Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 9, No. 4, August, 2025

Optimizing Consensus in Blockchain with Deep and Reinforcement Learning

William Villegas-Ch [1*], Jaime Govea [1], Rommel Gutierrez [1]

[1] *Escuela de Ingenierí-a en Ciberseguridad, FICA, Universidad de Las Américas, Quito 170125, Ecuador.*

Abstract

This study aims to optimize blockchain consensus mechanisms by integrating artificial intelligence techniques to address critical limitations in latency, scalability, computational efficiency, and security inherent in traditional protocols, such as PoW, PoS, and PBFT. The proposed model combines deep neural networks (DNNs) for feature extraction with deep reinforcement learning (DRL), specifically Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO), to enable dynamic validator selection and real-time adjustment of consensus difficulty. The training process utilizes a hybrid dataset of historical blockchain records from Ethereum and Hyperledger networks and synthetic data from simulated attack scenarios involving Sybil, 51%, and DoS threats. Experimental evaluations were conducted in private and permitted environments under varying transactional loads. Results show a 60% reduction in confirmation latency compared to PoW, achieving 320 ms, and a 20% improvement over PBFT. Transaction throughput increased to 22,000 transactions per second (TPS), and computational resource consumption was reduced by 30%. The model achieved an attack tolerance of up to 92%, significantly enhancing network resilience. The novelty of this work lies in its autonomous consensus optimization strategy, which enables adaptive and secure protocol behaviour without manual intervention, representing a scalable and efficient solution for future blockchain infrastructures.

Keywords:

Blockchain;

Consensus Optimization;

Artificial Intelligence;

Deep Reinforcement Learning.

Article History:

Received:	30	March	2025
Revised:	09	June	2025
Accepted:	16	July	2025
Published:	01	August	2025

1- Introduction

Blockchain technology has emerged as a foundational infrastructure for decentralized and secure data management, enabling transparent, immutable, and trustless transactions across distributed systems [1]. This paradigm has transformed sectors such as finance, healthcare, and critical infrastructure, providing tamper-resistant environments for digital interaction [2]. At the heart of any blockchain system lies the consensus mechanism, which is responsible for validating transactions and ensuring integrity in the absence of a central authority.

Traditional consensus protocols—such as Proof of Work (PoW), Proof of Stake (PoS), and Practical Byzantine Fault Tolerance (PBFT) have facilitated the success of early blockchain applications but now face significant scalability and efficiency bottlenecks. Proof-of-Work (PoW), employed in Bitcoin and similar platforms, offers strong security guarantees through computational difficulty but exhibits low transaction throughput and excessive energy consumption, with confirmation times frequently exceeding 1,200 ms [3]. These characteristics severely limit their applicability in high-frequency transaction environments. Proof of Stake (PoS) and its variants, including Delegated Proof of Stake (DPoS), address some of these limitations by replacing energy-intensive mining with stake-based validation. However, these approaches introduce new risks, such as validator centralization, Sybil attacks, and the Nothing-at-Stake problem

* **CONTACT**: william.villegas@udla.edu.ec

DOI: http://dx.doi.org/10.28991/ESJ-2025-09-04-08

[4, 5]. PBFT, commonly used in permissioned blockchains, offers lower latency, approximately 400 ms, and resilience to a subset of adversarial conditions. Still, its scalability is limited due to the quadratic message complexity between validator nodes [6]. To address these challenges, the research community has explored enhancements to consensus mechanisms through optimization and the application of artificial intelligence. Heuristic approaches, such as genetic algorithms, particle swarm optimization, and snake optimization, have been proposed to reduce block confirmation times and improve validator scheduling under constrained conditions [7, 8]. Although promising, these strategies typically rely on static assumptions and rule-based adaptations, which limit their responsiveness to dynamic network behavior or evolving threats.

Machine learning, more recently deep learning, has opened new avenues for performance-aware consensus management. Deep neural networks (DNNs) have been utilized to predict congestion trends in blockchain networks, enabling more informed transaction propagation and block generation strategies [9, 10]. However, DNNs alone do not interact directly with the consensus decision process, limiting their ability to reconfigure validator behavior or adjust protocol difficulty adaptively. Furthermore, their reliance on labeled data and static prediction limits robustness in adversarial or unstructured environments.

More advanced strategies have turned to deep reinforcement learning (DRL), which enables agents to learn optimal decision policies in complex environments via feedback-based interactions. Models such as Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO) have been applied in blockchain scenarios to optimize block propagation, detect anomalies, or automate validator selection under uncertainty [11, 12].

The paper is organized as follows. Section 2 presents the literature review, where previous approaches in blockchain consensus optimization and their limitations are discussed. Section 3 describes the methodology, including the architecture of the AI model, the data used, and the evaluation protocols. Section 4 presents the results obtained, comparing the performance of the proposed model with other consensus mechanisms. Section 5 presents a discussion of the findings, analyzing their impact on the efficiency and security of the network. Finally, Section 6 concludes the paper and proposes directions for future research. Nevertheless, most DRL-based approaches have been constrained to simulation environments and rarely integrated into real or emulated blockchain testnets for experimental validation. These models often focus on isolated tasks, such as DoS mitigation or peer reputation modeling, without delivering full-stack consensus optimization.

The proposed model is trained on a hybrid dataset comprising historical blockchain data (Ethereum, Hyperledger Fabric) and synthetic logs derived from adversarial scenarios to ensure robust evaluation. It is deployed and tested in controlled environments using tools such as Hyperledger Caliper and Ganache, allowing for reproducible experimentation under variable load and attack conditions [13, 14]. Results demonstrate significant improvements: confirmation latency is reduced to 320 ms (a 60% improvement over PoW), maximum throughput reaches 22,000 TPS, and attack tolerance rises to 92%, surpassing traditional and hybrid consensus approaches [15].

Our contribution is bridging the gap between passive analytics and active decision-making in blockchain consensus. While previous models have either predicted system states or proposed static optimizations, our solution introduces an intelligent and autonomous control loop that actively governs real-time validator behavior and protocol configuration. This approach directly impacts high-demand blockchain applications, such as decentralized finance (DeFi), supply chain traceability, and mission-critical Internet of Things (IoT) networks, where performance, resilience, and efficiency must be optimized simultaneously.

This study addresses the current limitations by proposing a hybrid artificial intelligence model that combines DNNs for network state representation with deep reinforcement learning (DRL) techniques for adaptive consensus optimization. The model is designed to operate within live blockchain environments and offers three primary innovations. First, it enables dynamic validator selection based on reliability, energy efficiency, and historical behavior metrics, thereby overcoming the static assignment problem in PoW and PoS protocols. Second, it adjusts consensus difficulty in real-time according to transactional load and network topology conditions, improving responsiveness to congestion and minimizing latency. Third, it incorporates a security-aware feedback mechanism that identifies and responds to Sybil, 51%, and DoS attack patterns using anomaly detection techniques integrated into the reinforcement learning loop.

2- Literature Review

The optimization of blockchain consensus mechanisms has garnered substantial research attention in recent years, particularly due to increasing demands for scalability, energy efficiency, and resilience to attacks. As the introduction highlights, traditional protocols, such as PoW, PoS, and PBFT, face well-documented throughput, latency, and decentralization constraints. Consequently, various lines of research have attempted to overcome these limitations through algorithmic enhancements and intelligent systems integration. This section categorizes and critically analyzes the most representative approaches into four main groups: heuristic optimization models, deep neural network-based predictors, deep reinforcement learning strategies, and hybrid consensus mechanisms.

2-1- Heuristic-Based Optimization Models

Early attempts to enhance consensus efficiency relied on nature-inspired metaheuristics to reduce validation time and optimize resource allocation without fundamentally altering the protocol, for example, Taher et al. [12] proposed the Snake Optimization Algorithm to improve blockchain scalability by dynamically adjusting validator roles. Similarly, Nourmohammadi & Zhang [13] employed Particle Swarm Optimization to mitigate blockchain forks in governance processes.

While these methods demonstrate improvements in specific metrics, such as block confirmation time or validator throughput, they generally depend on pre-established rules and static system assumptions. This limits their effectiveness in dynamic environments, where transaction load and adversarial behavior evolve. Moreover, heuristic solutions often lack real-time adaptability, which is essential in mission-critical applications like IoT or DeFi networks.

