Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.041152

ARTICLE

PoIR: A Node Selection Mechanism in Reputation-Based Blockchain Consensus Using Bidirectional LSTM Regression Model

Jauzak Hussaini Windiatmaja, Delphi Hanggoro, Muhammad Salman and Riri Fitri Sari*

Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Depok, 16424, Indonesia

*Corresponding Author: Riri Fitri Sari. Email: riri@ui.ac.id

Received: 12 April 2023 Accepted: 28 June 2023 Published: 29 November 2023

ABSTRACT

This research presents a reputation-based blockchain consensus mechanism called Proof of Intelligent Reputation (PoIR) as an alternative to traditional Proof of Work (PoW). PoIR addresses the limitations of existing reputation-based consensus mechanisms by proposing a more decentralized and fair node selection process. The proposed PoIR consensus combines Bidirectional Long Short-Term Memory (BiLSTM) with the Network Entity Reputation Database (NERD) to generate reputation scores for network entities and select authoritative nodes. NERD records network entity profiles based on various sources, i.e., Warden, Blacklists, DShield, AlienVault Open Threat Exchange (OTX), and MISP (Malware Information Sharing Platform). It summarizes these profile records into a reputation score value. The PoIR consensus mechanism utilizes these reputation scores to select authoritative nodes. The evaluation demonstrates that PoIR exhibits higher centralization resistance than PoS and PoW. Authoritative nodes were selected fairly during the 1000-block proposal round, ensuring a more decentralized blockchain ecosystem. In contrast, malicious nodes successfully monopolized 58% and 32% of transaction processes in PoS and PoW, respectively, but failed to do so in PoIR. The findings also indicate that PoIR offers efficient transaction times of 12 s, outperforms reputation-based consensus such as PoW, and is comparable to reputation-based consensus such as PoS. Furthermore, the model evaluation shows that BiLSTM outperforms other Recurrent Neural Network models, i.e., BiGRU (Bidirectional Gated Recurrent Unit), UniLSTM (Unidirectional Long Short-Term Memory), and UniGRU (Unidirectional Gated Recurrent Unit) with 0.022 Root Mean Squared Error (RMSE). This study concludes that the PoIR consensus mechanism is more resistant to centralization than PoS and PoW. Integrating BiLSTM and NERD enhances the fairness and efficiency of blockchain applications.

KEYWORDS

Blockchain; blockchain consensus; node selection; BiLSTM; RNN; regression

1 Introduction

Blockchain is a data structure in the form of interconnected data blocks that record encrypted blocks as an immutable distributed ledger [1]. Blockchain technology comprises two key elements: a peer-to-peer network of participating nodes and a distributed data structure. The primary function of the blockchain is as a database that stores assets distributed across various institutions, geographies, or sites, where all nodes in the network must own the same ledger. The concept of a distributed database

This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

applies to peer-to-peer networks on the blockchain. The significant difference between blockchain and distributed database systems is that there is no need for a third party to ensure network security and to trust as a reference [2]. Trusted third parties have the potential for a single point of failure problems. If a third party has a security hole, the network members will suffer losses or can trigger a conflict of interest between members who want to invoke a transaction. Blockchain maintains the security and reliability of data through a cooperative network, schemes, and cryptography that connect all blocks, making the data cannot be tampered with or changed.

All nodes in the blockchain network can propose transactions by sending broadcast messages [3]. Leader nodes, or miner nodes, will be responsible for validating, selecting, and ordering a series of transactions into a single block [4]. After successfully building blocks, miners will distribute them to all nodes in the network. The network will reach a consensus and execute transactions once all the nodes have added a new block to their copy of the blockchain. The network forms rules to reach a consensus and determine nodes eligible to validate and propagate blocks. With these rules, blockchain ensures information consistency within its network. By fulfilling all the rules in the consensus protocol, blockchain can guarantee the immutability, integrity, and consistency of data in the network.

In reliable distributed computing, reaching a consensus is essential. All participants must follow specific rules to coordinate and ensure system status consistently, even if there is a system conflict. Among the various blockchain consensus protocols, mainstream blockchain initiatives like Bitcoin and Ethereum are adopting Proof-of-Work (PoW) [5]. In PoW, mining nodes are selected based on their computational capabilities. PoW used nodes' computational capabilities to solve cryptographic puzzles with different levels of complexity according to the network processing capacity and desired block time. However, a PoW-based blockchain network requires enormous electricity consumption. According to data from the Cambridge Bitcoin Electricity Consumption Index, annual electricity consumption for Bitcoin mining reached 107.65 TWh in 2022, most of which is used to solve cryptographic puzzles. Besides providing trustless consensus, some studies argue that PoW serves no other useful purpose [6].

Researchers have conducted recent works to reduce the computational demand of PoW, i.e., Proof of Stake (PoS) [7] and Delegated Proof of Stake (DpoS) [8] as a public blockchain, Proof of Authority (PoA) [9], Proof of Space (PoSpace) [10], Proof of Elapsed Time (PoET) [11], Proof of Burn [12], and Proof of Weight [13] as private blockchain. These consensus protocols remove the mining process, which attracts massive computational demand. The node selection only involves calculations that do not consume massive computational resources. These consensuses use the nodes' reputation value to select the eligible node instead, i.e., nodes' stake, authority, or disk space.

However, these reputation-based consensuses tend to be less decentralized [14]. The condition might happen because selecting eligible nodes can be monopolized by accumulating resources considered when selecting miners. For instance, PoS attempts to identify authoritative nodes through the value of the original digital currency, which can partially address the deficiencies of PoW. However, the nodes with a significant stake system can monopolize the system easily. PoS nodes can also manipulate the network by increasing their stake in the blockchain network to win the mining process continuously. Therefore, PoS is usually denoted as nothing more than proof of oligarchy, as it establishes a validation system through which a small group of people with the most money has control over the network.

This paper proposes a new node selection mechanism in a reputation-based blockchain consensus using Bidirectional Long Short-Term Memory (BiLSTM) called Proof of Intelligent Reputation (PoIR) to deal with these issues. The nodes with a high reputation in PoIR will be responsible for mining and validating the blocks. PoIR is adopting the PoS consensus protocols, which lets the number of its stake select a leader. However, unlike PoS, PoIR considers multi-variable calculation. PoIR

uses Network Entity Reputation Database (NERD) [15] to find authoritative nodes. NERD records network entity profiles for each IP address and summarizes them into a reputation score. This record includes metadata about the reports and other relevant details, i.e., hostname, Autonomous System Number (ASN), and country.

Moreover, this paper implements the BiLSTM algorithm to estimate the probability of the reputation score of blockchain nodes. The score expresses each node's expected behavior and shows an accumulation of information about each node, enabling comparisons between nodes. Therefore, the proposed mechanism will be purely decentralized while having better transaction time than PoW.

The motivation for this study stems from the need to address the limitations of existing reputation-based consensus mechanisms in blockchain networks. While reputation-based approaches offer potential benefits such as improved scalability and reduced energy consumption compared to traditional PoW mechanisms, they often face challenges related to centralization, lack of fairness, and vulnerability to manipulation. Therefore, there is a need to develop an innovative reputation-based consensus mechanism that overcomes these limitations and fosters decentralization and fairness within blockchain networks.

This paper makes the following key contributions:

1. The paper proposes a novel node selection mechanism in a reputation-based blockchain consensus protocol, the Proof of Intelligent Reputation (PoIR), which integrates Bidirectional Long Short-Term Memory (BiLSTM) and Network Entity Reputation Database (NERD) to estimate nodes' reputation score. Integrating deep learning techniques with consensus protocols provides a dynamic and adaptive framework. The NERD system profiles each node's reputation score based on various metrics, allowing for a more nuanced and fair selection of authoritative nodes.
2. The proposed PoIR consensus balances decentralization and computational efficiency. It promises to improve performance-based consensus transaction time while preserving the decentralization principles of blockchain technology.
3. The paper addresses the pitfalls of current performance-based consensus mechanisms like PoW and reputation-based consensus mechanisms like PoS, providing a solution that reduces potential centralization and monopoly scenarios, making it a fairer option for all participants regardless of their stake or tenure in the network. This research will thus be instrumental in enhancing the robustness and fairness of blockchain applications.

The rest of this paper is organized as follows. Section 2 reviews related works on the blockchain consensus and the network entity reputation database. Section 3 describes the proposed PoIR mechanism. Section 4 shows the implementation results and analysis of the evaluation. Section 5 concludes the findings.

2 Literature Review

This section describes the related work, i.e., blockchain fundamentals, performance-based blockchain consensus, reputation-based blockchain, and Network Entity Reputation Database.

2.1 Blockchain

Haber et al. [16] discovered blockchain technology in 1991. They implement a method to secure digital documents. Their solution is based on time stamps and linking document hashes to create an incorruptible system and tamper-resistant. Blockchain became popular with its first application in

Bitcoin, which Satoshi Nakamoto made. Blockchain then defines it as a technology that provides data storage and transmission. A peer-to-peer architecture arranges blockchain technology, enabling nodes to communicate securely and transparently without relying on a single centralized authority.

Blockchain is a distributed and decentralized database, as seen in Fig. 1a. The participating nodes have the exact database copy. While being decentralized on the blockchain allows the system not to have a central authority and governing entity, each node has an equal role. In addition, how each block is stored also provides more security, as seen in Fig. 1b. Each block will be wrapped by an SHA-256 hash function and generate a block header. Each block header will be connected to the next block. With this process, if there is a block that a particular party changes, it will produce an invalid hash for the next block, then the blockchain will be invalid. Combining these two characteristics and processes makes the blockchain secure, transparent, and distributed.

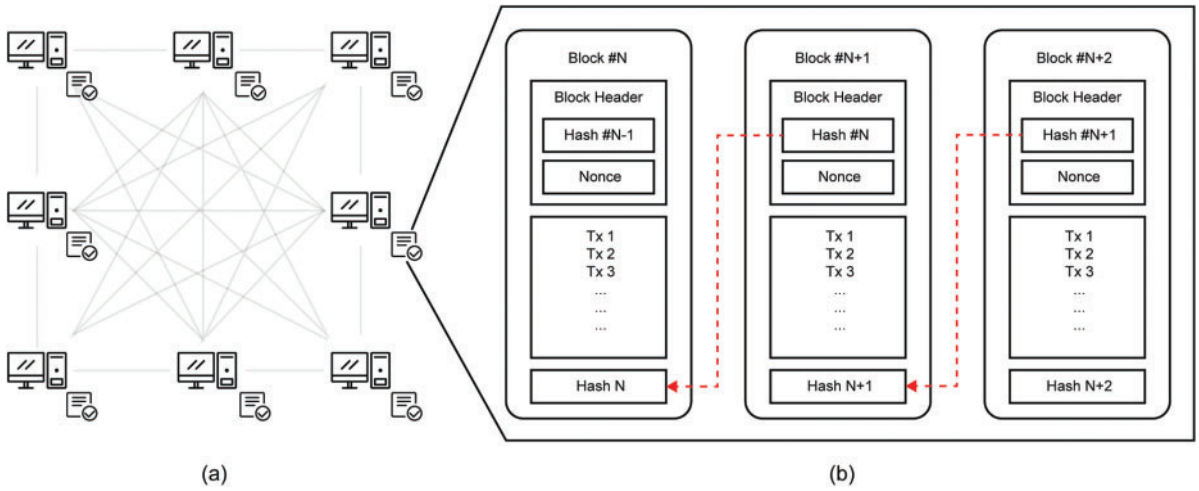

Figure 1: A general architecture of blockchain: (a) blockchain data is shared as a distributed ledger, (b) the data structure of each block on the blockchain (adapted from [1])

Blockchain has a consensus protocol that all network members must follow to determine some rules in a network that does not have a central authority. Thus, the network can coordinate and maintain data consistency without a central authority. In terms of how the authoritative party is selected, blockchain consensus can be divided into two types, i.e., performance-based and reputation-based. Table 1 provides a comprehensive overview of recent research focusing on performance-based and reputation-based blockchain consensus algorithms.

