulnerability detection.
- The framework linking semi-supervised detection with blockchain-based audit trails for verifiable trust.
- A scalable pipeline improving recall on rare vulnerabilities while minimizing annotation cost.
- Demonstration of edge-ready integration for secure and transparent device authentication.

II. Background and Related Studies

A. Blockchain-Based Smart Contracts and Security in Consumer IoT

Blockchain enables decentralized trust and tamper-evident data exchange in consumer IoT ecosystems, supporting secure coordination among heterogeneous devices without centralized control [14], [15]. Smart contracts—self-executing programs on platforms like Ethereum—automate services, data monetization, and identity verification. Integrating blockchain into fog and edge layers further enhances scalability and latency in smart home and industrial IoT applications [14].

Yet, increasing reliance on on-chain logic exposes IoT systems to numerous vulnerabilities, including reentrancy, delegatecall misuse, integer overflow/underflow, and weak access control [10], [16]. Incidents such as The DAO (2016) and Parity Wallet (2017) highlight their severity. Over fifty vulnerability classes have been identified across DeFi, NFT, and IoT systems, driven by complex execution semantics and limited semantic analysis [10]. Off-chain oracles and multi-chain interactions further expand the attack surface, emphasizing the need for robust vulnerability detection in blockchain-enabled IoT.

B. Evolution from Static to Deep Learning–Based Smart Contract Security

Traditional auditing tools such as Mythril, Oyente, Slither, and SmartCheck rely on symbolic and static analysis to detect issues like reentrancy and transaction-order dependencies [10], [16]. However, rule-based detection yields high false positives, poor adaptability, and depends on scarce labeled datasets with verified ground truth [14], [16], limiting scalability in IoT and blockchain systems. Deep learning methods address these gaps through adaptive, data-driven modeling. Examples include Lightning Cat (CodeBERT+CNN–LSTM) [10], BiLSTM–XAI frameworks [17], and SCsVulLyzer's GA-optimized profiling [18], alongside CNN, GNN, LSTM, and attention-based variants [19]–[24].

Recent smart-contract detectors mainly use transformers and multimodal inputs to capture syntax, semantics, and execution behavior [25]. PureChain instead emphasizes label efficiency and auditability, proposing a semi-supervised pipeline (pseudo-labeling with Mean Teacher) for label-scarce, edge-constrained settings, alongside tamper-evident logging of model inferences for verifiable security monitoring.

C. Semi-Supervised Learning and Automated Security

The scarcity of labeled datasets for smart contract vulnerability detection has motivated the use of semi-supervised learning methods that leverage large unlabeled corpora [16], [26]. Approaches such as pseudo-labeling, Mean Teacher, and self-training expand training data by assigning high-confidence pseudo-labels to unlabeled samples. Fazliani et al. (2025) demonstrated a 3–5% recall improvement for rare vulnerabilities without compromising precision, validating pseudo-labeling's utility in blockchain anomaly detection [26].

Hybrid approaches integrating active and semi-supervised learning further improve label efficiency. Sun et al. (2023) proposed ASSBert, combining uncertainty-based querying with confidence-driven pseudo-labeling. Trained on 20,829 contracts spanning six vulnerability classes, ASSBert surpassed both fully supervised and conventional semi-supervised baselines in F1 and recall. These iterative query–pseudo-label cycles underscore the scalability and labeling efficiency of hybrid learning for smart contract vulnerability detection [13].

D. Blockchain-Integrated Trust and Auditing Solutions

Blockchain can serve as a trust anchor for transparent security auditing. Frameworks such as PureChain, Blockchain-Based IoT Access Control, and BCM Hierarchical Ledgers use on-chain provenance to verify software authenticity and enforce distributed access control [1], [14], [27]. In healthcare and consumer IoT, decentralized NGAC and ABAC frameworks leverage blockchain for secure identity management and traceable data exchange [15]. Yet, few systems integrate deep or semi-supervised detection with on-chain, tamper-evident audit trails. We bridge this gap by logging risk vectors, model/version hashes, and system events on the PureChain ledger, enabling verifiable transparency without sacrificing edge deployability.