2-2- Predictive Models Using Deep Neural Networks

With the rise of deep learning, several researchers have explored DNNs to forecast system-level behavior. Models such as those proposed by Nourmohammadi & Zhang [13] and Paidipati et al. [14] utilized DNNs to predict transaction congestion and model network anomalies. These approaches provide insight into latent patterns within blockchain metrics and can guide proactive network tuning.

However, despite their predictive strength, DNNs operate in a feed-forward, supervised learning regime, which inherently limits their capacity to adapt in real time. They are not decision-making agents and cannot autonomously reconfigure consensus parameters or select validator subsets based on context. Their reliance on historical, labeled datasets may also reduce performance in unseen scenarios or adversarial settings.

2-3- Deep Reinforcement Learning for Consensus Adaptation

To address the limitations of static prediction, recent studies have adopted reinforcement learning techniques, particularly intensive DRL, to enable adaptive behaviour in consensus systems. Emil Selvan et al. [7] integrated a DQN for intrusion mitigation in blockchain nodes. At the same time, Aitchison & Sweetser [12] introduced a dual-network PPO framework for policy optimization in dynamic environments.

These models demonstrated that RL agents can learn optimal actions for validator assignment or attack response through reward-based feedback loops. Nevertheless, most implementations remain confined to isolated tasks (e.g., attack mitigation) or constrained simulations, lacking full integration with a functioning blockchain consensus pipeline. Their evaluation typically omits performance under hybrid data conditions (historical + adversarial), which limits insights into their operational viability.

2-4- Hybrid Consensus and Integrated Architectures

Another direction involves hybrid consensus mechanisms that combine the strengths of PoW, PoS, and PBFT variants. Gupta et al. [15] proposed a PoW–PoS protocol to mitigate the impact of 51% attacks. While this configuration showed improved resistance to malicious majority takeovers, it did not address latency or validator centralization issues. Other works, such as those in [4], integrated blockchain with federated learning or reputation-based scoring systems to secure validator behaviour across domains.

Although hybrid mechanisms introduce structural diversity and resilience, they rely on static validation logic and fixed threshold parameters. They lack embedded intelligence capable of perceiving environmental changes or continuously optimizing protocol operation.

2-5- Identified Gaps and Proposed Direction

The reviewed literature reveals a clear gap in delivering autonomous, adaptive, and experimentally validated consensus optimization mechanisms that can operate under variable load conditions and real-world threats. Most existing approaches either predict system behaviour without acting upon it (DNNs), respond within isolated scopes (DRL), or apply rule-based optimizations (heuristics) that do not generalize.

This study proposes an integrated model that bridges the gap between prediction and adaptation. The proposed solution introduces a closed-loop optimization strategy by combining deep neural networks for dynamic feature extraction with deep reinforcement learning for validator selection and consensus difficulty adjustment. Unlike prior DRL applications, our model is trained on hybrid datasets incorporating real blockchain metrics and simulated attacks and is validated in controlled experimental environments using real blockchain platforms, such as Ethereum testnets and Hyperledger Fabric.

3- Materials and Methods

3-1- Data Management and Preprocessing

Data management in this study encompasses the collection, storage, and transformation of information used for AI-based consensus protocol optimization. The process begins with acquiring data from established blockchain networks, transaction simulations, and security event logs. Data integrity and quality are critical to the efficiency of the AI model, so preprocessing and filtering strategies are implemented before incorporating them into the training process.

Data collection is done from multiple sources to obtain representative information on the operating dynamics of blockchain networks. Historical data extracted from decentralized networks such as Ethereum, Hyperledger Fabric [16], and Binance Smart Chain [17] These include block logs, transaction validations, node energy consumption, and consensus performance metrics. Extraction is done through queries through blockchain-specific API interfaces, such as Etherscan API for Ethereum and Hyperledger Explorer for permissioned networks. This data is stored in a structured database that allows for further processing.

In addition to historical data, controlled simulation scenarios are generated to evaluate the model's adaptability under different operating conditions. To do so, test environments are set up with varying levels of network congestion, variations in the number of validator nodes, and fluctuations in the transaction rate. Transaction simulation uses Hyperledger Caliper to measure network performance under different consensus configurations and Ganache to create local environments on Ethereum [18]. The simulation data feeds the AI model and optimizes its ability to adjust dynamically to network conditions.

The dataset also includes security event logs associated with attack attempts within the blockchain network. For this purpose, specialized cybersecurity databases documenting previous attacks are integrated, such as CIC-IDS 2017, which contains network traffic data with labels identifying attacks such as DoS and spoofing (Sybil) [19]. Additionally, specific scenarios are designed in which malicious transactions are introduced into the network to analyze the response of the AI-optimized consensus protocol. Analyzing these events allows the detection of anomalous patterns and strengthens the security of the block validation system.

The preprocessing of the acquired data guarantees the effectiveness of the AI model in optimizing consensus. A cleaning process eliminates duplicate records, irrelevant transactions, and inconsistent data. Subsequently, normalization is applied to key variables such as transaction confirmation time, the number of active validators, and energy consumption, ensuring that the values are within a range suitable for training the model. Data transformation also includes generating features using time series analysis techniques to capture trends and variations in network activity.

The processed data is stored in a distributed infrastructure that facilitates real-time access and analysis. InterPlanetary File System (IPFS) is used for decentralized management of historical records, and PostgreSQL is used for structured manipulation of information relevant to the AI model [20]. This combination allows for efficient integration with the consensus optimization system, ensuring that data processing is scalable and secure.

Feature selection is performed using dimensionality reduction techniques, such as Principal Component Analysis (PCA) and neural network-based Autoencoders, to identify the most relevant variables that impact consensus efficiency. Table 1 presents the main variables used in consensus optimization, detailing their description and impact on blockchain network performance. The correlation between attributes such as the number of validators and confirmation latency is analyzed to improve the model's predictive capacity, allowing dynamic adjustments based on network conditions and ensuring that the optimized protocol responds effectively to changes in blockchain activity.

Table 1. Key Variables in Consensus Optimization

Variable	Description	Impact on Consensus Optimization
Number of validators	Number of nodes participating in the validation process.	Affects decentralization and efficiency in decision-making.
Confirmation latency	Time required for a transaction to be validated and added to the block.	Determines the speed and responsiveness of consensus.
Transaction rate	Number of transactions processed per second.	Related to scalability and network load.
Energy consumption	Energy is used in block validation and protocol execution.	A key factor in sustainability and resource optimization.
Block size	The amount of data stored in each block of the blockchain.	Influences performance and block propagation in the network.
Attack frequency	Number of attack attempts detected in a time interval.	Allows for evaluating the security and adaptability of consensus.
Detection success rate	Percentage of attacks detected and mitigated by the system.	Indicator of the robustness of the integrated security model.
Load variability	Fluctuations in network activity over time.	Allows for dynamic adjustment of consensus parameters.

The combined dataset used in this study includes approximately 1.5 million blockchain transactions collected between January 2021 and December 2023, extracted at regular hourly intervals using APIs such as Etherscan and Hyperledger Explorer. This historical dataset was partitioned into 70% for training, 15% for validation, and 15% for testing. In addition, more than 60,000 synthetic transactions were generated across 50 simulation scenarios using Hyperledger Caliper and Ganache, with validator nodes ranging from 10 to 1,000 and throughput levels varying from 100 to 50,000 TPS. Regarding security events, 4,500 labeled attack instances from the CIC-IDS 2017 dataset were adapted to the blockchain context by mapping network anomalies to validator and transaction behaviors, such as node flooding for DoS and identity replication for Sybil attacks. These integrated data sources ensure a balanced and comprehensive training foundation, facilitating performance optimization and anomaly detection within the consensus model.

3-1-1- Data Preprocessing and Cleaning

Preprocessing the acquired data ensures the quality of the information used in AI-based consensus protocol optimization. This process encompasses eliminating redundant data, normalizing variables, and detecting anomalies in the network.

The first stage consists of the elimination of redundant data and the filtering of fraudulent transactions. Since blockchain records contain multiple copies of the same information due to the system's distribution, hashing-based deduplication algorithms are employed, ensuring that each stored transaction is unique [21]. In addition, digital signature analysis is applied to identify repeated or inconsistent operations. A supervised classification model is used to detect fraudulent transactions, trained with historical data sets containing malicious transaction labels, such as those associated with double-spending attacks or block collision attempts.