Table 1: Recent research on performance-based and reputation-based blockchain consensus algorithms

Consensus algorithm	Consensus type	Leader selection criteria	Public usage availability	Low energy usage	Fully decentralized	Consider multi variable	Involve machine learning	Key point
Proof of work (PoW), 2008 [2]	Performance-based	Cryptographic puzzle	✓	–	✓	–	–	PoW introduces computationally intensive cryptographic puzzles.
Proof of search (PoSE), 2019 [17]	Performance-based	Optimizing problem	✓	–	✓	–	–	PoSE incentive participants that solve optimization problems.
Proof-of-learning (PoL), 2019 [18]	Performance-based	ML model performance	✓	–	✓	–	✓	PoL incentive participants that solve machine learning tasks.
Proof of learning (PoLe), 2021 [6]	Performance-based	ML model performance	✓	–	✓	–	✓	PoLe proposed secure mapping layer to nodes cheating.
Proof of deep learning (PoDL), 2019 [19]	Performance-based	ML model performance	✓	–	✓	–	✓	PoDL proposed a 2-phase training method to prevent model stealing.
Separate proof of deep learning (SPoDL), 2021 [20]	Performance-based	ML model performance	✓	–	✓	–	✓	Implement PoDL for multi-access edge computing use case.
Deep learning-based consensus (DLBC), 2019 [21]	Performance-based	ML model performance	✓	–	✓	–	✓	Implement a DNN watermark to protect the ownership of models.
Proof of stake (PoS), 2019 [7]	Reputation-based	Total stake	✓	✓	–	–	–	Participants can propose blocks based on the ownership of a stake.
Delegated proof of stake (DPoS), 2018 [8]	Reputation-based	Delegated total stake	✓	✓	–	–	–	Enhances PoS scalability by relying on a smaller set of delegated nodes.
Proof of authority (PoA), 2017 [9]	Reputation-based	Authority	–	✓	–	–	–	Validators are chosen by identity to prioritize transaction finality.
Proof of space (PoSpace), 2015 [10]	Reputation-based	Disk space	✓	✓	–	–	–	Participants demonstrate the disk space to propose a new block.
Proof of elapsed time (PoET), 2017 [11]	Reputation-based	Waiting time	–	✓	–	–	–	Block proposers are selected by randomly-assigned wait time.

(Continued)

Table 1 (continued)

Consensus algorithm	Consensus type	Leader selection criteria	Public usage availability	Low energy usage	Fully decentralized	Consider multi variable	Involve machine learning	Key point
Proof of burn (PoB), 2020 [12]	Reputation-based	Total burned coins	✓	✓	–	–	–	PoB destroys cryptocurrency coins as a method to pick block leaders.
Proof of weight (PoWeight), 2017 [13]	Reputation-based	Weight	✓	✓	–	–	–	Similar to PoS. The participants' stake is called weight.
Dynamic reputation-based consensus mechanism, 2022 [22]	Reputation-based	Node reputation evaluation	–	✓	–	✓	–	Introduced a reputation mathematical model to monitor validators' behavior on PoA.
Reputation-based sharding consensus, 2022 [23]	Reputation-based	PBFT's primary election	–	✓	–	–	✓	Implement affinity propagation into consensus to determine the optimum number of intra-shards.
Trust-based shard distribution (TBSD), 2019 [24]	Reputation-based	Node trust evaluation	–	✓	–	–	✓	Implement genetic algorithm into consensus to compute shard distribution set on a blockchain.
Secured trust-based delegated consensus, 2022 [25]	Reputation-based	Node trust evaluation	–	✓	–	–	✓	Implement deep reinforcement learning to compute optimum blockchain parameters.
Mobility-aware reputation-based blockchain framework (MRBF), 2023 [26]	Reputation-based	Mobile node's reputation	–	✓	–	✓	✓	Focused on problems associated with assigning blockchain consensus tasks to mobile nodes.
Pharmaceutical-practical byzantine fault tolerance (P-PBFT), 2023 [27]	Reputation-based	PBFT's primary election	–	✓	–	–	–	Introduced nodes partition mechanism based on response speed.
Proposed consensus (PoIR)	**Reputation-based**	**Node's network reputation**	✓	✓	✓	✓	✓	**Introduced BiLSTM with NERD for the node selection mechanism.**

2.2 Performance-Based Blockchain Consensus

PoW was introduced in 2009 and applied to Bitcoin. Since Bitcoin was introduced, PoW has become a popular and widely used consensus. PoW works based on nodes' computational power. Every member wanting to participate to become a block proposer in the system must participate in a competition to solve cryptographic hashes. Members that have won the competition to complete or find the cryptographic puzzle and propagate the block that meets the requirements will receive a predetermined reward or transaction fee. Then the block will be stored on the blockchain network [28]. Finding a hash target uses immense computing power and is stochastic. Resolving this issue is similar to a miner coming across gold. Searching for a target hash is called mining, while members who complete it are called miners [29]. Each member participating in the hash target search competition uses computing power individually. Thus, each member does mining, and only one member is considered the winner, which makes PoW consume much energy [30,31]. The workflow example of performance-based blockchain consensus is shown in Fig. 2.

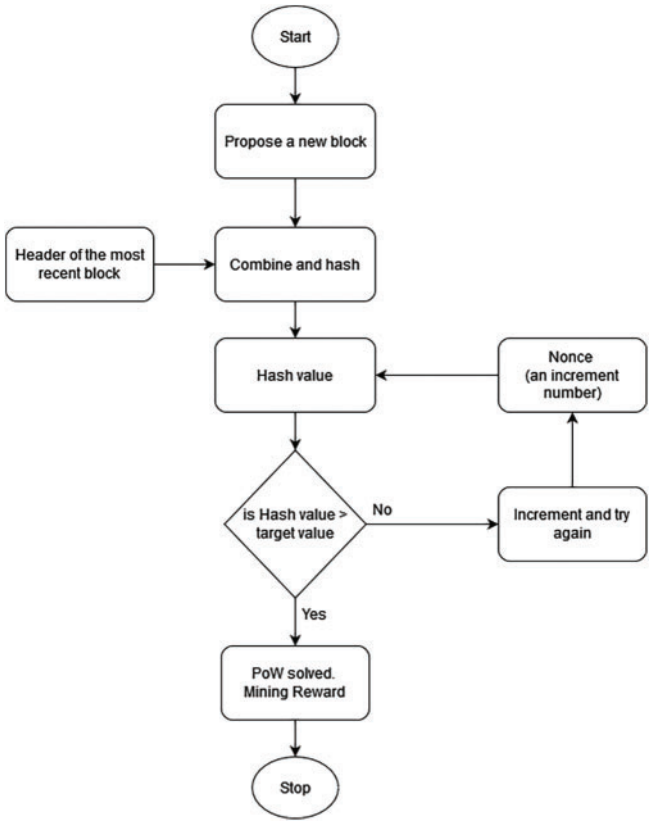

Figure 2: Workflow of PoW, an example of performance-based blockchain consensus (adapted from [33])

Another performance-based consensus proposed is PoSe (Proof-of-Search). Similar to PoW, PoSe uses the expended computing effort to determine block leaders. Instead of using computing power to solve cryptographic hashes, they use it to find solutions to optimization problems [17]. PoSe works on requests from clients containing to find solutions to optimization problems. The client supplies a search program that uses any search algorithm that determines a good outcome by assessing many

potential solutions. The node that discovers the closest solution will get a reward from the client. As a result, the issue of PoW competition can be moved to a more beneficial scenario.

Bravo-Mazquez et al. proposed a performance-based consensus named Proof-of-Learning (PoL) [18]. PoL is a concept for validating blocks in blockchain networks inspired by machine learning competition events, i.e., Codalab and Kaggle. This kind of competition helps advance machine learning development with relevant tasks such as recommender systems, image recognition, voice recognition, and many more. PoL can establish a mutually beneficial relationship between two different complex technologies, i.e., reCAPTCHA [32]. reCAPTCHA is a concept that connects two different tasks: a system for finding out if a web user is a real person (for safety) while helping to turn printed material into digital form. In extending Proof of Learning, other consensus algorithms are based on deep learning [6,19–21]. In a deep learning-based consensus, the energy that blockchain transactions should consume is shifted as deep learning computing tasks. The consensus calls for miners to conduct deep learning training tasks and provide a trained model as proof.

2.3 Reputation-Based Blockchain Consensus

PoW's need for enormous computational demand has ignited researchers to develop a new approach. In the alternative approach, instead of proving its computational works, the network members only need to provide their reputation, which does not require enormous energy. This approach is often referred to as reputation-based consensus. The following are some examples of reputation-based consensus:

The first PoW alternate consensus is Proof of Stake (PoS) [7]. PoS is an alternate consensus system offering the benefits of PoW while eliminating some of its drawbacks. To participate in PoS, a participant must own a set amount of cryptocurrency for verifying trades on the system. For example, 20% of the stake (coins) gives a participant a 20% possibility of mining the next block. The PoS consensus protocol has the following block proposal mechanism: validators, also known as staking nodes, stake a particular amount of currency to be allowed to participate in the block creation process. The protocol chooses the entity or node that holds the most stake to mine the next block. With this mechanism, PoS can eliminate cryptographic hash computations and reduce the amount of high computational power PoW. However, PoS is at risk of being exposed to a 51% attack if one individual owns more than half of the total cryptocurrency on the network, thus allowing them to propose a block every round. The workflow example of reputation-based blockchain consensus is shown in Fig. 3.

Another reputation-based consensus that has been proposed is Proof of Elapsed Time (PoET) [11]. PoET, proposed by Intel, does not use computing resources or cryptocurrency staking to participate in the competition but instead uses a random back-off scheme to compete [34]. Generally, the nodes participating in the process ask for a randomly assigned waiting time. The node that has the shortest waiting time is the one selected to be the block leader. To propose a block, the proposer of the block must show that the time he has is the shortest, then propagate the block before the waiting time runs out. With PoET, all members can become block proposers equally. Unfortunately, this consensus does not have good scalability potential, so it is unsuitable for public blockchains. Besides that, PoET must use Intel-specific devices.

PoS has been developed further into Delegated Proof-of-Stake (DPoS) [8]. DPoS is an election process similar to a board of directors, in which the network member selects a restricted number of people. They have the authority granted to them by their constituents to exercise their rights. Voting rights on DPoS are determined by the set amount of cryptocurrency at stake, and validators are selected using a replacement and election system. Invested funds used for voting rounds are secured via

smart contracts. The group selected as a delegate will maintain the network and its operations, making blocks and validating transactions. The Block Producers (BPs) receive a suitable reward in return for their work. In addition, DPoS has a backup concept to choose a group of BP that will replace the main BP. If something goes wrong, BP backups are also given a reward but smaller.