This section outlines the shift from static analysis to deep and semi-supervised learning in smart contract security, underscoring gaps in labeled data and blockchain-auditable integration. Our approach combines Conv1D-based semi-supervised pseudo-labeling with PureChain trust anchoring for scalable, verifiable vulnerability detection.

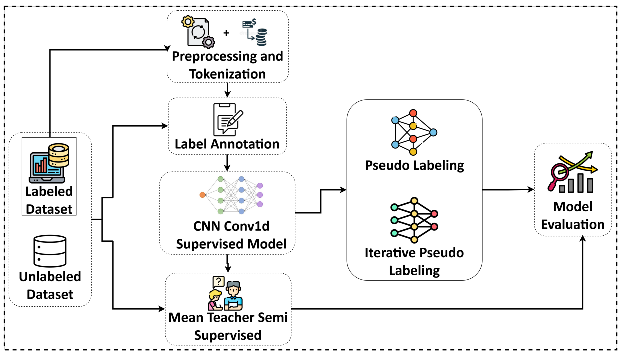
Preprocessing and Tokenization
Label Annotation
CNN Conv1d Supervised Model
Mean Teacher Semi Supervised
Labeled Dataset
Unlabeled Dataset
Pseudo Labeling
Iterative Pseudo Labeling
Model Evaluation

Fig. 1: Proposed System Model for Vulnerabilities Detection in Smart Contracts

III. System Methodology

The PureChain-enabled smart contract security pipeline is architected for scalable vulnerability detection, trust management, and device auditing in consumer IoT and edge environments. This section introduces the high-level system components, workflow, and integration between advanced deep learning, semi-supervised learning strategies, and the PureChain blockchain platform, shown in Figure 1.

A. Dataset Preparation and Labeling

Comprehensive sets of smart contracts are gathered from public blockchain sources and IoT deployment logs. The labeled dataset D_L contains contracts tagged for vulnerabilities, including reentrancy (RE), delegatecall (DE), integer overflow/underflow (IO/OF), and timestamp dependency (TP). Unlabeled contracts D_U are retained for semi-supervised experiments. To ensure balanced evaluation, stratified sampling splits labeled data into training, validation, and test subsets.

B. Preprocessing & Tokenization

Each contract undergoes code normalization: removal of comments, whitespace, and boilerplate, and is tokenized with a tailored vocabulary of 10,000 unique tokens. A trainable embedding layer maps token indices to $\mathbb{R}^{128}$ vectors. All sequences are padded or truncated to length T for GPU-efficient batch training.

C. Model Architecture

The proposed architecture aims to efficiently detect vulnerabilities within smart contract code using a multi-kernel convolutional neural network (CNN). This model is designed to process tokenized contract code and classify it for multiple vulnerabilities, including reentrancy, delegatecall, integer overflow/underflow, and timestamp dependency.

1) Embedding Layer: The input to the model is a token sequence S consisting of T tokens, each corresponding to a specific part of the contract code. These tokens are embedded into dense vectors of dimension 128, resulting in an embedding matrix $E \in \mathbb{R}^{T\times 128}$. This matrix represents the symbolic code as numerical vectors, enabling efficient processing by the subsequent convolutional layers.

2) Convolutional Feature Extraction: The feature extraction process utilizes three parallel 1D convolutional layers, each with 128 channels, to scan the embedded contract code at different granularities: Conv1 with a kernel size of 3, Conv2 with a kernel size of 4, and Conv3 with a kernel size of 5. These layers capture both fine-grained and broader patterns within the code. The outputs from all three layers are concatenated into a 384-dimensional feature vector that encapsulates the learned representations for each kernel size.