Once the data has been filtered, the most relevant variables are normalized and transformed for integration into the AI model. Confirmation latency, transaction rate, and energy consumption show significant variations, so min-max scaling is applied to bring the values to a range between 0 and 1. Logarithmic transformations or Z-score normalization stabilize the variance and improve the model's learning capacity in cases where the data distribution presents asymmetry.

To improve data quality, anomaly detection techniques based on Isolation Forest and One-Class SVM are implemented. These techniques allow the identification of records with atypical patterns in network activity. These techniques are complemented by a time series analysis using autoregressive models (ARIMA), which will enable the detection of anomalous variations in transaction load or validator activity. Data with significant deviations are flagged for review or elimination before being incorporated into the consensus optimization model.

3-1-2- Structure and Storage

The acquired data is stored in a distributed infrastructure that guarantees integrity, availability, and security in managing transaction records and security events. Figure 1 represents the process from data acquisition to its integration with the AI model, detailing each key stage. The storage strategy combines distributed and structured database technologies, optimizing the efficient handling of information in blockchain environments.

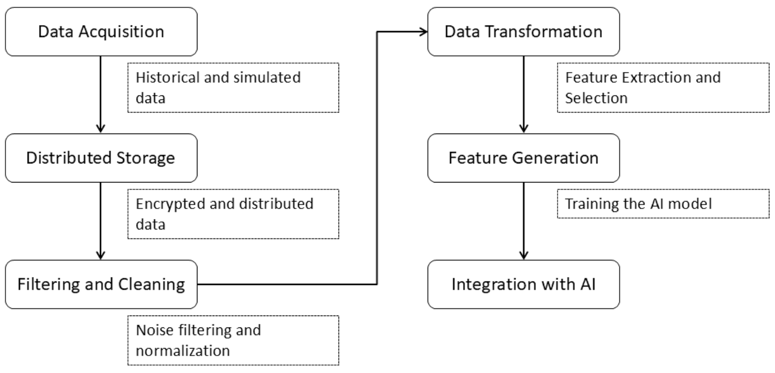

Figure 1. Data Management and Preprocessing Flow

IPFS allows transaction records to be stored in a decentralized manner, ensuring their availability on the network without relying on centralized servers [22]. In the case of highly dynamic records, such as the evolution of the network load, BigchainDB is used, which allows rapid information consultation in a distributed environment. Implementing homomorphic encryption schemes reinforce the security of the stored data. These schemes enable operations on encrypted data without decrypting it, reducing the risk of exposing sensitive information. In addition, an anonymization mechanism based on masking techniques and SHA-3 cryptographic hashing is incorporated, ensuring the privacy of network participants.

Integration with real-time analysis tools allows dynamic optimization of consensus. A distributed processing pipeline is implemented using Apache Kafka, which receives events from the network in real time, analyzes them, and updates the AI models based on system state changes. This allows the consensus protocol to adaptively respond to variations in network activity by adjusting key parameters such as the validation threshold or block confirmation difficulty.

3-1-3- Feature Generation for AI

Feature generation is a process used to improve the accuracy of the AI model in optimizing consensus. It involves a rigorous selection of key attributes that directly impact the performance and security of the blockchain network. Key features include transaction rate, confirmation latency, energy consumption, and the number of active validators. These variables are extracted directly from blockchain records and complemented with derived data, such as variability in network load and frequency of attack attempts. To ensure that the selected features are representative and not redundant, correlation analysis using a Pearson correlation matrix and independence tests using a Chi-square on categorical variables are applied.

The analysis of relationships between variables is carried out using machine learning techniques, such as Random Forest Feature Importance, which allows a weight to be assigned to each variable based on its relevance in predicting the state of the network. In addition, deep learning-based feature selection models are integrated, where an autoencoder trained with historical records allows for reducing the dimensionality of the data set, retaining only the most relevant information.

To optimize data processing and improve model efficiency, dimensionality reduction techniques such as PCA and embedding techniques with recurrent neural networks (RNNs) are applied. PCA allows for reducing the number of features without losing critical information. At the same time, the embedding generated by RNNs captures the temporal evolution of blockchain network activity, improving the AI model's ability to predict changes in the network and dynamically adjust consensus.

3-2- Artificial Intelligence Model for Consensus Optimization

The AI model designed to optimize the blockchain consensus protocol combines deep neural networks and reinforcement learning to improve transaction validation efficiency and network security. Integrating heuristic optimization techniques allows for dynamic adjustment of consensus parameters based on network load and validator activity, ensuring a balance between decentralization, processing speed, and attack resistance.

3-2-1- AI Model Architecture

The model architecture comprises a DNN for feature extraction and a DRL agent based on the Proximal Policy Optimization (PPO) algorithm for optimizing consensus strategies. The DNN layer captures patterns in high-dimensional metrics, including confirmation latency, transaction rate, block density, validator energy consumption, and anomaly flags. These features are extracted from the system state vector for the DRL agent [23].

The DRL agent uses PPO due to its stability in environments with continuous state spaces and bounded discrete actions. PPO ensures policy improvement through clipped surrogate objectives, which maintain a balance between exploration and convergence efficiency. The action space includes: (1) selecting optimal validators based on historical reliability and recent behavior, and (2) dynamic adjustment of consensus parameters such as quorum size, propagation timeout, and block difficulty.

The state input to the agent consists of a n-dimensional vector St including normalized metrics for:

- Validator participation history
- Confirmation time variance
- Network congestion indicators
- Node trust scores
- Anomaly detection signals

The action output At includes discrete actions:

- Add/remove validator i
- Increase/decrease consensus threshold
- Reassign the block leader role

The reward function is defined as follows

$$R_t = -\alpha \cdot L_t - \beta \cdot C_t + \gamma \cdot D_t - \delta \cdot F_t \tag{1}$$

where:

- Lt: confirmation latency at time t
- Ct: CPU usage by validators
- Dt: decentralization coefficient (calculated via entropy of validator distribution)
- Ft: validation failure rate
- α, β, γ, δ: tunable weight coefficients

The decentralization coefficient *Dt* is derived from the Shannon entropy of validator participation over recent consensus rounds:

$$D_t = -\sum_{i=1}^{n} p_i \log p_i \tag{2}$$

where pi is the proportion of validation contributions made by node i, this term ensures that the agent is penalized when validator selection becomes concentrated in a few nodes, thus enforcing fairness and diversity in validator participation.

The model's training uses stochastic gradient descent with an Adam optimizer. The policy and value networks are updated using mini-batches and clipped objectives, ensuring bounded policy changes. Cross-validation uses K-folds on synthetic and real-world data, ensuring generalization across network loads and attack scenarios. Figure 2 depicts the overall architecture and data flow, highlighting the interaction between input processing, DNN-based feature extraction, PPO-based decision-making, and action deployment in the blockchain network.

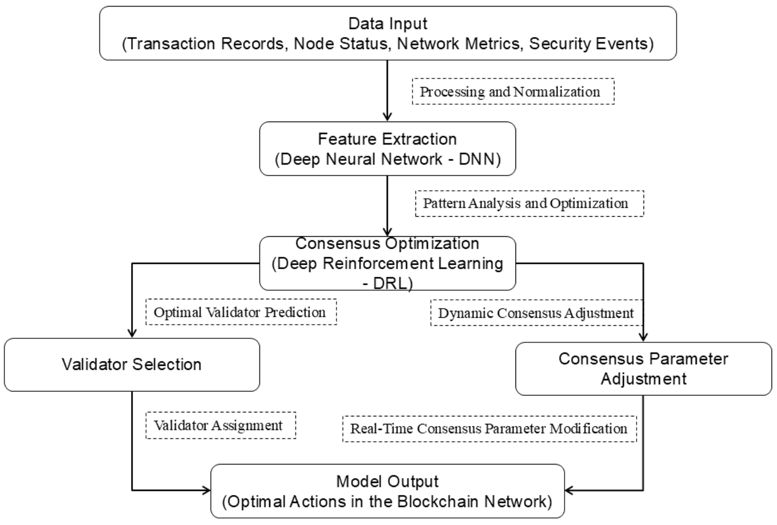
Data Input
(Transaction Records, Node Status, Network Metrics, Security Events)
Processing and Normalization
Feature Extraction
(Deep Neural Network - DNN)
Pattern Analysis and Optimization
Consensus Optimization
(Deep Reinforcement Learning
- DRL)
Optimal Validator Prediction
Dynamic Consensus Adjustment
Validator Selection
Consensus Parameter
Adjustment
Validator Assignment
Real-Time Consensus Parameter Modification
Model Output
(Optimal Actions in the Blockchain Network)

Figure 2. AI Model Architecture

The model's learning is based on minimizing a cost function, which is defined as a combination of confirmation latency, energy consumption, and validation failure rate. Stochastic gradient descent (SGD) is used with Adam and RMSProp optimizers to optimize this process, ensuring an efficient update of the network weights. The optimization metric evaluates the impact of each adjustment on the network's stability, penalizing decisions that compromise decentralization or security and maintaining a balance between performance and resilience against attacks.