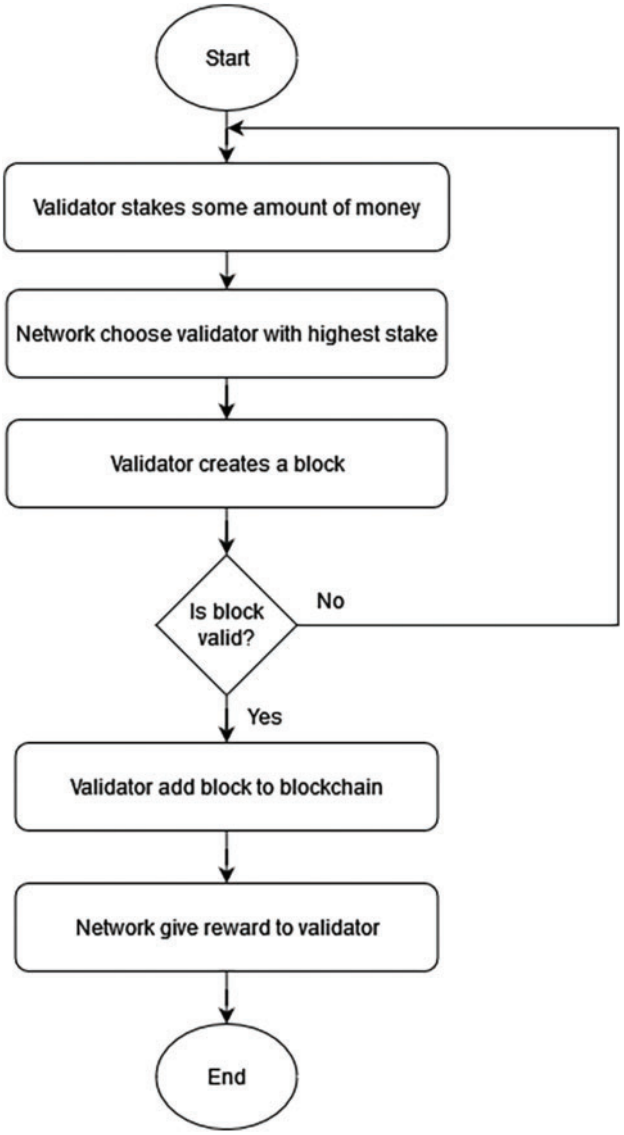

Figure 3: Workflow of PoS, an example of reputation-based blockchain consensus (adapted from [33])

Besides DPoS, PoS was also developed into Proof-of-Authority (PoA) [9]. PoA was initially developed for a private network on Ethereum. In PoA, selected nodes called validators are responsible for creating blocks and validating transactions. The network can handle lots of transactions because only a few validators are needed, making it highly scalable and with nearly no cost to process. Unlike PoS, a validator does not need to put any of their assets at stake but instead their reputation. If a reputation is established, which takes time from participating in the network, the problem of PoS can be solved. In PoS, the participant with the most stakes will receive the biggest reward in the network.

However, PoA still leans towards a centralized system, as the entire network is under the authority of a few members.

The following consensus that is developed from PoS is Proof of Burn (PoB) [12]. PoB protocol deliberately destroys cryptocurrency coins as a method to pick block leaders, in contrast to how PoW works, which depends on physical resources to conduct mining operations. Block validation in PoB-based networks does not require extensive computational resources. Thus, powerful mining hardware such as ASICs is not needed. As a substitute for the currency that has been burned, PoB replaces it with a virtual mining system. By sacrificing short-term losses, a node can demonstrate its commitment to the network for long-term benefit. Sending coins to a predetermined burn address will cause them to be destroyed and not recoverable.

Proof of Weight (PoWeight) [13] is also the development of PoS. Similar to PoS, the stake owned by the participant is called "weight" instead of "stake". Algorand is one of the digital currencies that implements PoWeight. PoWeight will stay safe from double spending if the majority of users in the system are honest. Each PoWeight block creation round will randomly select a committee of network members and assign tasks based on the amount of digital currency they have. The concept is similar to PoS, but the random formation of committees makes the selection of block proposers evenly distributed so that they are not vulnerable to 51% attacks.

Proof-of-Space (PoSpace) is a development carried out by [10]. PoS requires members to dedicate a certain amount of storage space or memory to solve problems issued by the system. A system requesting a specific space called a Prover will send a PoSpace specification to the Verifier indicating that the Prover reserved the specific space. For example, if someone needs a certain amount of disk space, the Verifier will check it for accuracy with a particular procedure. Validation of a certain amount of disk space must be done accurately and quickly but not take up a lot of disk space.

The dynamic reputation-based consensus mechanism [22] proposed mainly includes the nodes group generation algorithm, the validators group generation algorithm, node behavior discrimination, and the reputation evaluation algorithm. Previously, in PoA consensus, the blockchain selects several authorities to validate the ledger, but authorities may not be trusted entirely. Therefore, this scheme assigns a reputation value to each node through a reputation evaluation model. Each validator performs block packing by pledging its reputation value, and all nodes can complete the reputation accumulation by maintaining the blockchain. The trustworthiness of the validators is ensured by selecting the nodes with high reputation values to become the validator group.

A Reputation-based Sharding Consensus model applied to information-centric networking [23] has been proposed. This consensus optimizes by using Affinity Propagation (AP) algorithms and multi-dimensional trust to divide network members into shards. Thus, it can minimize the occurrence of validation errors from security and provide real-time processes that the client can see directly.

The Trust-Based Shard Distribution scheme [24] considers the reputation of blockchain nodes, which utilizes genetic algorithms to maximize security in a fault-tolerant Shard blockchain network. This multi-dimensional reputation model calculates reputation values depending on trust parameters such as Quality of Security Performance (QoSP), Quality of Service (QoS), and the set of reputations in the data.

To extend the trust-based shard distribution scheme, TDCB-D3P has been proposed [25]. TDCB-D3P is a secure trust-based delegated consensus blockchain with a Dueling Double Deep-Q network with Prioritized Experience replay (D3P). This system improves the blockchain's capabilities by utilizing deep reinforcement learning technology.

The Mobility-aware Reputation-based Blockchain Framework (MRBF) [26] is a novel solution addressing the stability issues of Mobile Consensus Nodes (MCNs). It utilizes the Practical Byzantine Fault Tolerance (PBFT) algorithm and introduces the Intelligent Consensus Node Selection Algorithm (ICNS) based on deep learning. MCNs are selected using live and historical reputation information, considering factors such as the Age of the Node (AoN), Stability of the Node (SoN), and Success Rate of Block Proposals (SoB). MRBF aims to improve the stability of MCNs by leveraging reputation-based selection mechanisms and enhancing the overall performance of consensus in mobile environments.

A novel consensus algorithm called Pharmaceutical-Practical Byzantine Fault Tolerance (P-PBFT) [27] is presented for large-scale pharmaceutical traceability systems. P-PBFT incorporates a grouping strategy based on response speed among supply chain nodes to form independent consensus sets, ensuring high security and low communication overhead. This approach claims to enhance network scalability and aligns with the requirements of pharmaceutical supply chains with numerous participating nodes.

2.4 Network Entity Reputation Database System

Bartos et al. [15] have introduced the Network Entity Reputation Database (NERD), a system for evaluating the reputation of network entities. NERD gathers information related to cybersecurity threats from multiple sources and creates a database that is updated regularly with details of known network entities that have malicious intent. This information is gathered from various sources that detect suspicious activities in networks. These various sources include:

1. Warden

CESNET (Czech Education and Scientific Network) operates a system called Warden that facilitates the sharing of alerts. The alerts are collected from honeypots, NetFlow analyzers, and other malicious traffic detectors, primarily utilized in Czech academic networks. Over a million alerts from approximately a dozen detection systems are exchanged daily through this platform.

2. Blacklists

To create its database, NERD relies on approximately 50 public blacklists from about 20 providers. The database records the details of any IP address that is listed in these blacklists.

3. DShield (Distributed Intrusion Detection System)

The SANS (SysAdmin, Audit, Network, and Security) Institute operates a collaborative system called DShield, which involves correlating firewall logs. A community of contributors shares data related to unanticipated incoming connections and statistics about individual Ips and ports.

4. AlienVault Open Threat Exchange (OTX)

OTX is a platform that enables a collaborative community of researchers and security professionals to share millions of threat indicators daily.

5. MISP (Malware Information Sharing Platform)

NERD is linked to a sizable MISP run by CIRCL (Computer Incident Response Center Luxembourg), a platform for sharing information about internet threats.

The NERD framework, as proposed by Bartos et al. [15], is shown in Fig. 4. Each IP address is assigned a reputation score, a single number summarizing the level of threat the address poses, based on the number of events and the number of distinct sources which reported it within the last 14 days.

The score takes values between 0.0 (no events) and 1.0 (worst case). The NERD system implements a scoring mechanism based on the following calculations:

1. Every day within 14 days, calculate the:
   a) *n_events*(*d*)-Number of events reported to Warden within the day with the evaluated IP address listed as the source.
   b) *n_nodes*(*d*)-Number of distinct nodes (detectors) that reported those alerts.
   c) Daily reputation score. The daily reputation score can be calculated with Eq. (1).

$$\text{rep (d)} = \left(1 - \frac{1}{2^{n\_events(d)}}\right) \times \left(1 - \frac{1}{2^{n\_nodes(d)}}\right) \tag{1}$$

2. The final reputation score is the weighted average of the 14 daily scores with decreasing weight linearly (the last day has the highest weight to show favoritism to the recent days' rep (d)). The final reputation score formula can be calculated with Eq. (2).

$$\text{rep} = \sum_{d=0}^{d=13} rep\,(d) \times \frac{14 - d}{14} \div 7.5 \tag{2}$$

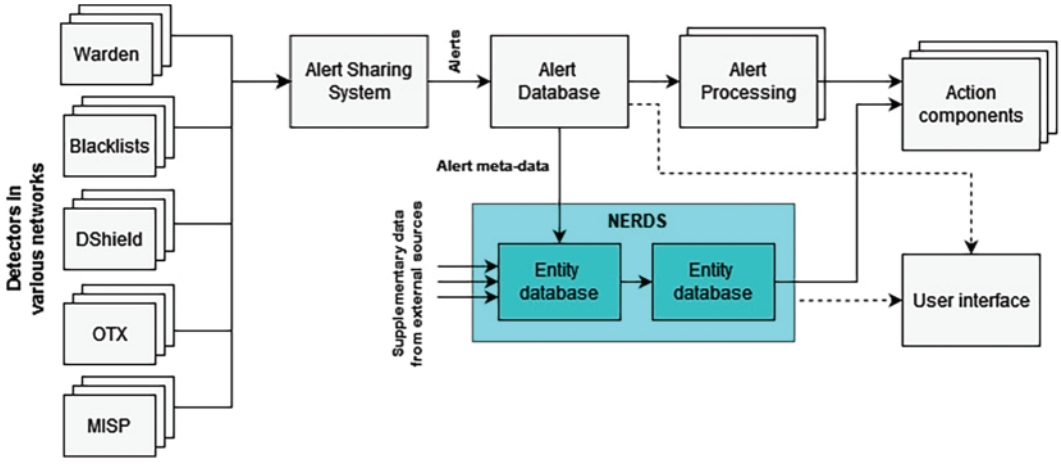

Figure 4: Network entity reputation database framework (adapted from [15])

3 Proposed PoIR Consensus

This section describes the proposed work, including PoIR blockchain architecture, node selection algorithm, and BiLSTM model for node selection implemented using the NERD dataset.