3) Dense Layers and Dropout Regularization: The concatenated feature vector is passed through two fully connected layers (also known as linear layers). The first layer projects the 384-dimensional vector into a 256-dimensional space, followed by a dropout layer with a probability $p = 0.5$ to prevent overfitting during training. The second fully connected layer reduces the dimensionality to 4, corresponding to the four vulnerability classes. Mathematically, the forward pass through these layers is represented using these Equations 1 2 and 3.

$$h = \text{Concat}(\text{Conv1}(E), \text{Conv2}(E), \text{Conv3}(E)), \tag{1}$$

$$z_1 = \text{Dropout}(\text{ReLU}(\text{fc}_1(h))), \tag{2}$$

$$\hat{y} = \sigma(\text{fc}_2(z_1)), \tag{3}$$

where h denotes the concatenated output of the convolutional layers, fc_1 and fc_2 are the fully connected layers, σ is the sigmoid activation function, and $\hat{y}$ represents the final predicted probability for each vulnerability class.

4) Activation Function: A sigmoid activation function is applied at the output layer, transforming each of the four class scores into a probability value between 0 and 1. These probabilities represent the model's confidence that a given contract contains each vulnerability.

A Conv1D-based architecture is selected for efficient deployment in resource-constrained IoT and edge environments. Unlike transformer and graph-based models with quadratic complexity $\mathcal{O}(T^2)$, Conv1D operates with linear complexity $\mathcal{O}(T\cdot k)$, enabling low-latency inference and reduced memory overhead. Convolutional filters effectively capture localized syntactic and semantic code patterns, particularly with multi-kernel designs. To offset limited expressiveness, semi-supervised regularization: iterative pseudo-labeling and Mean Teacher consistency—is employed, improving generalization without increasing architectural complexity. As demonstrated in Section IV, this approach achieves competitive detection performance while remaining suitable for scalable edge deployment.

D. Model Training

We evaluated three core methodologies for model training, such as supervised baseline, Pseudo-labeling, and mean teacher semi-supervised learning.

1) Supervised Baseline: For the supervised baseline, we employed a CNN Conv1d model trained exclusively on the labeled contracts. The model is optimized using binary cross-entropy loss, as defined in Equation 4.

$$L_{\text{sup}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\left[y_{ik}\log\hat{y}_{ik} + (1-y_{ik})\log(1-\hat{y}_{ik})\right], \tag{4}$$

where $\hat{y}_{ik} = \sigma(f_\theta(x_i))_k$ and σ is the sigmoid activation function. The Adam optimizer is used to minimize the loss during training.

2) Pseudo-Labeling (Single and Iterative): For Pseudo-Labeling, the model initially predicts pseudo-labels for the unlabeled contracts x_j. If the highest confidence prediction $\hat{y}_{jk}^U$ for class k exceeds a threshold $\tau = 0.9$, the contract is assigned a pseudo-label $\tilde{y}_j = 1$. These pseudo-labeled samples are then added to the labeled dataset D_L, and the model is retrained. This process is repeated iteratively, with pseudo-labeling and retraining cycles continuing until performance on the validation set plateaus, as Shown in Equation 5.

$$D_L \leftarrow D_L \cup \{(x_j, \tilde{y}_j) : \max_k \hat{y}_{jk}^U > \tau\}, \tag{5}$$

where D_L is the labeled dataset and $\tilde{y}_j$ is the pseudo-label assigned to the unlabeled sample.

3) Mean Teacher Semi-Supervised Learning: For Mean Teacher Semi-Supervised Learning, a student-teacher dual model is employed. The teacher model's weights are updated using an exponential moving average (EMA) of the student model's weights, as shown in Equation 6.

$$\theta' \leftarrow \alpha\theta' + (1-\alpha)\theta, \tag{6}$$

where θ and θ' are the weights of the student and teacher networks, respectively, and α is the EMA rate. A consistency loss is enforced between the student and teacher predictions on the unlabeled data $x \in D_U$ using Equation 7.

$$L_{\text{consist}} = \mathbb{E}_{x\in D_U}\left[\|\sigma(f_\theta(x)) - \sigma(f_{\theta'}(x))\|^2\right], \tag{7}$$

The total training objective combines the supervised loss and the consistency loss, as shown in Equation 8.

$$L_{\text{total}} = L_{\text{sup}} + \lambda L_{\text{consist}}, \tag{8}$$

where λ balances the supervised and unsupervised losses.