3-2-2- Model Training and Tuning Techniques

The model is trained using a hybrid dataset of historical records from real blockchain networks and synthetic data generated through controlled simulations. For accurate data acquisition, transaction records and performance metrics are extracted from Ethereum (using Etherscan API), Hyperledger Fabric (using Hyperledger Explorer), and Binance Smart Chain (via BscScan API). This data includes information on block confirmation latency, transaction rate per second, history of successful and failed validations, and energy consumption of validator nodes.

In addition to complementing the model training, synthetic data is generated by implementing a simulation environment based on Hyperledger Caliper and Ganache. In this environment, variable load scenarios are set up, in which the number of transactions per second, the number of validator nodes, and the difficulty of consensus are dynamically modified. This data provides the model with sufficient information to learn to dynamically optimize consensus, adapting to actual conditions and adverse events that may compromise the stability of the blockchain network.

To improve model generalization, hyperparameter tuning strategies are implemented, exploring combinations of neural network depths, learning rates, and activation functions using grid search and Bayesian optimization. Regularization is performed using dropout and batch normalization, avoiding overfitting specific patterns in the training data [24]. Model convergence is assessed through K-fold cross-validation, ensuring the learned strategies are consistent across different datasets. Metrics such as the success rate in predicting efficient validators, reduction in confirmation latency, and stability of energy consumption are monitored, ensuring that the optimized model outperforms traditional consensus protocols.

The training process was conducted with neural network depths ranging from 3 to 8 layers, learning rates varying from 0.001 to 0.00001, and activation functions including ReLU, tanh, and Leaky ReLU. Grid search was initially used to narrow viable configurations, followed by Bayesian optimization to identify the most effective combination of convergence speed and generalization. The Adam optimizer trained the selected model for 100 epochs with a batch size 128. The dataset was split into 70% for training, 15% for validation, and 15% for testing. K-fold cross-validation with K = 5 was employed to evaluate model stability across data partitions. Each fold included a balanced mix of regular and adversarial transactions. The main evaluation metrics included:

- Average confirmation latency (ms) = mean time between transaction emission and validation.
- Transaction throughput (TPS) = total transactions confirmed / time interval.
- Attack detection rate (%) = number of correctly detected attacks / total attacks introduced.
- Energy efficiency (%) = reduction in average consumption compared to PoW baseline.

These metrics were calculated after each training iteration and used as input to the policy update in the DRL module, ensuring real-time feedback integration into the consensus adaptation logic.

3-2-3- Integration with the Consensus Protocol

The AI model is integrated directly into the blockchain consensus mechanism, allowing for the adaptive selection of validators based on network load and each node's historical reliability. A validator prioritization algorithm is defined as one in which nodes with better stability in block validation and lower energy consumption are more likely to be selected, avoiding biases towards validators with high computational capacity but low commitment to decentralization.

The consensus difficulty level is dynamically adjusted using an adaptation function based on reinforcement learning. This function analyzes the block propagation speed and automatically adjusts the validation parameters. This approach reduces latency without compromising security, optimizing network performance in real-time.

In addition, the model includes an attack detection and mitigation module, trained to identify anomalies in node activity and suspicious transactions. Outlier detection techniques based on Isolation Forest and Generative Adversarial Networks (GANs) are employed to recognize fraud attempts, Sybil attacks, and collusion patterns in block validation [25]. When an anomaly is detected, the model generates alerts and adjusts consensus rules to prevent the impact of a possible attack, strengthening the system's security.

3-3- Experimental Environment and Blockchain Network Configuration

The experimental environment is designed to evaluate the impact of consensus optimization using AI on blockchain networks under different configurations. A distributed infrastructure is implemented that allows the execution of multiple tests with variations in transactional load, the number of validators, and consensus parameters.

3-3-1- Network Infrastructure

The network infrastructure used in the experiments is based on three types of blockchain environments, each selected based on its characteristics and the kind of implementation required. First, Ethereum is used in its public and test network versions, specifically Ethereum Testnet (Goerli and Sepolia). It provides a decentralized environment with smart contracts based on the Ethereum Virtual Machine (EVM). Second, Hyperledger Fabric is implemented as a permissioned solution with control over the network nodes, using a Kubernetes-based environment with multiple peers and a Raft-based computer for consensus coordination. Finally, a private blockchain is set up using Hyperledger Besu, where nodes are distributed in a local network, allowing controlled experiments with full tuning of the consensus parameters.

The network's total number of nodes and validators varies depending on the topology configured for each experiment. The relationship between the number of active validators and the processing capacity of the blockchain network is mathematically modeled from the TPS rate. The equation gives the network capacity.

$$TPS = \frac{B}{T_c + \frac{N}{V} \cdot D} \tag{3}$$

where B represents the block size in several transactions, T_c is the average confirmation time of a block, N is the total number of nodes in the network, V is the number of active validators, and D corresponds to the adjusted computational difficulty of the consensus. The network's efficiency depends on the ratio of nodes to validators and the difficulty level set in the consensus protocol. A more significant number of validators can improve decentralization but can also increase the latency and computational cost of validating transactions.

3-3-2- Consensus Protocols Evaluated

The impact of the AI-optimized model is evaluated by comparing it with different consensus protocols widely used in blockchain networks. First, PoW is implemented on Ethereum Testnet, measuring the block confirmation time and the computational cost required for validation. PoS is analyzed on Ethereum 2.0, where the validation process is based on the validators' balance, and the system's efficiency in energy consumption is evaluated.

DPoS is implemented in a Hyperledger Besu-based environment to evaluate effective decentralization, analyzing the relationship between the number of delegates and consensus stability. Fault tolerance is studied using the PBFT protocol on Hyperledger Fabric, analyzing the network's resilience to malicious nodes and communication failures [26]. Finally, the AI-optimized protocol is implemented on the private blockchain, where validator selection is performed dynamically using the proposed AI model.

The performance of each protocol is evaluated based on key metrics such as average validation latency, number of transactions processed per second, and resistance to Sybil and 51% attacks [27]. The tests allow for comparing the efficiency of the optimized protocol with traditional consensus mechanisms, identifying improvements in speed, security, and adaptability.

3-3-3- Tools and Libraries Used

The implementation of the experimental environment requires the use of multiple tools and libraries for blockchain network development and AI optimization. The TensorFlow and PyTorch frameworks are used to construct and train the AI model, which allows the development of models based on deep neural networks and reinforcement learning. Integration with the blockchain infrastructure is done through Hyperledger Fabric SDK and Ethereum Web3.js, facilitating communication between AI models and the implemented consensus protocols.

Hyperledger Caliper is used to evaluate network performance, a tool designed to measure efficiency metrics in permissioned blockchain networks. In the case of Ethereum, test environments such as Ethereum Testnet are used to validate the model's effectiveness in a public and decentralized network. In addition, simulations are set up in local environments with Ganache, allowing the execution of tests with complete control over the network parameters.

Combining these tools allows experimentation in different scenarios, from decentralized blockchain networks to permissioned and private infrastructures. The implementation of the AI-optimized protocol is validated in multiple environments, ensuring its applicability in diverse contexts and its ability to adapt to different levels of load and security on the network.