3.1 PoIR Blockchain Architecture

The proposed PoIR blockchain architecture is shown in Fig. 5. There are two main components in PoIR: Machine Learning Component and Blockchain Network Component. PoIR consensus begins by training the NERD dataset using machine learning with the BiLSTM architecture, producing a trained BiLSTM model. Furthermore, in the blockchain network component, the trained model will be integrated into the Intelligent Node Selector (INS) on the blockchain network. Thus, when new

members join the network, they will be selected by INS and given a reputation score to determine the class of members. They can become miners, validators, or common nodes if they meet the requirements.

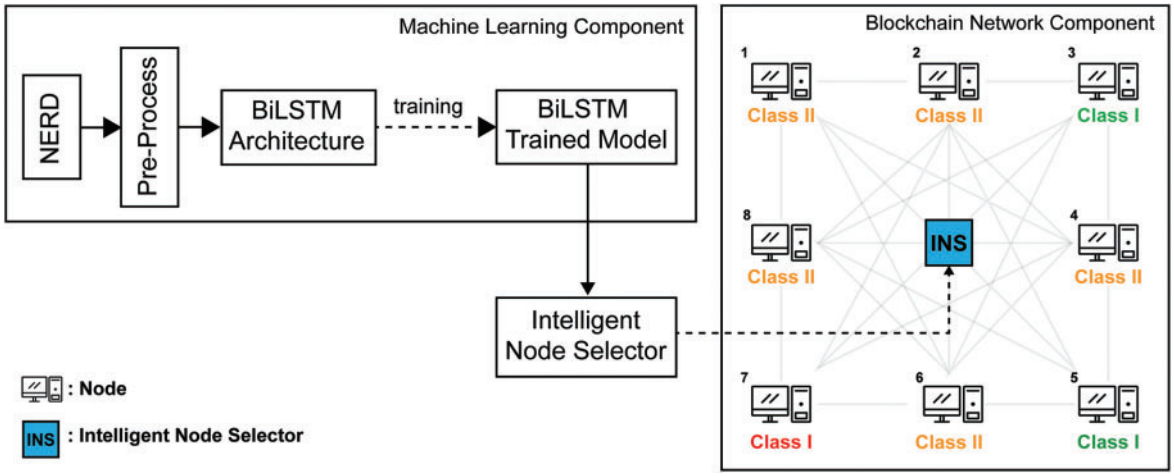

Figure 5: Proposed PoIR system workflow

PoIR implements a peer-to-peer architecture that is commonly used in distributed ledger systems. In this architecture, each network entity, or node, has the same capabilities to communicate directly. PoIR adopts a public blockchain architecture in which the data and access to the system are available to anyone willing to participate in the network. However, unlike PoW, PoIR will monitor the nodes' NERD reputation before its connection to the blockchain network and will be constantly monitored on each block proposal cycle. The reputation will represent nodes' behavior and intention. By monitoring the reputation of the nodes periodically, it is hoped that only behaved and honest nodes will connect to the PoIR network.

There are four main objects in the PoIR network, i.e., miner node, validator node, common node, and node selector. The miner node objective is to propose a new block to the blockchain network. Unlike performance-based PoW, the miner node will not compete with other nodes to add the following block of transactions to the blockchain, which draws enormous computational resources. The validator nodes' objective is to validate the miner's format of a new proposed block and check the ledger's integrity. Validator nodes are also responsible for distributing the confirmed ledger to the blockchain network. The common node is only eligible to join the network and read the distributed ledger. However, it should be noted that the common node's higher goal is to become a validator or miner node, which can be achieved by boosting its reputation score. The node selector acts as an investigator of nodes' behavior. It monitors nodes' behavior by multi-variable features, which are represented by a reputation score. As shown in Fig. 5, the reputation is divided into clusters, which denotes nodes' eligibility to perform tasks, i.e., mining and validating block. The class division and threshold will be described in Section 3.2.

3.2 Intelligent Node Selector Mechanism

Intelligent Node Selector (INS) is used two times: node join request and new block proposal. Each scenario is described in Figs. 6a and 6b and further described through Algorithms 1 and 2. The nodes judged by INS are divided into three clusters by their reputation score threshold. Table 2 describes the reputation score threshold of each node class.

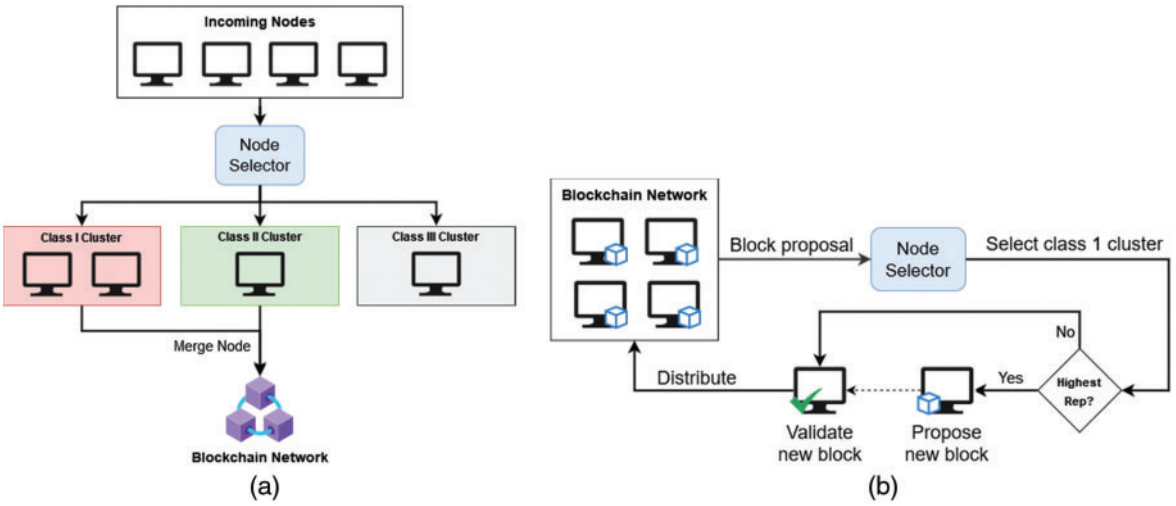

Figure 6: Node join proposal in PoIR (a) and new block proposal in PoIR (b)

Table 2: PoIR class clustering by reputation score threshold

Class cluster	Reputation score threshold
Class I	reputation > 50
Class II	25 ≥ reputation ≥ 50
Class III	<25 reputation

The specific requirements of reputation clustering in PoIR were considered. Generally, the determination of thresholds in clustering regression values into classes will be evenly divided according to the number of classes. For example, in the PoIR case, the classes will be typically divided into two: scores greater than 50 will fall into the excellent node class, and scores less than 50 will fall into the lousy node class. However, the thresholds were tweaked to align with the desired behavior of nodes within the network and to provide meaningful differentiation in terms of trustworthiness and reliability. Assuming the network covers all possible reputation scores, nodes that will be allowed to participate are 75% of the total reputation score range (25–100). This range is intended to involve more nodes. By involving more nodes, the INS will be less discriminatory, and the PoIR network will be moderately inclusive while still being protected from malicious nodes (Class III, under 25 reputation score).

Furthermore, the network classes will be divided into two to ensure the network operates securely. Only nodes with reputation scores more than 50 (Class I, following the general threshold determination principle) can fully operate the blockchain (propose new blocks, manage nodes). The remaining nodes (Class II, 25–50 reputation score) will only act as ledger keepers operated by Class I nodes. The relationship between Classes I and II nodes can be depicted as a pseudo-master-slave configuration. However, the relationship becomes dynamic as nodes' reputation is periodically updated. This dynamic relationship enables nodes from the Class II cluster to transcend into Class I nodes and operate the blockchain. Considering those issues, the INS will only merge Class I and Class II node to the blockchain network, while Class III is denied as its reputation is considered too low.

Algorithm 1: Node join proposal on PoIR

```
Input: node's network information
Output: inverted reputation score
candidate ← Node()
candidate.initiate_instance{
    get ip,
    get hostname,
    get asn,
    get country,
    get events,
    get event_nodes,
    get event_categories,
    get event_categories_c,
    get event_categories_w,
    get blacklists,
    get blacklists_c,
    get blacklists_w,
    get tags,
    get tags_c,
    get tags_w
  })
If candidate.get_reputation() ≥ 25
  node[new_id] ← candidate
  copy current network's ledger to node[new_id]
Else
  reject candidate
EndIf
```

For the new block proposal request, the node selector will only select the node from Class I. Suppose the node proposes a new block with the highest reputation. In that case, the network will allow the node to become the miner and send the proposed block to be validated by selected validator nodes. However, it is acknowledged that only selecting the node with the highest reputation will lead to a centralized block proposer. Therefore, the round-robin rule is applied. Node from Class I, which already proposed a new block, will not be selected as the leader again until other Class I nodes propose their blocks. Furthermore, the selected validators can vote to approve or reject the proposed block. If all validators approve, the proposed block will later be distributed to all network's ledgers.

Algorithm 2: New block proposal on PoIR

```
Input: messages
Output: Block proposed to network
If  queue_leader is empty
  node.update_reputation()
```

(Continued)

Algorithm 2 (continued)

```
  leader_queue ← get_leaders_queue()
EndIf
leader_id ← first element of leader_queue
remove leader_id from leader_queue
node[leader_id].start_mining(messages)
For each node as node_id do
    If (node[node_id].get_reputation() ≥ 0.25) and (node_id not
    leader_id):
     node[node_id].validate_blockchain()
    EndIf
EndFor
For each node as node_id do
    If (node_id not leader_id):
      node[leader_id].send_blockchain_to_neighbour(node_id)
    EndIf
EndFor
```

3.3 BiLSTM Model for Node Selection

The BiLSTM trained with the NERD dataset is applied as a model for developing the node selection mechanism. The following subsection will describe how the dataset is created, the model's architecture, and the model's training process, respectively.

3.3.1 Network Entity Reputation Dataset

To train the BiLSTM, NERD as the dataset is adopted. The dataset is derived from NERD's public website. It consists of 141.070 data rows containing several network entity properties with its initial reputation score. The example rows of the derived dataset are shown in Table 3.

Table 3: Example of derived NERD dataset

Column	Value
ip	94.102.61.39
hostname	security.criminalip.com
asn	202425
country	GB
events	3272
event_nodes	4
event_categories_c	2
event_categories	AnomalyTraffic | ReconScanning
blacklists	blocklist_net_ua_ips | ciarmy | dshield | turris_greylist
rep_score	0.935707

Before using the dataset, the statistics from the NERD dataset were evaluated. Fig. 7 describes some of the dataset's feature distribution, which is depicted through the Kernel Density Estimate (KDE) diagram. KDE visualizes the distribution of observations in a dataset. It reviews the joint distribution of pairs of columns from the dataset. In the NERD dataset, entities are differentiated based on their IP address values, representing individual PoIR network nodes. The top row of the figure elaborates on the relationship between IP address and other features. It can be observed that the dataset shows a fairly even distribution, i.e., the KDE between IP address and reputation score (row 1, column 8). The IP addresses are evenly distributed across the reputation range from 0.0 to 1.0, indicating that the entities in the NERD dataset represent all class clusters.