The evaluated supervised baseline, Pseudo-labeling, and mean teacher semi-supervised learning provide practical strategies for training a vulnerability detection model using both labeled and unlabeled data. The choice of method depends on the availability of labeled data and the desired trade-off between supervised and unsupervised learning.

E. PureChain System Integration

After convergence, the CNN is deployed within the PureChain framework for real-time device authentication and smart contract auditing. Detected vulnerabilities are encoded as risk vectors and immutably logged on-chain, enabling transparent and verifiable audits. Containerized edge and on-device deployments ensure scalability and reproducibility. Supervised, pseudo-labeling, and mean-teacher training regimes are evaluated using standard classification metrics, with ablation on labeled-data ratios and assessments of runtime and audit-logging overhead.

IV. Performance Evaluation

A. Dataset Description and Experimental Setup

Two Solidity corpora were used: SC_label dataset [1] (labeled) and smartbugs-wild dataset [2] (unlabeled). The labeled set includes 1,773 contracts, split into 1,773 training, 222 validation, and 222 testing samples across four vulnerability classes: Reentrancy (RE), Delegatecall (DE), Integer Overflow/Underflow (IO/OF), and Timestamp Dependency (TP). The unlabeled corpus contains 47,398 contracts. Each contract was tokenized into sequences of 512 tokens drawn from a 10000-word vocabulary, producing tensors of size (1773 × 512) for labeled and (47398 × 512) for unlabeled inputs.

Experiments were conducted using Python in Google Colab leveraging libraries such as PyTorch 2.1, data preprocessing and visualization utilities were implemented using Pandas v2.2.1, NumPy v1.26.4, Matplotlib v3.9.0 and scikit-learn, The Colab environment was configured with a Tesla T4 GPU, 12 GB RAM, and a virtual machine running Ubuntu 20.04 on a Windows 11 Pro system with an Intel i5-12400F CPU, 32GB RAM, and an NVIDIA GeForce RTX 3050 GPU. Training utilized the Adam optimizer (lr $= 1 \times 10^{-4}$, batch size = 32) with early stopping based on validation F_1. Three models were evaluated: a supervised Conv1D baseline; Iterative Pseudo-Labeling (IPL) with pseudo-labels added at confidence $\tau = 0.9$ until validation F_1 plateaued; and a Mean Teacher (SSL) model with exponential moving average ($\alpha = 0.99$) and consistency weight ($\lambda = 1.0$). Experiments were repeated three times, reporting mean ± standard deviation, with ASSBert [13] included for reference.

B. Evaluation Metrics and Overall Results

To assess the performance of the proposed Conv1D-based semi-supervised pseudo-labeling framework, four variants were evaluated: labeled-only, non-iterative pseudo-labeling, iterative pseudo-labeling, and mean teacher. The evaluation covered four primary smart contract vulnerability classes, namely, Reentrancy (RE), Dangerous Delegatecall (DE), Integer Overflow (OF), and Timestamp Dependency (TP), using precision, recall, F1-score, and overall accuracy standard classification metrics.

TABLE I: Per Class Precision.

Model	RE	DE	OF	TP
Labeled-only	0.9919	1.0000	0.9474	0.8235
Non-iterative Pseudo-labeling	0.9836	1.0000	0.9000	0.7879
Iterative Pseudo-labeling	0.9677	1.0000	0.7361	0.6800
Mean Teacher	1.0000	1.0000	0.9833	1.0000

Table I shows that the Mean Teacher model achieves perfect precision (1.0000) across all vulnerability classes, outperforming both labeled-only and pseudo-labeling baselines. The supervised model achieves high but inconsistent precision, whereas pseudo-labeling degrades due to noisy labels, leading to integer overflow/underflow (OF) and timestamp dependency (TP). Overall, Mean Teacher offers superior precision and robustness under label scarcity.