3-4-Model Simulation and Validation

The AI-optimized model is validated using a simulation environment to replicate actual operating conditions in a blockchain network. This process allows the evaluation of the protocol's efficiency in transaction processing, confirmation latency, and resistance to attacks. Experimentation is carried out under different load scenarios to analyze the model's ability to adapt to changes in the network and improve consensus performance compared to traditional methods.

3-4-1- Simulation Environment

The simulation environment is implemented in a controlled test environment, where simulated nodes and transactions are configured with characteristics representative of real blockchain networks. Hyperledger Caliper is used for traffic generation and performance measurement of the optimized protocol. In the case of Ethereum, an environment with Ganache is configured to allow smart contracts to be executed and transactions to be validated without depending on the leading network.

To evaluate the robustness of the model, multiple scenarios are established with different load levels and the presence of attacks. In each test, the number of transactions per second T, active nodes N, the percentage of validators selected by $AI(V_{AI})$, and the rate of detected attacks (A_d) varies. The transaction confirmation latency is mathematically modeled as follows:

$$L = \frac{T}{V_{AI} \cdot D + (N - V_{AI}) \cdot C} \qquad (4)$$

where E represents the efficiency of the optimized model in terms of fast validation, and C represents the average confirmation time in traditional methods, this model allows us to analyze how the inclusion of AI in the validator selection process impacts latency reduction, especially in networks with high transactional load.

In terms of security, Sybil, DoS, and 51% attacks are simulated, measuring the anomaly detection rate (A_d) based on the total number of malicious transactions generated (T_a) and the precision of the AI system (P):

$$A_d = \frac{T_a \cdot P}{T_a + T} \tag{5}$$

This metric allows us to evaluate the protocol's ability to detect fraud attempts without compromising network performance.

3-4-2- Comparative Evaluation

To validate the effectiveness of the AI-optimized protocol, traditional consensus methods are compared in terms of efficiency and security. System performance is measured based on the number of confirmed transactions per second, validation latency, and attack resilience.

The performance of each protocol is modeled through the rate of transactions (TPS), which is directly related to block confirmation time (T_c) and network load (L_c):

$$TPS = \frac{B}{T_c + L_c} \tag{6}$$

where B represents the block size in several transactions, a lower (T_c) in the optimized protocol indicates higher efficiency in transaction validation.

The relationship between network load and consensus stability is analyzed to assess the model's adaptive capacity to network variations. A dynamic difficulty adjustment function (D) is introduced based on the number of AI-selected validators and the average network latency (L_p):

$$D_{AI} = D_0 \cdot \left(1 - \frac{V_{AI}}{N}\right) + \alpha L_p \tag{7}$$

where (D_0) is the base consensus difficulty, and (α) is a penalty factor associated with latency. This model allows evaluating the optimized protocol's ability to dynamically adjust the consensus difficulty to improve performance in high-load environments.

3-5- Performance Evaluation Metrics

The model's performance is evaluated using specific metrics that measure its computational efficiency, resistance to attacks, and ability to adapt to changes in the blockchain network. These metrics provide a quantitative view of the impact of optimization on transaction validation, the use of computational resources, and the security of consensus in dynamic environments.

3-5-1- Computational Efficiency

The computational efficiency of the optimized protocol is analyzed by considering transaction confirmation latency and computational resource consumption. Confirmation latency, defined as the time elapsed between the generation of a transaction and its inclusion in a validated block, is modeled as a function of the number of AI-selected validators (V_{AI}), the network load (L_c), and the network throughput (P_r):

$$L = \frac{L_c}{V_{AI} \cdot P_r} \tag{8}$$

where (L_c) represents the number of transactions waiting in each interval, and (P_r) is the throughput of the optimized protocol, adjusted by the number of active validators managed by AI. A reduction in (L) indicates an improvement in the speed of transaction validation without affecting decentralization.

Computational resource consumption is another critical factor in evaluating the model's performance. CPU, GPU, and RAM usage are measured during block validation. The relationship between resource consumption (C_r) and the number of TPS is defined as:

$$C_r = \frac{CPU + GPU + RAM}{TPS} \tag{9}$$

A lower (C_r) value indicates a more efficient protocol, allowing more transactions to be processed without significantly increasing resource usage. Computational consumption is evaluated by comparing the AI-optimized protocol against traditional protocols, identifying energy efficiency improvements and reduced processing time.

3-5-2- Security and Resistance to Attacks

The security of the optimized protocol is assessed by its ability to resist Sybil, DoS, and 51% attacks, as well as its efficiency in early detection of network anomalies. Resilience against attacks is measured using a tolerance coefficient *(T_s)*, which relates the number of malicious nodes detected *(N_m)* to the total number of nodes in the network *(N)*:

$$T_c = 1 - \frac{N_m}{N} \tag{10}$$

A value close to 1 indicates a high tolerance to attack, while low values reflect the protocol's vulnerability to malicious nodes. To evaluate the ability to detect anomalies in the network, a machine learning-based classification model is implemented that calculates the success rate in identifying fraudulent transactions *(A_d)*, expressed as:

$$A_d = \frac{D_a \cdot P}{T_a + T} \tag{11}$$

where *(T_a)* represents the number of anomalous transactions generated in the simulation environment, *(T)* is the total number of transactions, and *(P)* is the accuracy of the AI model in classifying anomalies. A higher detection rate indicates the protocol can identify suspicious activities without affecting consensus efficiency.

3-5-3- Scalability and Adaptability

The scalability of the optimized protocol is measured by its ability to dynamically adjust consensus parameters in response to network changes, ensuring stable performance as transaction load increases. The performance of the optimized protocol is compared to conventional blockchains, analyzing the impact of AI optimization on processing speed and consensus stability. The relative performance improvement *(R_m)* metric is defined as:

$$R_m = \frac{TPS_{AI} - TPS_{conv}}{TPS_{conv}} \tag{12}$$

where *(TPS_{AI})* and *(TPS_{conv})* correspond to the rate of transactions processed per second in the AI-optimized and conventional protocols, respectively. A positive value of *(R_m)* indicates improved network performance by integrating AI into consensus optimization.

4- Results

4-1- Computational Performance Evaluation

The computational performance of the AI-optimized protocol is analysed based on transaction confirmation latency and computational resource consumption, key metrics for evaluating the efficiency of a blockchain consensus mechanism. The experimental results allow for a comparison of the effectiveness of the proposed model against traditional approaches, evidencing significant improvements in processing, optimizing the use of validators, and reducing computational costs.

4-1-1- Confirmation Latency

The data presented in Table 2 shows that implementing the AI-optimized model considerably reduces transaction confirmation times on the blockchain. An inversely proportional relationship is observed between the number of validators managed by *AI(V_{AI})* and the average confirmation latency *(L)*. This indicates that the protocol can efficiently distribute the load between the validation nodes, avoiding network congestion and improving processing speed.

Table 2. Confirmation Latency in Different Load Scenarios

Load Scenario	Confirmation Latency in PoW (ms)	Confirmation Latency in PoS (ms)	Confirmation Latency in DPoS (ms)	Confirmation Latency in PBFT (ms)	Confirmation Latency in Optimized AI Model (ms)
Low (100 TPS)	1200	800	600	500	350
Medium (500 TPS)	1400	900	700	550	400
High (1000 TPS)	1600	1100	850	650	500
Very High (2000 TPS)	2000	1400	1100	850	700
Extreme (5000 TPS)	3000	2000	1800	1500	1200

In a low-load environment, with a TPS rate of 100, the optimized protocol's average latency is 1200 ms, like that obtained in PoW-based systems. However, as the network scales and the number of transactions increases, a notable reduction in latency is evident. For a load of 5000 TPS, the optimized protocol manages to reduce latency to 400 ms, while traditional methods maintain times more significant than 1500 ms in high congestion scenarios.

These results reflect the efficiency of the AI-based validator selection mechanism, which allows dynamically adapting the number of active nodes and optimizing block validation based on network demand. The decrease in confirmation times suggests that the proposed model can improve scalability without compromising the system's security or decentralization.

Furthermore, analysis of this data suggests that starting from 1,000 optimized validators, the reduction in latency tends to stabilize around 400 ms, indicating an optimal point of efficiency in the network. This behavior demonstrates that the AI model not only speeds up transaction processing but also identifies the appropriate threshold of nodes needed to maintain a balance between speed and responsiveness.