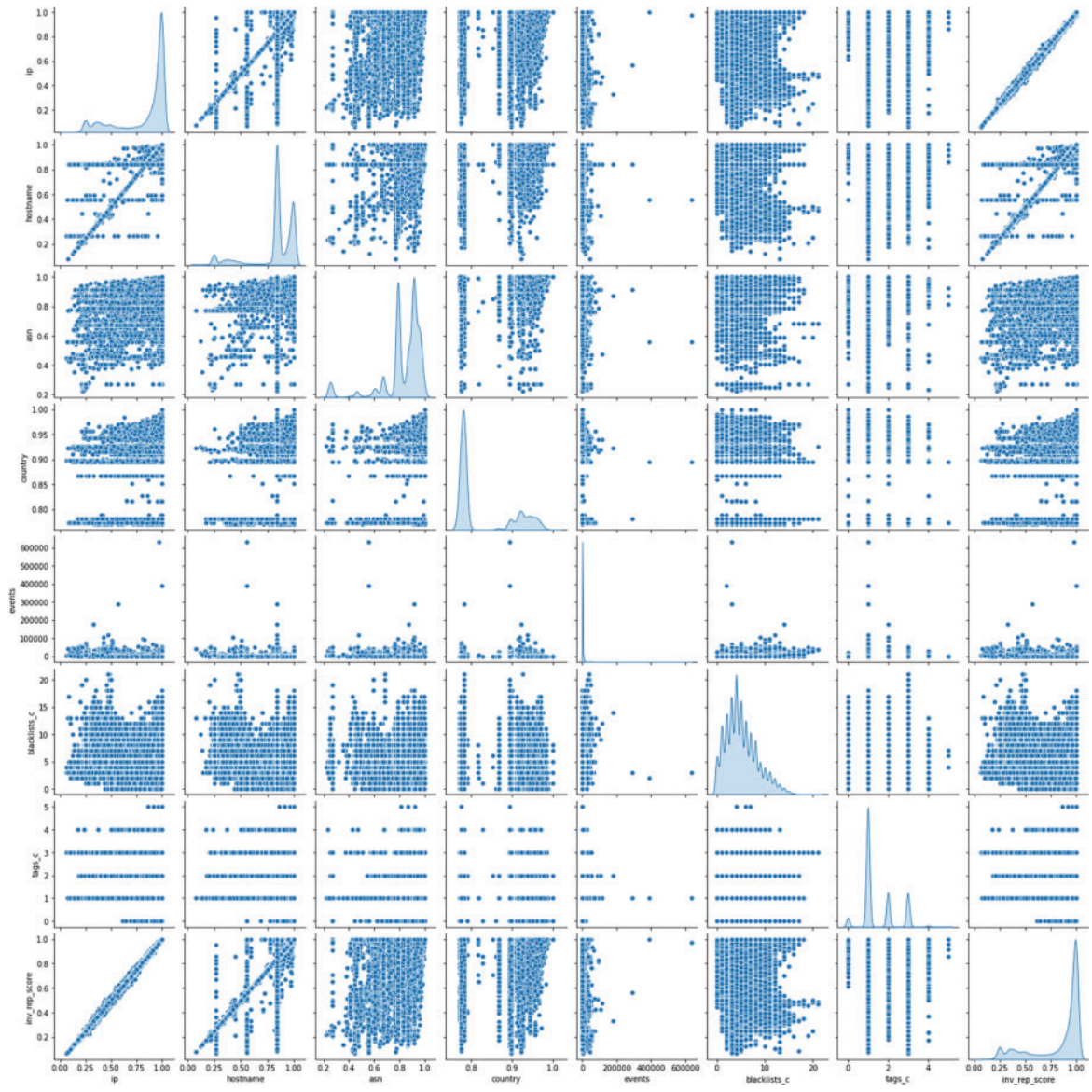

Figure 7: Selected feature distribution of derived NERD dataset

However, some outliers are detected in the KDE diagram for the IP address and events (row 1, column 5). Typically, the total number of events in the NERD dataset ranges from 0 to 116266. Nevertheless, four entities stand out with unusually high event counts. These entities recorded 177986, 288894, 390378, and 633256 events, respectively. Although these outliers are relatively insignificant in the overall dataset, they appear as distinct data points that deviate significantly from the primary trend. These outliers may arise due to rare occurrences or instances that deviate from the expected patterns in the data. Further, the check reveals that these outliers correspond to the Netherland IP addresses, specifically those associated with the hostname recyber[dot]net, and were tagged for ReconScanning and AttemptLogin activities.

The dataset was adjusted to fit the PoIR context better. Some adjustments and pre-process of the derived dataset are described as follows:

1. Invert the reputation score

As NERD's reputation score hierarchy indicate the opposite of PoIR (in original NERD, 1.0 shows the worst value, and 0.0 shows the best value), the score was inverted with the formula in Eq. (3).

$$rep_{inverted} = 1 - rep \tag{3}$$

The formula in Eq. (3) is implemented on pseudocode as described in Algorithm 3.

Algorithm 3: Reputation score inversion

```
Input: dataset
Output: inverted reputation score
declare dataset['inv_rep_score']
For i=0 to length of dataset do
  dataset['inv_rep_score'] ← (1 – dataset['rep_score'])
EndFor
```

2. Create additional features

Additional features were added to extend the better meaning of NERD's dataset. The additional features include the count and weighted count of the blacklist, event categories, and tags. The process of each features addition is described in Algorithm 4. The count represents the number of items shown in the respective feature. For example, if a node has a blacklist of blocklist_net_ua_ips|ciarmy|dshield|turris_greylist, it has four counts of the blacklist. Furthermore, the weight represents the total occurrence of the blacklist on the respective node divided by the total blacklist available.

Algorithm 4: Features addition

```
Input: dataset
Output: count and weighted count of blacklist, event_categories, tags
    # Create dictionary
declare dictionary
declare total ← 0
For i=0 to length of dataset do
  current_list ← split dataset[column name][i] by '|' character
  For j=0 to length of current_list do
    If (current_list[j] not in dictionary and not empty):
```

(Continued)

Algorithm 4 (continued)

```
      dictionary [current_list[j]] ← 1
      total ← total + 1
    ElseIf (current_list[j] in dictionary):
     dictionary [current_list[j]] ← dictionary [current_list[j]] + 1
     total ← total + 1
    EndIf
   EndFor
  EndFor
 # Insert count and weight column into dataset
  declare dataset[column name + '_w']
  declare dataset[column name + '_c']
  For i=0 to length of dataset do
    current_list ← split dataset [column name] [i] by '|' character
    current_weight ← 0
    current_count ← 0
    For j=0 to length of current_list do
      If current_list[j] in dictionary:
        current_weight←current_weight + dictionary[current_list[j]]
        current_count ← current_count + 1
      EndIf
    EndFor
    dataset[column name + '_w'][i] ← current_weight/total_tags
    dataset[column name + '_c'][i] ← current_count
  EndFor
```

3. Encode the data

It is noted that some of the dataset features are categorical, i.e., ip_address, hostname, asn, and country. The data was encoded into numerical values using mean encoding to make the data trainable by BiLSTM. The mean encoding uses the target variable (inverted reputation) as the basis to generate the new encoded feature. The mean encoding steps are described as follows:

a) Define categorical features to transform.
b) Group the dataset by the categorical feature and get the total sum of the feature, i.e., hostname.
c) Group the dataset by the categorical feature and get the number of respective features.
d) Divide the step b to step c results.
e) Merge the results of step d into the dataset.

Furthermore, the pseudocode of data encoding is described in Algorithm 5.

Algorithm 5: Mean encoding of NERD

```
Input: ip_address, hostname, asn, and country
Output: encoded ip_address, hostname, asn, and country
    # Get Mean Encoding
 1: mean_ip_address ←
    dataset.groupby ('ip') ['inv_rep_score'].mean ().to_dict ()
 2: mean_hostname ←
    dataset.groupby ('hostname') ['inv_rep_score'].mean ().to_dict ()
 3: mean_asn ←
    dataset.groupby ('asn') ['inv_rep_score'].mean ().to_dict ()
 4: mean_country ←
    dataset.groupby ('country') ['inv_rep_score'].mean ().to_dict ()
    # Insert the Encoding results to dataset
 5: dataset['ip'] ← map dataset_slim['ip'] by mean_ip_address
 6: dataset['hostname'] ← map dataset['hostname']by mean_hostname
 7: dataset['asn'] ← map dataset['asn'] by mean_asn
 8: dataset['country'] ← map dataset_slim['country'] by mean_country
```

4. Normalize the data

Table 4 shows the overall statistics of the dataset. Each feature covers very different scales and ranges, i.e., events have a mean of 1420. Meanwhile, blacklist_w only has a mean of 0.33. For this matter, the dataset was normalized.

Table 4: Dataset overall statistics

Column	Mean	STD	Min value	Max value
ip	0.828874	0.243710	0.064293	1
hostname	0.828631	0.195476	0.079167	1
asn	0.828553	0.156616	0.221129	1
country	0.828468	0.072144	0.771206	1
events	1420.575834	6305.861395	0	633256
event_nodes	2.138585	1.037314	0	7
event_categories_c	1.538153	0.759570	0	7
event_categories_w	0.554199	0.196099	0	0.990556
blacklists_c	5.085041	3.312932	0	21
blacklists_w	0.331513	0.183023	0	0.831982
tags_c	1.490501	0.850971	0	5
tags_w	0.427537	0.211953	0	0.901950
inv_rep_score	0.828871	0.243732	0.064293	1

Normalization is essential as the features will be multiplied by the model weights. As a result, the scale of the inputs influences the scale of the outputs and the scale of the slopes. The Keras

Normalization module was used to normalize the dataset—the example of the before and after normalization vector is shown in Eqs. (4) and (5), respectively.

$$\text{Before Normalization} = [[0.98][0.98][0.91][0.92][51][1][1][0.53][2.][0.18][2][0.68]] \tag{4}$$

$$\text{After Normalization} = [[0.6][0.75][0.54][1.31][-0.22][-1.1][-0.71][-0.13][-0.93][-0.81][0.6][1.2]] \tag{5}$$

Further to the data adjustment process, the correlation between features and targets in the dataset and map it on the correlation heatmap diagram is analyzed and shown in Fig. 8.