As shown in Table II, the Mean Teacher model attains the highest recall across most classes, achieving perfect recall for RE, DE, and OF, and near-perfect performance for TP (0.9677). In contrast, iterative pseudo-labeling exhibits a severe recall drop for DE (0.10), indicating sensitivity to class imbalance and noise. The labeled-only model performs well but underperforms on minority classes, underscoring the importance of Mean Teacher's semi-supervised regularization for robust detection of both common and rare vulnerabilities.

TABLE II: Per Class Recall.

Model	RE	DE	OF	TP
Labeled-only	1.0000	0.8000	0.9153	0.9032
Non-iterative Pseudo-labeling	0.9836	0.7000	0.9153	0.8387
Iterative Pseudo-labeling	0.9836	0.1000	0.8983	0.5484
Mean Teacher	1.0000	1.0000	1.0000	0.9677

Table III summarizes the aggregate precision–recall outcomes. The Mean Teacher achieves consistently superior, balanced class-wise F1-scores: 1.00 for RE, DE, and OF, and 0.94 for TP. Both supervised and non-iterative pseudo-labeling models yield strong F1 for RE and OF but exhibit reduced stability for DE and TP. The iterative pseudo-labeling variant, while computationally efficient, performs poorly on rare classes, underscoring the Mean Teacher's distinct advantage in comprehensive vulnerability coverage.

TABLE III: Per Class F1-Score.

Model	RE	DE	OF	TP
Labeled-only	0.9959	0.8889	0.9310	0.8615
Non-iterative Pseudo-labeling	0.9836	0.8235	0.9076	0.8125
Iterative Pseudo-labeling	0.9756	0.1818	0.8920	0.6071
Mean Teacher	1.0000	1.0000	0.9916	0.9368

A comparative evaluation was performed against the ASSBert framework, which integrates active learning and semi-supervised BERT for smart contract vulnerability detection [13] and with an Optimized DeBERTa [25]. Table IV

[1]https://www.kaggle.com/code/tranduongminhdai/smartcontract-vulnerablity-detection/input

[2]https://github.com/smartbugs/smartbugs-wild

shows that the proposed *Mean Teacher* model outperformed ASSBert across all shared vulnerability types, achieving F1-scores of 1.0000 (Reentrancy), 1.0000 (Dangerous Delegatecall), and 0.9916 (Integer Overflow), compared to ASSBert's range of 0.79–0.89. Overall, the proposed approach yielded an average performance gain of approximately 16–20%, while remaining fully automated and computationally efficient, requiring no manual labeling.

TABLE IV: Comparison of our best model with ASSBert [13] (20% labeling) and Optimized DeBERTa [25].

Model	Metric	Reentrancy	Integer Overflow	Timestamp Dep.
Ours	Prec	1.0000	0.9833	1.0000
	Rec	1.0000	1.0000	0.9677
	F1	1.0000	0.9916	0.9368
	Macro-Avg.	0.9944	0.9892	0.9761
[13]	Acc	0.7900	0.8570	0.7860
	Prec	0.5170	0.6170	0.5740
	Rec	0.7300	0.6580	0.7300
	Macro-Avg.	0.8110	0.5693	0.7060
[25]	Acc	0.95	0.95	0.92
	Prec	0.95	0.1	0.1
	Rec	0.98	0.1	0.77
	F1	0.97	0.1	0.87

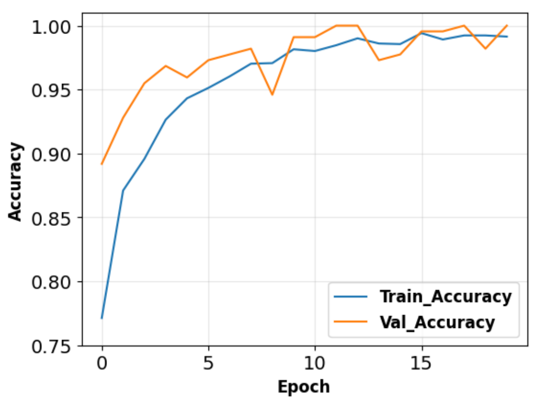

Fig. 2: Accuracy Learning Curve of Mean Teacher SSL

The accuracy and loss curves for the Mean Teacher model clearly demonstrate its effectiveness and stability. In Figure 2, accuracy rises rapidly and plateaus within a few epochs, indicating fast convergence and robust representation learning with minimal overfitting. The higher, more stable plateau compared to supervised and pseudo-labeling baselines highlights the advantage of semi-supervised consistency regularization.