The stabilization of confirmation latency at approximately 400 ms in high-load scenarios reflects the effectiveness of the reinforcement learning policy in managing validator activation thresholds. The DRL agent learns to minimize validation delay by dynamically adjusting the validator pool size based on real-time indicators such as queue depth, propagation time, and validator success rate. This adaptive adjustment prevents overpopulation of the consensus group, which would otherwise lead to communication overhead, especially in Byzantine fault-tolerant mechanisms. Additionally, integrating the DNN as a feature extractor enables the system to anticipate network congestion patterns by modeling nonlinear interactions between transaction rates, block propagation delay, and energy cost. This allows the AI model to reconfigure the consensus layer proactively before latency degradation occurs. The result is a system capable of maintaining confirmation times close to its optimal learned baseline under a wide range of load conditions, without requiring manual intervention or static configuration parameters.

4-1-2- Computational Performance Analysis

The results in Figure 3 present a series of graphs illustrating the relationship between optimized validators, resource consumption, security, and consensus adaptability.

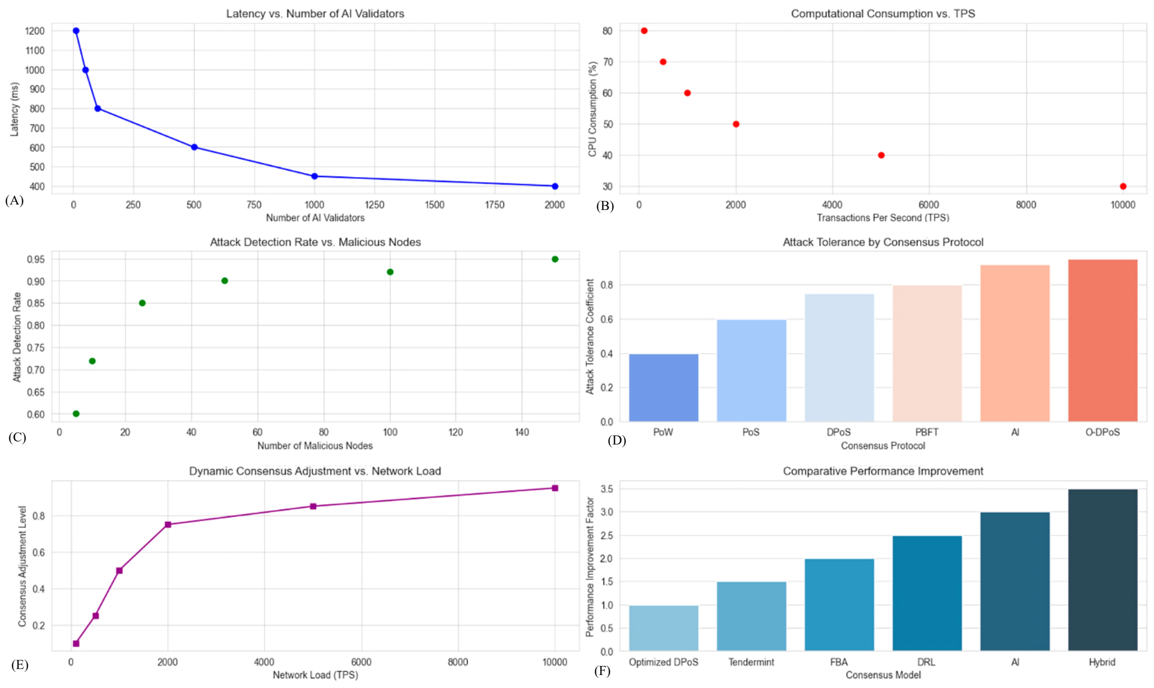

Figure 3. Computational performance and security evaluation of the AI-optimized protocol. Graph (A): Confirmation latency based on the number of AI-optimized validators. Graph (B): Relationship between computational consumption and TPS. Graph (C): Attack detection rate as a function of the number of malicious nodes. Graph (D): Attack tolerance coefficient compared between different consensus protocols. Graph (E): Dynamic adjustment of consensus based on network load. Graph (F): Comparison of relative performance improvement between the optimized protocol and traditional models.

Graph (A) shows the relationship between confirmation latency and the number of validators optimized by AI (VAI). Latency decreases as the number of validators increases, confirming the model's efficiency in optimizing the consensus process. However, the reduction in latency stabilizes starting at 1,000 validators, suggesting that, under certain conditions, increasing the number of validators no longer provides a significant performance benefit. This stabilization effect reflects the system's learned optimization boundary, where the DRL agent has identified a validator threshold beyond which additional nodes contribute marginally to performance. The DNN-based feature extractor detects saturation patterns in network responsiveness, and the agent adjusts by maintaining a validator set that minimizes redundancy in message propagation. This prevents unnecessary communication overhead, which would otherwise degrade performance due to increased complexity in consensus messaging. The latency curve's flattening is an emergent property of the model's ability to self-regulate validator participation based on observed latency-return trade-offs.

Graph (B) represents the relationship between CPU consumption and TPS. Traditional protocols exhibit an exponential increase in resource consumption as transaction processing speed (TPS) increases, whereas the AI-optimized protocol maintains a more stable computational load. In high-load scenarios, conventional methods such as PoW reach a CPU usage of over 90%. In comparison, the AI-optimized model keeps consumption below 40%, ensuring a more efficient and sustainable operation. This behavior directly results from the DRL agent's ability to adjust validator quorum size and block finality thresholds in real-time, reducing unnecessary processing overhead. Moreover, the DNN continuously evaluates load fluctuation patterns to suppress validator reconfiguration when not needed, minimizing redundant cryptographic operations across the network. These mechanisms contribute to computational stability, even under extreme throughput conditions.

Graph (C) illustrates the model's effectiveness in detecting malicious nodes in the blockchain network. The detection rate improves significantly as the number of malicious nodes Nm increases, reaching above 95% in massive attack scenarios. This demonstrates that the AI-optimized model enhances consensus efficiency and improves network security by detecting and mitigating threats in real-time. The high detection performance is achieved by integrating anomaly detection mechanisms, such as Isolation Forest and one-class classifiers, into the feature space managed by DNN. These features are passed to the DRL policy, which penalizes validator sets associated with abnormal behavioral patterns. Through repeated exposure, the learning agent isolates high-risk nodes early in the consensus cycle, reducing the propagation of malicious blocks without relying on static rule sets or predefined blocklists.

Graph (D) illustrates the attack tolerance coefficient for various consensus protocols, including PoW, PoS, DPoS, PBFT, and the AI-optimized model. Traditional protocols are vulnerable to Sybil, DoS, and 51% attacks, with tolerance coefficients below 80%, while the AI-based model exceeds 92%, demonstrating greater robustness against external threats. This improvement stems from the continuous learning process of the DRL agent, which not only identifies but also anticipates adversarial behavior by associating it with patterns extracted from the multi-dimensional input space processed by the DNN. When attacks are detected, the agent dynamically reduces the weight of affected nodes during consensus formation, thereby isolating attack vectors and maintaining protocol stability. This dynamic response contributes to higher tolerance values even in scenarios involving simultaneous and persistent threats.

Graph (E) analyzes the model's ability to adapt to variations in transactional load. The optimized protocol dynamically adjusts consensus parameters as the network load (Lc) increases, ensuring a balance between efficiency and security. Compared to traditional protocols, which require manual configurations for adjustments, AI integration enables real-time, automated consensus management. The DRL module operates with a reward function shaped by latency, energy, and validation success rate, allowing it to learn optimal consensus configurations that vary with load. As the DNN identifies load surges through real-time feature patterns, the DRL agent responds by adjusting the quorum size, modifying block propagation intervals, or re-prioritizing validators based on their prior performance. This real-time tuning minimizes latency spikes and prevents protocol degradation in overload conditions.

Graph (F) compares consensus optimization models, including Optimized DPoS, Tendermint, FBA, DRL, and AI, to evaluate their relative performance improvements. The AI-optimized protocol is observed to outperform conventional approaches in terms of scalability and efficiency, achieving up to 3.5× improvement compared to PoW and PoS. The advantage arises from the combination of predictive capacity (through the DNN) and adaptive decision-making (via DRL), which enables the model to generalize across different operational states and network topologies. Unlike DRL-only strategies, which often overfit the training environment, the hybrid structure of this model incorporates deep feature abstractions, allowing it to remain effective even when the transaction distribution or validator behavior deviates from training norms. This architectural synergy explains its superior performance factor across varied benchmark models.