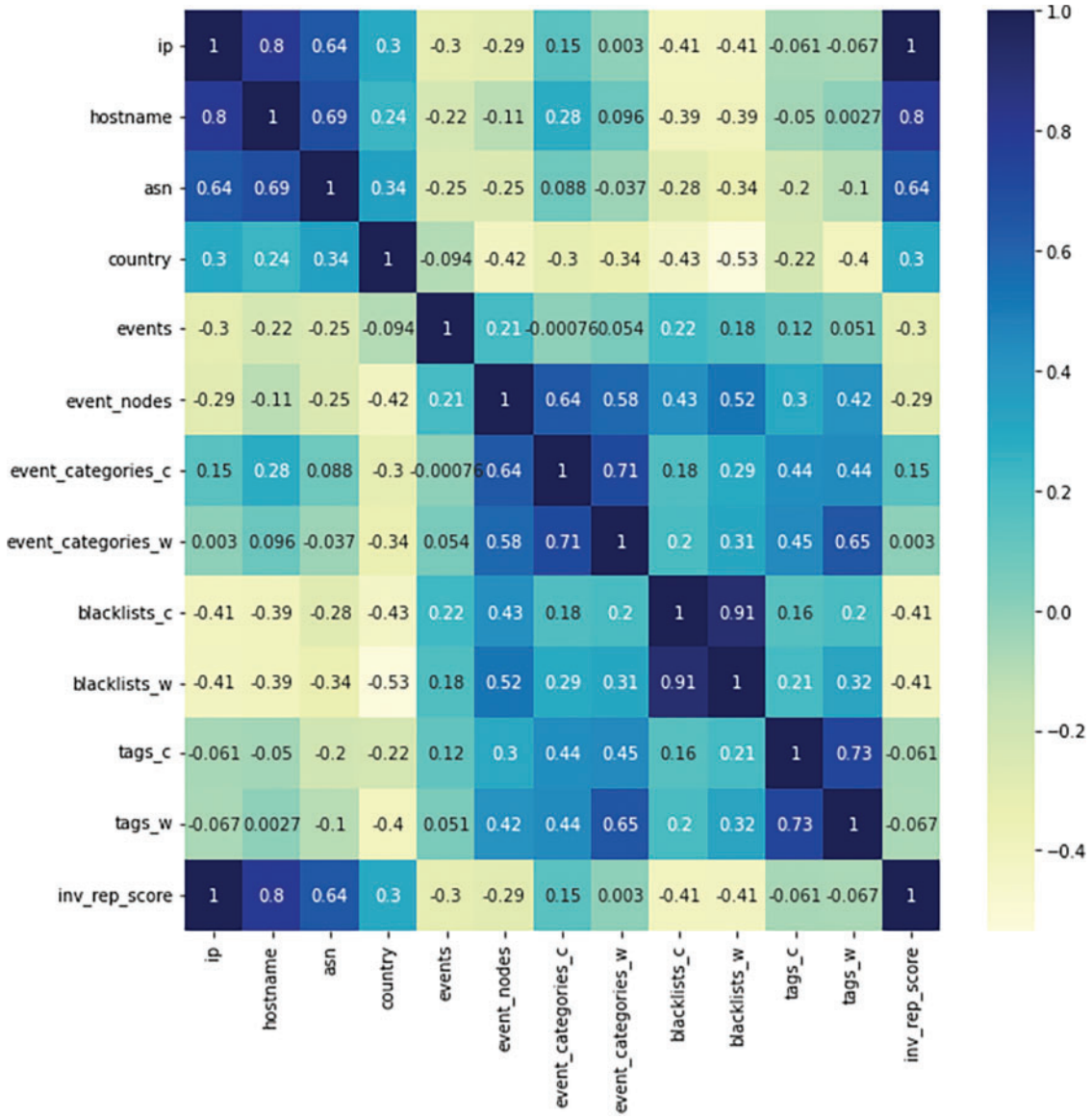

Figure 8: Correlation heatmap of derived NERD dataset

The correlation heatmap is calculated using the Pearson Correlation Coefficient formula. Pearson Correlation Coefficient measures the strength and relationship between features and targets. Pearson Correlation Coefficient is examined using the formula in Eq. (6).

$$r = \frac{n\left(\sum xy\right) - \left(\sum x\right)\left(\sum y\right)}{\sqrt{\left[n\sum x^2 - \left(\sum x\right)^2\right]\left[n\sum y^2 - \left(\sum y\right)^2\right]}} \tag{6}$$

In the given formula, n refers to the total number of pairs of stocks that are being analyzed. The symbols $\sum xy$, $\sum x$, $\sum y$, $\sum x^2$, and $\sum y^2$ represent the sum of products of paired stocks, the sum of all x scores, the sum of all y scores, the sum of the squared values of x scores, and the sum of the squared values of y scores, respectively.

3.3.2 Model Architecture and Training

The Bidirectional Long Short-Term Memory (BiLSTM) network architecture is adopted in this work. LSTM is a variant of Recurrent Neural Network (RNN) with the learning ability of long-term dependencies. Long short-term memory units are composed of four components: a cell, a forget gate, an input gate, and an output gate, distinct from recurrent neural networks. Within a predetermined period, the cell stores data, and gates manage the passage of information to and from the cell. This process consists of two states: a hidden state and a cell state. In an LSTM cell, the cell state serves as a form of memory, while the hidden state is responsible for storing the output of the cell. Thus, a hidden state and a cell input decide the information stored in the cell, i.e., to discard or store new information.

Fig. 9 depicts the architecture of the BiLSTM network, which is an extension of the traditional LSTM model featuring an additional layer. This second layer is connected through hidden-to-hidden linkages that allow information to flow in reverse temporal order, as shown in Fig. 9b. The model can incorporate information from past and future contexts during training. The forward and backward pass outputs are added element-wise to predict the output of the ith word, as indicated in Eq. (7).

$$h_i = h_i^f + h_i^b \tag{7}$$

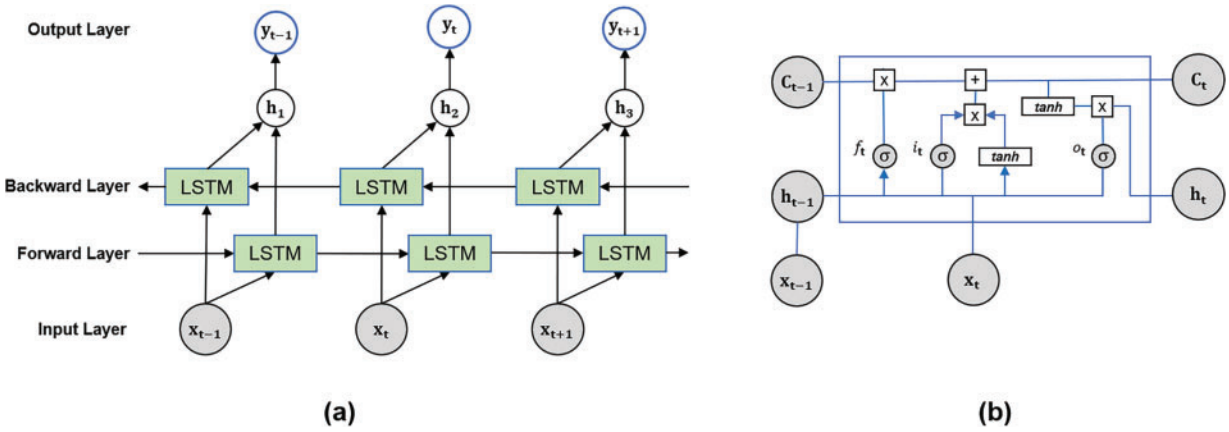

Figure 9: BiLSTM network architecture (a) and detailed LSTM cell (b)

The BiLSTM model consists of multiple LSTM cells, and the architecture of an LSTM cell is shown in Fig. 9a. At time step t, the forget gate, which implements a sigmoid function (σ), determines whether to retain or forget information in the cell state. To make this decision, it takes input x_t and

receives the output of the previous hidden state h_{t-1}, and uses Eq. (8) to calculate the probability of retaining current information.

$$f_t = \sigma(W^f x_t + U^f h_{t-1}) \tag{8}$$

Once the forget gate in the LSTM cell determines the probability of retaining new information, the LSTM cell decides whether to keep that information in the cell state. Additionally, the input gate, which is implemented using a sigmoid function, determines the values to be updated based on Eq. (9). Meanwhile, the tanh function generates a candidate vector $\hat{C}_t$, which is combined with the cell state to update it. Eq. (10) is used to generate this candidate vector.

$$i_t = \sigma(W^i x_t + U^i h_{t-1}) \tag{9}$$

$$\hat{C}_t = tanh(W^n x_t + U^n h_{t-1}) \tag{10}$$

Then, Eq. (11) is used to update the previous cell state C_{t-1} into a new cell state C_t.

$$C_t = f_t \times C_{t-1} + i_t \times \hat{C}_t \tag{11}$$

Eq. (11) uses the forget gate f_t to regulate the gradient passes through the LSTM cell, which helps mitigate issues of vanishing or exploding gradients commonly observed in recurrent neural networks. After the cell state C_t is updated, the sigmoid function at the output gate is used to determine which portion of the cell state should be returned, as specified in Eq. (12). This resulting cell state C_t is then passed through a tanh function and multiplied with another sigmoid function to produce the hidden state h_t as shown in Eq. (13).

$$o_t = \sigma(W^o x_t + U^o h_{t-1}) \tag{12}$$

$$h_t = o_t \times tanh(C_t) \tag{13}$$

The model architecture build and train process is described in Algorithm 6. There are three main activities in Algorithm 6. The first activity, which is shown in lines 1–27, is the architecture definition. This activity defines a function to create the model in bidirectional and unidirectional ways. A function to fit models with the dataset is also defined. In addition, models to train are defined, i.e., BiLSTM, BiGRU, UniLSTM, and UniGRU. The second activity in lines 28–31 explains the model's training process. The third activity, shown in lines 32–35, describes the model's save process.

Algorithm 6: Training process

```
Input: train dataset: X_train, y_train
Output: trained model
      # Define bidirectional model
  1:  Function create_model_bidirection(units, model_name):
  2:    model ← Sequential()
  3:    forward_layer ← model_name(units, return_sequences ← True)
  4:    backward_layer ← model_name(units, return_sequences ← True,
        go_backwards ← True)
  5:    model add Bidirectional layer using forward_layer and
        backward_layer
```

(Continued)

Algorithm 6 (continued)

```
    model add 0.1 Dropout layer
    model add Dense layer using 3 neuron and relu activation
    model add Dense layer using 1 neuron as output
    model compile with root_mean_squared_error and Adadelta 0.25
    clipnorm optimizer
     return model
 EndFunction
# Define unidirectional model
 Function create_model_unidirection(units, model_name):
   model ← Sequential()
   model add model_name(units, return_sequences ← True)
   model add 0.1 Dropout layer
   model add Dense layer using 3 neuron and relu activation
   model add Dense layer using 1 neuron as output
   model compile with root_mean_squared_error and Adadelta 0.25
   clipnorm optimizer
   return model
 EndFunction
# Define fit models function
 Function fit_model(model_name):
   model_name fit using X_train, y_train, epochs ← 100,
   validation_split ← 0.2
 EndFunction
# BiLSTM, BiGRU, UniLSTM & UniGRU
model_bilstm ← create_model_bidirection(6, LSTM)
model_bigru ← create_model_bidirection(6, GRU)
model_lstm ← create_model_unidirection(6, LSTM)
model_gru ← create_model_unidirection(6, GRU)
# Train models
bilstm ← fit_model(model_bilstm)
bigru ← fit_model(model_bigru)
lstm ← fit_model(model_lstm)
gru ← fit_model(model_gru)
# Save trained models
bilstm.save('poir_bilstm')
bigru.save('poir_bigru')
lstm.save('poir_lstm')
gru.save('poir_gru')
```

4 Results and Analysis

This section discusses the BiLSTM model evaluation, monopoly attack evaluation, and transaction time evaluation. The evaluation environment is described in Table 5. The models are implemented and evaluated using Python 3.7. Google Colaboratory platform [35] is used to develop the BiLSTM model and PoIR blockchain architecture.

Table 5: Evaluation environment description

Parameters or variables	Values
Hardware specification	
a. CPU model name	Intel(R) Xeon(R) 2.30 GHz
b. RAM	12 GB
c. GPU	Nvidia K80
d. GPU memory	12 GB
Models evaluation parameter	
e. Epoch	100
f. Train-validation ratio	8:2
g. Optimizer	Adadelta
h. Evaluation dataset	14000 rows
i. Loss evaluation metric	Root mean squared error
j. Models comparison	BiLSTM, UniLSTM, BiGRU, UniGRU
Blockchain evaluation parameter	
k. Block proposal round	1000 iteration
l. Consensus comparison	PoW, PoS
m. Messages per proposal	Total of 8000–10000 randomized messages

4.1 Model Evaluation

Approximately 20% of the total rows of the training dataset are used for validation in the training phase. To evaluate the performance of the model, BiLSTM is compared with other Recurrent Neural Network models, i.e., Unidirectional LSTM (UniLSTM), Bidirectional Gated Recurrent Unit (BiGRU), and Unidirectional Gated Recurrent Unit (UniGRU).