Figure 3 shows a smoothly decreasing loss with minor oscillations, confirming stable optimization. The concurrent high and steady validation accuracy suggests effective collaboration between student and teacher networks, reducing noisy-label propagation and improving generalization to minority-vulnerable classes. Overall, these learning curves visually corroborate the numerical improvements in Table I, Table II, and Table III, confirming that semi-supervised training enhances not only peak accuracy and F1-scores but also model reliability and generalization across diverse smart contract vulnerabilities. The results collectively demonstrate that the proposed semi-supervised Conv1D framework attains

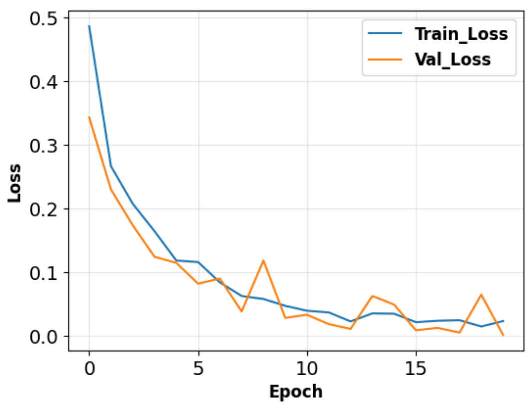

Fig. 3: Loss Learning curve of Mean Teacher SSL

transformer-level detection accuracy with far lower computational and labeling costs. The Mean Teacher model strikes a robust balance between precision and recall, confirming the effectiveness of pseudo-labeling for scalable, efficient smart contract vulnerability detection.

C. Ablation Studies

Ablation experiments highlight the influence of key hyperparameters on model performance. With labeled data ($\approx$ 1773 contracts) and all 47,398 unlabeled samples, the Mean Teacher model maintained $> 91\%$ macro F_1, confirming strong label efficiency. Performance peaked at a pseudo-label threshold $\tau = 0.9$, while lower τ introduced noise, reducing precision by $\approx 1.8\%$. Optimal stability was achieved for $\alpha \in [0.99, 0.999]$ and $\lambda \approx 1.0$; larger λ values over-regularized training. Class-wise analysis showed the greatest recall gains for minority classes: *Reentrancy* (+4.3 pp) and *Delegatecall* (+3.9 pp), demonstrating the SSL model's effectiveness in recovering underrepresented vulnerabilities.

D. Runtime and Audit-Logging Overhead

On an edge-class GPU, inference latency averages $\approx$ 3.5 ms per contract ($\approx$ 280 contracts/s). PureChain's on-chain audit step adds $\approx$ 0.7 s per transaction, including gas commitment, to immutably record the risk vector, model hash, and timestamp. Each audit entry occupies $\approx$ 1.2 KB on the ledger; weekly aggregation minimizes blockchain storage growth while preserving auditability.

E. Discussion

Semi-supervised learning improves recall and macro F_1 with reduced labeling effort. Iterative pseudo-labeling matches mean teacher performance at lower cost, while the mean teacher offers smoother convergence and slightly higher accuracy. *PureChain* enables transparent, reproducible security analysis and supports efficient edge deployment, achieving $\sim 25\,\text{ms}$ contract latency on lightweight nodes.

V. Conclusions

This paper presents PureChain, a blockchain-enabled framework that integrates lightweight Conv1D models with semi-supervised learning for secure, auditable smart-contract analysis in IoT-edge environments. By combining iterative pseudo-labeling and Mean Teacher training, PureChain achieved macro-F1 above 99% with minimal labeled data, while blockchain-based audit logging ensured transparency and integrity of security events.