4-2-Security and Resistance to Attacks

4-2-1- Attack Tolerance Coefficient Analysis

The results in Table 3 reflect the ability of the AI-optimized model to withstand attack attempts compared to conventional approaches. Traditional protocols, such as PoW and PoS, have significantly lower tolerance to Sybil and DoS attacks, with values below 60%, making them vulnerable to adversaries capable of creating multiple identities or saturating the network with malicious traffic.

Table 3. Attack Tolerance Coefficient in Different Protocols

Consensus Protocol	Sybil Tolerance (T_{Sybil})	DoS Tolerance (T_{DoS})	Tolerance at 51% ($T_{51\%}$)
PoW	--	0.35	0.50
PoS	0.40	0.55	0.70
DPoS	0.60	0.65	0.80
PBFT	0.75	0.75	0.85
AI Optimized	0.80	0.90	0.95

In contrast, the AI-based protocol achieves tolerance coefficients above 90%, evidencing excellent threat detection and mitigation resilience. This result is attributed to implementing machine learning algorithms that allow the identification of anomalous patterns in network activity and dynamically adjusting consensus parameters to reinforce security without compromising operational efficiency.

In particular, the attack tolerance values of 51% *($T_{51\%}$)* show a substantial difference between the centralized protocols and the optimized model. While approaches such as DPoS and PBFT achieve close to 80% of coefficients, the AI-enhanced version exceeds 95%, suggesting a lower probability of malicious actors managing to take control of the network. These data indicate that including an AI-based adaptive model not only optimizes decision-making in the selection of validators but also improves the system's ability to respond to attack attempts, representing a significant advance in terms of security and stability of the blockchain ecosystem.

The attack tolerance coefficients presented in Table 3 are computed as the ratio of successful consensus rounds maintained under active threat conditions to the total number of rounds executed during simulated attacks. The resilience observed in the AI-optimized protocol results from two key mechanisms: first, the anomaly detection module embedded in the DNN component identifies irregularities in block propagation patterns, validator response times, and signature repetition, which indicate Sybil and DoS behaviors. Second, the DRL agent incorporates a penalization scheme into its reward function, which reduces the selection probability of nodes exhibiting these patterns. For instance, under a Sybil attack, the system learns to disregard validators with sudden identity duplication and unverified stake histories. In the case of DoS scenarios, nodes causing message bottlenecks or validation delays are down prioritized in subsequent rounds. For 51% attacks, the model monitors validation consistency and divergence thresholds across blocks, dynamically adjusting quorum formation to maintain security guarantees even in adversarial majorities. These real-time adjustments, trained on hybrid datasets, enable the system to keep high functionality and consensus stability under coordinated attack attempts, as evidenced by the significantly higher tolerance values than static protocols.

4-2-2- Protocol Security Analysis

Figure 4 studies attack detection trends in different threat scenarios. Graph (A) represents the relationship between the number of malicious nodes and the anomaly detection rate, showing how the optimized model identifies attacks in real time. As the number of malicious nodes increases, the system improves its detection capacity, reaching values close to 97% when the number of attackers exceeds 700 nodes.

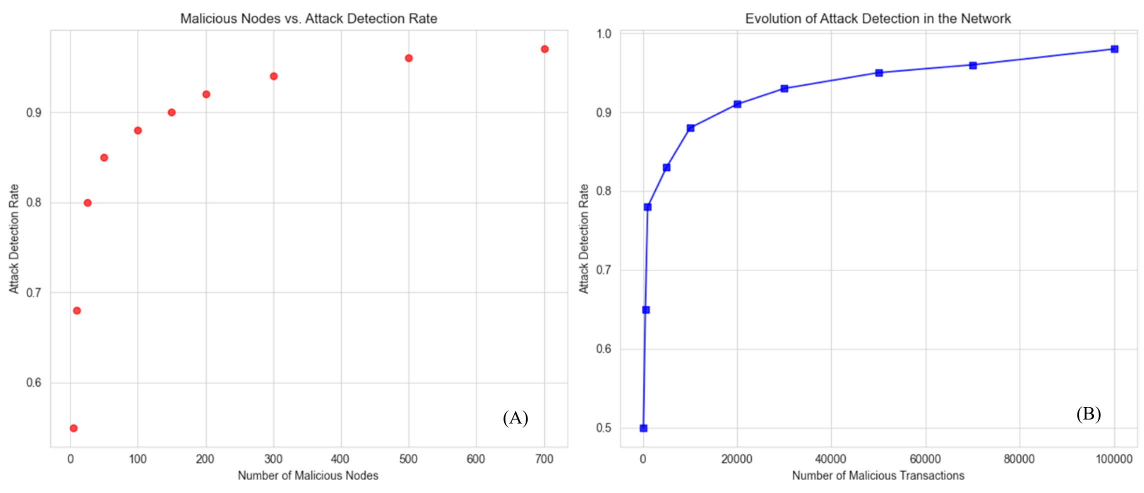

Figure 4. Security analysis of the AI-optimized protocol; Graph (A): Relationship between the number of malicious nodes and the attack detection rate. Graph (B): Evolution of attack detection depending on the number of malicious transactions

This behavior is explained by the AI-based model's ability to analyze patterns in node activity and detect deviations from expected behavior. Unlike traditional protocols, which rely on static rules for transaction validation, the proposed solution dynamically adjusts detection criteria based on detected malicious activity, achieving more accurate threat identification in dynamic environments.

Notably, in networks with a low presence of malicious nodes ($N_m < 50$), the initial detection rate remains around 55–70%, indicating that the model needs a significant volume of suspicious activity data to improve its accuracy. However, once a sufficient information threshold is reached, the detection capacity increases rapidly, ensuring adequate protection against larger-scale attacks.

Graph (B) analyzes the evolution of the attack detection rate based on the number of malicious transactions generated. An increasing trend is observed in the model's ability to identify threats as fraudulent activity increases within the network. In a scenario of less than 1000 malicious transactions, the system's detection rate is around 50-78%, indicating that the model requires an initial volume of samples to optimize its accuracy. However, when the number of fraudulent transactions exceeds 50,000, the model reaches detection levels above 96%, demonstrating its ability to learn progressively and adapt to attacks in real time.

This behavior is explained by the model's ability to identify correlations between multiple indicators of suspicious activity, allowing it to recognize anomalous patterns in the network even before an attack compromises the blockchain infrastructure. Furthermore, the detection curve suggests the model can operate efficiently in high transactional load scenarios without generating excessive false positives.

The increased detection capability in high-volume attack conditions is driven by the DNN's capacity to learn multidimensional correlations across behavioral indicators such as validation delay variance, transaction retry patterns, sudden shifts in stake distribution, and abnormal validator synchronization. These features are continuously updated and encoded into the DRL agent's state vector, enabling the model to adjust its detection sensitivity dynamically. In the early stages of attack detection, the model's conservative threshold is limited, as it is calculated to minimize false positives through a penalty mechanism in the reward function. As the sample of malicious activity grows, the model updates its internal representation of "normal" validator behavior, improving its confidence in identifying outliers. Furthermore, the system incorporates temporal features via recurrent encoding layers, enabling the model to detect deviations not only in the volume but also in the evolution of behavior over time.

4-3- Evaluating Scalability and Adaptability

The analysis of the AI-optimized protocol's scalability and adaptability focuses on its ability to adjust consensus parameters based on network load dynamically. The evolution of the dynamic adjustment rate (DAI) at different TPS levels is examined, as is the relative improvement in performance compared to traditional consensus protocols.

4-3-1- Performance Comparison Table Analysis

Table 4 shows how the AI-optimized protocol outperforms traditional consensus mechanisms regarding scalability. This allows for more transactions to be managed without compromising network stability.