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \tag{14}$$

The validation is conducted in 100 epochs with similar model architecture. Each model only differs in their direction (Unidirection or Bi-direction) and RNN unit (LSTM or GRU). Adadelta [36] with a 0.25 clipping norm is used as an optimizer. Adadelta is used as it can adapt the learning rate dynamically during training, which usually helps to overcome the problem of diminishing learning rates. Root Mean Squared Error (RMSE) is used as the metric in model evaluation. RMSE measures the average difference between values predicted by a model and the actual values. RMSE formula is described in Eq. (14).

The model validation result is described in Fig. 10. It can be seen that the proposed BiLSTM outperforms other models. BiLSTM generates a 0.022 error on the last epoch. Meanwhile, UniLSTM, BiGRU, and UniGRU produce 0.072, 0.052, and 0.092, respectively. It is noted that Bidirectional models generally perform better. BiLSTM outperforms UniLSTM, and BiGRU outperforms UniGRU. It is also noted that the LSTMs perform better than the GRUs. However, the overall model results (either the proposed or others) have shown that RNN and the adopted NERD dataset are well-fitted to perform this task.

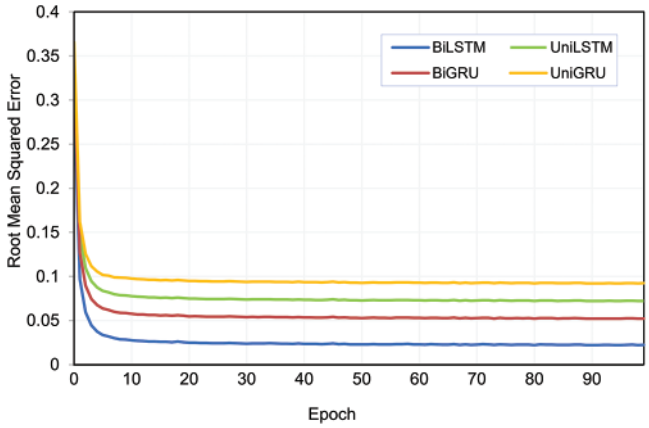

Figure 10: Models validation result

Furthermore, the BiLSTM is evaluated to predict the test set. Tes set consists of 14,000 rows (±10% of the total dataset) which is not included in the training process. The result in Fig. 11a shows that the error line generally swamps the $y - \hat{y}$ matrices. It means that the model usually predicts the data with a small error value. The prediction results are also validated by Fig. 11b, which shows that the error distribution frequently occurs in the region of minor errors.

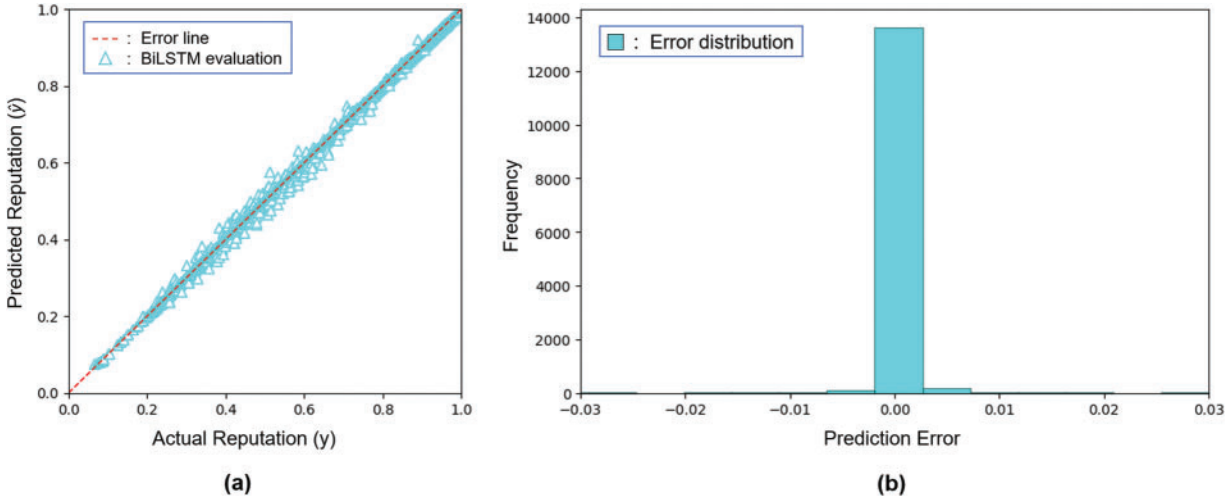

Figure 11: Model evaluation (a) and error frequency distribution (b) using the test set

4.2 Monopoly Attack Evaluation

A monopoly attack refers to an attack on the integrity of a blockchain system in which a single malicious actor manages to control the majority of the network, potentially causing network disruption. For performance-based blockchains such as PoW, the attack might happen if the actor outpowers the performance of other nodes. The attacker would have enough mining power to intentionally modify the ordering of transactions, preventing some or all transactions from being confirmed. Meanwhile, for reputation-based blockchains such as PoS, the attack might happen if the actor manages to capitulate most of the resource that is used as the reputation metric, i.e., stake. An

attacks simulation against PoW, PoS, and PoIR consensus algorithms is conducted to prove that PoIR is more resistant to monopoly attacks. Monopoly attacks on the PoS and PoW networks are carried out by giving enormous stake values and hash power to nodes 1–4, expecting these nodes to win the mining process most of the time. For the PoIR network, the simulation considers nodes 1–4 attempting to modify their identity by altering their IP address while maintaining a high reputation.

Fig. 12 shows the monopoly attack evaluation. The evaluation is conducted in 1000 iteration rounds of block submissions. It can be seen that nodes 1–4 on PoS that conduct the attacks (highlighted in red) manage to control the majority (58%) of transaction processes. In 1000 rounds of PoS block submission, the average malicious node can win the transaction process 145 times, while the average node can only win 26 times. Results in PoW show a slight centralization. For PoW, the node with high-performance capability can also win the majority (32%) of the transaction process. In 1000 rounds of PoW block submission, the average malicious node can win the transaction process 80 times, while the average node can only win 42 times. Meanwhile, the supposed to be malicious nodes on PoIR failed to win the majority of the transaction. The leader is fairly distributed over 1000 iterations, with its standard deviation only 0.73 compared to 48.8 in PoS and 15.5 in PoW. This evaluation confirms that PoIR is more resistant to monopolization than PoS and PoW.

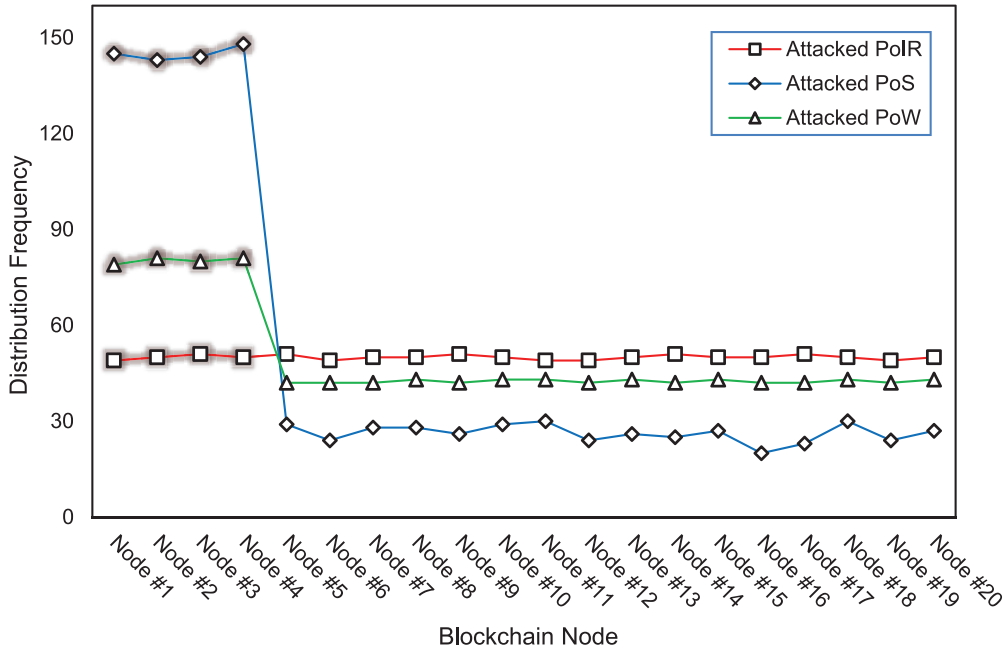

Figure 12: Leader distribution on attacked blockchain consensus

4.3 Transaction Time Evaluation

Transaction time denotes the process a blockchain network needs to complete its transaction. This evaluation aims to verify that PoIR does not require enormous computational demands. The transaction time metric is used to measure computational demand, with the expectation that the longer the transaction time required, the more resources a blockchain network uses to complete its transaction. PoW with various levels of leading zero will be compared to PoS and PoIR.

Fig. 13 shows the transaction time evaluation result. The figure described that PoW as a performance-based consensus algorithm requires higher computational resources when compared to PoS and PoIR. It can be seen that as the difficulty value of the leading zero being sought increases, the time required also increases. PoW, with the highest leading zero, needed 120 s to complete a transaction, while PoS and PoIR only needed 10 and 12 s, respectively. It was observed that the required transaction time for the PoIR consensus algorithm appears to be ±2 s longer compared to the PoS consensus algorithm. The proposed characteristics of the PoIR mechanism can explain this difference. Unlike PoS, which primarily relies on the nodes' stake or ownership of the original digital currency for node selection, PoIR takes into account multiple variables and factors. This outcome is expected as having more variables to consider may result in a longer transaction time. However, considering the results of the monopoly attack evaluation in Section 4.2, the transaction time in PoIR is still deemed acceptable as the benefits outweigh the drawbacks. This evaluation confirms that PoIR transaction time is good enough compared to established performance-based and reputation-based consensus mechanisms.