A. Limitations and Future Work

The proposed framework relies on a lightweight Conv1D model that efficiently captures token-level patterns but does not explicitly encode structural semantics, such as control flow or interprocedural dependencies, which may limit the detection of complex execution-driven vulnerabilities. In addition, semi-supervised pseudo-labeling remains sensitive to noisy or adversarial unlabeled data, and blockchain-based audit logging introduces latency and scalability constraints tied to consensus assumptions. Future work will address these limitations by integrating a structure-aware hybrid architecture trained with semi-supervised learning, alongside adaptive confidence thresholding. As well as robustness analysis under poisoned unlabeled data and extended evaluation, including detailed blockchain performance.

Acknowledgment

This work was partly supported by Innovative Human Resource Development for Local Intellectualization program through the IITP grant funded by the Korea government(MSIT) (IITP-2026-RS-2020-II201612, 33%) and by Priority Research Centers Program through the NRF funded by the MEST(2018R1A6A1A03024003, 33%) and by the MSIT, Korea, under the ITRC support program(IITP-2026-RS-2024-00438430, 34%).

References

[1] S. Sun, R. Du, S. Chen, and W. Li, "Blockchain-Based IoT Access Control System: Towards Security, Lightweight, and Cross-Domain," *Ieee Access*, vol. 9, pp. 36 868–36 878, 2021.

[2] V. Gugueoth, S. Safavat, S. Shetty, and D. Rawat, "A Review of IoT Security and Privacy Using Decentralized Blockchain Techniques," *Computer Science Review*, vol. 50, p. 100585, 2023.

[3] L. A. C. Ahakonye, C. I. Nwakanma, and D.-S. Kim, "Tides of Blockchain in IoT Cybersecurity," *Sensors*, vol. 24, no. 10, p. 3111, 2024.

[4] D. Liu, J. Zhang, Y. Wang, H. Shen, Z. Zhang, and T. Ye, "Blockchain Smart Contract Security: Threats and Mitigation Strategies in a Lifecycle Perspective," *ACM Computing Surveys*, 2025.

[5] D. Ressi, A. Spanò, L. Benetollo, M. Bugliesi, C. Piazza, and S. Rossi, "Vulnerability Detection in Solidity Smart Contracts via Machine Learning: A Qualitative Analysis," *Blockchain: Research and Applications*, p. 100390, 2025.

[6] L. A. C. Ahakonye, C. I. Nwakanma, J.-M. Lee, and D.-S. Kim, "Agnostic CH-DT Technique for SCADA Network High-Dimensional Data-Aware Intrusion Detection System," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10 344–10 356, 2023.

[7] H. Ding, Y. Liu, X. Piao, H. Song, and Z. Ji, "SmartGuard: An LLM-enhanced framework for smart contract vulnerability detection," *Expert Systems with Applications*, vol. 269, p. 126479, 2025.

[8] Z. Guo, "Blockchain-Enhanced Smart Contracts for Formal Verification of IoT Access Control Mechanisms," *Alexandria Engineering Journal*, vol. 118, pp. 315–324, 2025.

[9] J. Crisostomo, F. Bacao, and V. Lobo, "Machine Learning Methods for Detecting Smart Contracts Vulnerabilities within Ethereum Blockchain- A Review," *Expert Systems with Applications*, p. 126353, 2025.

[10] X. Tang, Y. Du, A. Lai, Z. Zhang, and L. Shi, "Deep learning-Based Solution for Smart Contract Vulnerabilities etection," *Scientific Reports*, vol. 13, no. 1, p. 20106, 2023.

[11] 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, p. 7246, 2023.

[12] N. Li, Y. Liu, L. Li, and Y. Wang, "Smart Contract Vulnerability Detection Based on Deep and Cross Network," in *2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA)*. IEEE, 2022, pp. 533–536.

[13] X. Sun, L. Tu, J. Zhang, J. Cai, B. Li, and Y. Wang, "ASSBert: Active and Semi-Supervised BERT for Smart Contract Vulnerability Detection," *Journal of Information Security and Applications*, vol. 73, p. 103423, 2023.