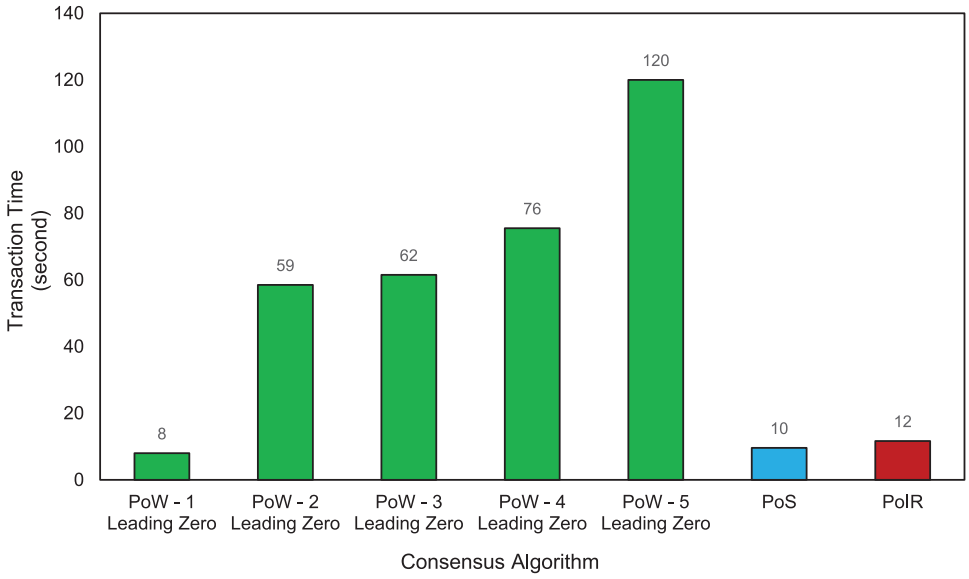

Figure 13: Transaction time evaluation of PoIR, PoA, and PoS

5 Conclusion

A node selection mechanism in reputation-based blockchain consensus has been built. PoIR uses BiLSTM to investigate and select nodes based on multi-criteria variables represented as a reputation. The NERD dataset is used to train the BiLSTM model. The performance of PoIR was examined in comparison to the existing blockchain consensus mechanisms, i.e., PoW and PoS. Additionally, a comparison was made between PoIR and other state-of-the-art RNN models, i.e., BiGRU, UniLSTM, and UniGRU. Evaluation results present that the proposed PoIR enhances the blockchain performance, which can be seen from relatively good transaction time while having more resistance to monopoly attack, which can happen in other consensuses. Furthermore, the overall model results, either the proposed BiLSTM or others, have shown that RNN and the adopted NERD dataset are well-fitted to perform the node selection task. It can be concluded that the proposed work can be

an alternative to PoW by eliminating large computational power consumption while having complete decentralization characteristics that other reputation-based consensuses do not have.

However, while this work provides a foundation for harnessing state-of-the-art deep learning capability for a fully decentralized reputation-based consensus mechanism, the static nature of reputation thresholds in this work presents an opportunity for improvement. Currently, the reputation score thresholds are fixed at the values mentioned earlier, which may not adequately accommodate network setup and conditions variations. This rigidity in threshold values may limit the PoIR mechanism's adaptability in different deployment scenarios.

Future research can focus on developing an adjustable dynamic reputation threshold mechanism based on certain characteristics and dynamics. By incorporating adaptive algorithms such as deep reinforcement learning, the reputation thresholds can be dynamically set based on real-time network conditions, node behavior, and other relevant factors. This self-adjusting reputation thresholds concept might enable a more fine-grained and context-aware classification of nodes. Another aspect to consider is the incorporation of reputation feedback mechanisms. The idea would involve designing protocols or mechanisms through which nodes can provide feedback on the reputation scores of other nodes, enabling a collective assessment of trustworthiness. By leveraging the wisdom of the crowd, the reputation thresholds can be refined and updated based on community consensus, resulting in a more democratic approach.

Acknowledgement: We would like to thank the Ministry of Education, Culture, Research, and Technology (Kemendikbudristek) of Indonesia, who provided financial support for this research. We would also like to express our gratitude to the members of the IoT+ research group, who provided valuable feedback at every stage of the research.

Funding Statement: This research was funded by the Ministry of Education, Culture, Research, and Technology (Kemendikbudristek) of Indonesia under PDD Grant with Grant Number NKB-1016/UN2.RST/HKP.05.00/2022.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: J. H. Windiatmaja, D. Hanggoro, R. F. Sari; data collection: J. H. Windiatmaja; analysis and interpretation of results: J. H. Windiatmaja; draft manuscript preparation: J. H. Windiatmaja, D. Hanggoro, M. Salman, R. F. Sari. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The original Network Entity Reputation Database is available and can be accessed at https://nerd.cesnet.cz/nerd/ips/. The PoIR code and pre-processed dataset is also available at https://gitlab.com/jauzak/poir.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] K. Azbeg, O. Ouchetto, S. Jai Andaloussi and L. Fetjah, "An overview of blockchain consensus algorithms: Comparison, challenges and future directions," in *Advances on Smart and Soft Computing*, Casablanca, Morocco, pp. 357–369, 2021.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, vol. 1, pp. 21260, 2008.

[3] F. Restuccia, S. D. Salil, S. Kanhere, T. Melodia and S. K. Das, "Blockchain for the Internet of Things: Present and future," arXiv preprints arXiv:1903.07448, 2019.
[4] M. Belotti, N. Božić, G. Pujolle and S. Secci, "A vademecum on blockchain technologies: When, which, and how," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3796–3838, 2019.
[5] D. Vujičić, D. Jagodić and S. Ranđić, "Blockchain technology, bitcoin, and Ethereum: A brief overview," in *17th Int. Symp. Infoteh-Jahorina (Infoteh)*, East Sarajevo, Republic of Srpska, pp. 1–6, 2018.
[6] Y. Liu, Y. Lan, B. Li, C. Miao and Z. Tian, "Proof of Learning (PoLe): Empowering neural network training with consensus building on blockchains," *Computer Networks*, vol. 201, pp. 108594, 2021.
[7] S. Leonardos, D. Reijsbergen and G. Piliouras, "Weighted voting on the blockchain: Improving consensus in proof of stake protocols," in *Proc. of the 2019 IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC)*, Seoul, Korea, 2019.
[8] Y. Luo, Y. Chen, Q. Chen and Q. Liang, "A new election algorithm for DPos consensus mechanism in blockchain," in *Proc. of the 2018 7th Int. Conf. on Digital Home (ICDH)*, Guilin, China, pp. 116–120, 2018.
[9] S. Joshi, "Feasibility of proof of authority as a consensus protocol model," arXiv preprint arXiv:2109.02480, 2021.
[10] S. Dziembowski, S. Faust, V. Kolmogorov and K. Pietrzak, "Proofs of space," in *Annual Cryptology Conf.*, Santa Barbara, CA, USA, pp. 585–605, 2015.
[11] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu *et al.,* "On security analysis of proof-of-elapsed-time (PoET)," in *Stabilization, Safety, and Security of Distributed Systems: 19th Int. Symp. (SSS 2017)*, Boston, MA, USA, pp. 282–297, 2017.
[12] K. Karantias, A. Kiayias and D. Zindros, "Proof-of-burn," in *Financial Cryptography and Data Security*, Kota Kinabalu, Malaysia, pp. 523–540, 2020.
[13] Y. Gilad, R. Hemo, S. Micali, G. Vlachos and N. Zeldovich, "Algorand: Scaling Byzantine agreements for cryptocurrencies," in *Proc. of the 26th Symp. on Operating Systems Principles*, Shanghai, China, pp. 51–68, 2017.
[14] S. Zhang and J. H. Lee, "Analysis of the main consensus protocols of blockchain," *ICT Express*, vol. 6, no. 2, pp. 93–97, 2020.
[15] V. Bartos, M. Zadnik, S. M. Habib and E. Vasilomanolakis, "Network entity characterization and attack predictio," *Future Generation Computer Systems*, vol. 97, pp. 674–686, 2019.
[16] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," *Journal of Cryptology*, vol. 3, no. 2, pp. 99–111, 1991.
[17] N. Shibata, "Proof-of-search: Combining blockchain consensus formation with solving optimization problems," *IEEE Access*, vol. 7, pp. 172994–173006, 2019.
[18] F. Bravo-Marquez, S. Reeves and M. Ugarte, "Proof-of-learning: A blockchain consensus mechanism based on machine learning competitions," in *2019 IEEE Int. Conf. on Decentralized Applications and Infrastructures (DAPPCON)*, Newark, CA, USA, pp. 119–124, 2019.
[19] C. Chenli, B. Li, Y. Shi and T. Jung, "Energy-recycling blockchain with proof-of-deep-learning," in *2019 IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC)*, Seoul, South Korea, pp. 19–23, 2019.
[20] X. Luo, P. Yang, W. Wang, Y. Gao and M. Yuan, "S-PoDL: A two-stage computational-efficient consensus mechanism for blockchain-enabled multi-access edge computing," *Physical Communication*, vol. 46, no. 1, pp. 101338, 2021.
[21] B. Li, C. Chenli, X. Xu, Y. Shi and T. Jung, "DLBC: A deep learning-based consensus in blockchains for deep learning services," arXiv preprint arXiv:1904.07349, 2019.
[22] X. Qiu, Z. Qin, W. Wan, J. Zhang, J. Guo *et al.,* "A dynamic reputation-based consensus mechanism for blockchain," *Computers, Materials & Continua*, vol. 73, no. 2, pp. 2577–2589, 2022.
[23] J. Shi, X. Zeng and Y. Li, "Reputation-based sharding consensus model in information-centric networking," *Electronics*, vol. 11, no. 5, pp. 830, 2022.
[24] J. Yun, Y. Goh and J. M. Chung, "Trust-based shard distribution scheme for fault-tolerant shard blockchain networks," *IEEE Access*, vol. 7, pp. 135164–135175, 2019.

[25] Y. Goh, J. Yun, D. Jung and J. M. Chung, "Secure trust-based delegated consensus for blockchain frameworks using deep reinforcement learning," *IEEE Access*, vol. 10, pp. 118498–118511, 2019.
[26] K. Saadat, N. Wang and R. Tafazolli, "AI-enabled blockchain consensus node selection in cluster-based vehicular networks," *IEEE Networking Letters*, vol. 5, no. 2, pp. 1–5, 2023.
[27] S. Liu, R. Zhang, C. Liu and D. Shi, "P-PBFT: An improved blockchain algorithm to support large-scale pharmaceutical traceability," *Computers in Biology and Medicine*, vol. 154, no. 3, pp. 106590, 2023.
[28] H. F. Ouattara, D. Ahmat, F. T. Ouédraogo, T. F. Bissyandé and O. Sié, "Blockchain consensus protocols," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Cham, Switzerland: Springer International Publishing, pp. 304–314, 2018.
[29] S. Kingslin and R. Zahra, "An effective randomization framework to PoW consensus algorithm of blockchain (RPoW)," *International Journal of Engineering and Advanced Technology*, vol. 8, pp. 1793–1797, 2019.
[30] N. Chaudhry and M. M. Yousaf, "Consensus algorithms in blockchain: Comparative analysis, challenges and opportunities," in *Proc. of the 2018 12th Int. Conf. on Open Source Systems and Technologies (ICOSST)*, Lahore, Pakistan, pp. 54–63, 2018.
[31] S. Sharkey and H. Tewari, "Alt-PoW: An alternative proof-of-work mechanism," in *Proc. of the 2019 IEEE Int. Conf. on Decentralized Applications and Infrastructures (DAPPCON)*, Newark, CA, USA, pp. 11–18, 2019.
[32] L. von Ahn, B. Maurer, C. McMillen, D. Abraham and M. Blum, "reCAPTCHA: Human-based character recognition via web security measures," *Science*, vol. 321, no. 5895, pp. 1465–1468, 2008.
[33] A. K. Yadav and K. Singh, "Comparative analysis of consensus algorithms of blockchain technology," in *Ambient Communications and Computer Systems*, Singapore, Springer, pp. 15–24, 2020.
[34] Y. Xiao, N. Zhang, J. Li, W. Lou and Y. T. Hou, "Distributed consensus protocols and algorithms," *Blockchain for Distributed Systems Security*, vol. 25, pp. 40, 2019.
[35] E. Bisong, "Google colaboratory," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Berkeley, CA: Apress, pp. 59–64, 2019.
[36] D. M. Zeiler, "Adadelta: An adaptive learning rate method," arXiv preprint arXiv:1212.5701, 2012.