[14] S. Algarni, F. Eassa, K. Almarhabi, A. Almalaise, E. Albassam, K. Alsubhi, and M. Yamin, "Blockchain-Based Secured Access Control in an IoT System," *Applied Sciences*, vol. 11, no. 4, p. 1772, 2021.

[15] S. Salonikias, M. Khair, T. Mastoras, and I. Mavridis, "Blockchain-Based Access Control in a Globalized Healthcare Provisioning Ecosystem," *Electronics*, vol. 11, no. 17, p. 2652, 2022.

[16] H. Rameder, M. Di Angelo, and G. Salzer, "Review of Automated Vulnerability Analysis of Smart Contracts on Ethereum," *Frontiers in Blockchain*, vol. 5, p. 814977, 2022.

[17] M. H. Maturi, E. De La Cruz, S. R. Addula, A. R. Yadulla, R. K. Ravindran, G. S. Nadella, H. Gonaygunta, and K. Meduri, "Enhancing Smart Contract Security with Explainable AI: A Framework for Reentrancy Vulnerability Detection and Explanation," in *2025 Systems and Information Engineering Design Symposium (SIEDS)*. IEEE, 2025, pp. 386–391.

[18] S. HajiHosseinKhani, A. H. Lashkari, and A. M. Oskui, "Unveiling Vulnerable Smart Contracts: Toward Profiling Vulnerable Smart Contracts Using Genetic Algorithm and Generating Benchmark Dataset," *Blockchain: Research and Applications*, vol. 5, no. 1, p. 100171, 2024.

[19] M. Ozdag, "Ai-Driven Vulnerability Analysis in Smart Contracts: Trends, Challenges and Future Directions," *arXiv preprint arXiv:2506.06735*, 2025.

[20] R. Kiani and V. S. Sheng, "Ethereum Smart Contract Vulnerability Detection and Machine Learning-Driven Solutions: A Systematic Literature Review," *Electronics*, vol. 13, no. 12, p. 2295, 2024.

[21] D. Han, Q. Li, L. Zhang, and T. Xu, "A Smart Contract Vulnerability Detection Model Based on Graph Neural Networks," in *2022 4th International Conference on Frontiers Technology of Information and Computer (ICFTIC)*, 2022, pp. 834–837.

[22] X. Feng and W. Chen, "SC-CCA: A Deep Learning Framework for Smart Contract Vulnerability Detection Based on CNN, BiLSTM, and Self-Attention Mechanism," in *2024 6th International Conference on Frontier Technologies of Information and Computer (ICFTIC)*, 2024, pp. 96–100.

[23] Y. Yang, H. Xu, and S. Yao, "Enhancing Smart Contract Vulnerability Detection with Hybrid Pre-Trained Models on Source Code and Byte Code," in *2024 6th International Conference on Frontier Technologies of Information and Computer (ICFTIC)*, 2024, pp. 122–126.

[24] M. S. Khaliq, L. A. C. Ahakonye, J. M. Lee, and D.-S. Kim, "Hybrid DeBERTa-BiLSTM-CNN for Enhanced Smart Contract Vulnerability Detection," in *Proceedings of Symposium of The Korean Institute of Communications and Information Sciences Summer Conference (KICS SUMMER 2025)*, 06 2025.

[25] M. S. Khaliq, S. Bin Noor, S. K. Ghosh, L. Allen Chijioke Ahakonye, J. M. Lee, and D.-S. Kim, "Optimized deberta for efficient smart contract vulnerability detection," in *2025 30th Asia-Pacific Conference on Communications (APCC)*, 2025, pp. 1–5.

[26] S. Fazliani, M. M. Sorond, and A. Masoudifard, "Leveraging Ensemble-Based Semi-Supervised Learning for Illicit Account Detection in Ethereum DeFi Transactions," *arXiv preprint arXiv:2412.02408*, 2024.

[27] D.-S. Kim, I. S. Igboanusi, L. A. Chijioke Ahakonye, and G. O. Anyanwu, "Proof-of-Authority-and-Association Consensus Algorithm for IoT Blockchain Networks," in *2025 IEEE International Conference on Consumer Electronics (ICCE)*, 2025, pp. 1–6.