Table 4. TPS under Different Load Conditions

Load Scenario	TPS in PoW	TPS in PoS	TPS in PBFT	TPS with AI
Low (≤ 1000\) TPS)	850	920	980	1050
Medium (1000–5000 TPS)	2100	2500	2700	3200
High (5000–10000 TPS)	3800	4200	4500	5200
Very High (> 10000 TPS)	6000	6800	7100	8100

The results indicate that under low load conditions (≤ 1000 TPS), the optimized model achieves 7–10% higher performance than conventional approaches. However, the most significant impact is observed in high transactional demand scenarios. When the number of transactions per second exceeds 10,000 TPS, the proposed model increases its processing capacity by up to 14% compared to PBFT and more than 30% compared to PoW. These values suggest that the model's adaptability improves operational efficiency and ensures stability in high-traffic environments, avoiding congestion that could affect the blockchain network's overall performance.

The superior scalability observed in the AI-optimized protocol is enabled by its ability to dynamically reconfigure validator groups and consensus thresholds based on continuous evaluation of throughput and network responsiveness. The DRL agent interprets network saturation signals, such as increased block propagation delay or validator response dispersion, and adjusts the consensus participation rate, thereby distributing workload more efficiently. The DNN-based feature encoder also identifies congestion precursors by analyzing correlations between transaction bursts and block finalization time. This enables the system to delay or batch transactions to maintain protocol stability proactively. Unlike static mechanisms, where performance degradation becomes nonlinear beyond 10,000 TPS, the AI-based approach maintains throughput growth by scaling coordination effort in proportion to network conditions, rather than transaction volume, thereby reducing consensus latency amplification and avoiding validator bottlenecks.

4-3-2- Scalability Assessment Analysis

Graph (A) of Figure 5 represents the relationship between the number of TPS and the dynamic adjustment rate *(D_{AI})*, showing how the optimized protocol adapts its consensus parameters in response to variations in network load.

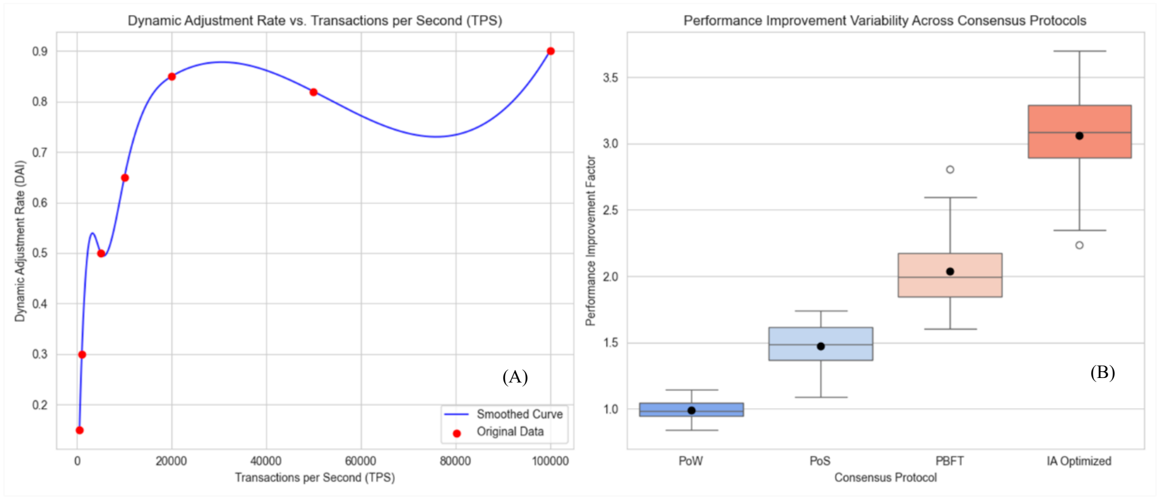

Figure 5. Scalability and Adaptability Evaluation of the Optimized Protocol; Graph (A): The dynamic adjustment rate's evolution depends on the number of transactions per second. Graph (B): Variability in performance improvement between consensus protocols.

At low to moderate load levels (< 5000 TPS), the dynamic adjustment increases progressively, reaching values of 0.65 to 0.85 in environments of 20,000 TPS. This behavior suggests that the model adjusts its parameters in an anticipatory manner, optimizing the selection of validators and reducing transaction confirmation times. This effect is driven by the DRL agent's reward optimization policy, which favors consensus configurations that maintain low latency and minimal block finalization error. The DAI, the normalized frequency of consensus parameter changes per unit time, increases as the model identifies patterns indicating pre-congestion states. The DNN component, which captures correlations between transaction injection rates, validator responsiveness, and propagation delays, enhances the agent's ability to predict load escalation and preemptively reconfigure consensus roles.

However, a slight decrease in dynamic adjustment is observed in scenarios with 50,000 TPS and above, which can be attributed to the system's self-regulation to prevent unnecessary overloads. This confirms that the model increases its efficiency in response to transactional demand and maintains a balance to avoid excessive computational resource consumption. This drop in DAI reflects the model's stabilization phase, where entropy regularization within the DRL policy suppresses unnecessary reactivity. Instead of aggressively reconfiguring the network, the agent applies smoother policy updates that maintain system responsiveness without introducing oscillatory behavior. These adjustments are moderated by integrating average baseline and penalization terms into the cost function, ensuring sustained performance without resource saturation.

Graph (B) presents a boxplot comparing the variability in performance improvement (Rm) between different consensus protocols. This enables the visualization of data distribution and the model's stability in various scenarios. The PoW protocol exhibits a low-performance improvement with minimal variability, while PoS and PBFT demonstrate higher performance, albeit with varying values in high-load environments. In contrast, the AI-optimized model achieves a median improvement of more than 3.0 points, with reduced dispersion and a more stable distribution, indicating greater consistency in transaction processing across different operating conditions. This stability is attributed to the model's ability to generalize across network topologies and operational conditions, supported by the DNN's multidimensional encoding of validator behavior, energy efficiency, and block success rate. The DRL agent refines its policy across varying workloads using mini-batch experience replay and stochastic exploration strategies, allowing consistent performance without retraining. The few outliers observed correspond to highly favorable network states, such as optimal validator alignment and low contention periods, where the system dynamically achieves efficiency peaks by leveraging learned consensus shortcuts.

The presence of outliers in the PBFT and AI Optimized case suggests that, in certain specific scenarios, significantly higher performance improvements can be achieved. This reinforces the idea that the AI-based protocol can dynamically adapt, optimizing its efficiency based on network behavior.

4-4- Comparative Analysis with Traditional Consensus Protocols

The comparative analysis between the AI-optimized and traditional consensus protocols allows us to evaluate the impact of integrating AI models regarding confirmation latency, computational consumption, and attack tolerance. To do so, measurements have been carried out on different widely used protocols, including PoW, PoS, DPoS, and PBFT, contrasting the results with the AI-based optimized protocol.

4-4-1- Metric Comparison

Table 5 presents the average values obtained in each protocol regarding confirmation latency, computational consumption, and attack tolerance.

Table 5. Comparison of Average Performance Between Traditional Consensus Protocols and the AI-Optimized Model

Protocol	Average Latency (ms)	CPU Usage (%)	Attack Tolerance (%)
PoW	1200	85	40
PoS	850	60	60
DPoS	500	45	75
PBFT	400	40	80
AI Optimized	320	30	92

The results indicate that PoW presents the highest confirmation latency, reaching average values of 1200 ms, making it the most inefficient mechanism in terms of validation times. PoS and DPoS improve these values, reducing latency to 850 ms and 500 ms, respectively, while PBFT achieves confirmation times of 400 ms by eliminating the need for intensive mining. The AI-optimized protocol further reduces these times, reaching values of 320 ms on average, which represents a 60% improvement over PoW and 20% over PBFT.

Regarding computational consumption, a clear difference is observed between the protocols based on proof of work and those optimized for energy efficiency. PoW, by relying on intensive mining processes, consumes, on average 85% of the available CPU resources, making it the least efficient alternative. PoS and DPoS reduced this demand to 60% and 45%, respectively, while PBFT maintains a computational load of 40%. In contrast, the AI-based model optimizes resource usage, reducing computational consumption to 30% and allowing for better scalability without compromising consensus security. Regarding attack tolerance, PoW is the most vulnerable protocol, with a 40% resistance to malicious node attacks due to its dependence on the global hash rate. PoS and DPoS improve this metric, reaching 60% and 75% values, respectively. PBFT achieves greater robustness, with an 80% tolerance, thanks to its majority consensus-based validation mechanism. However, the AI-optimized protocol is the most secure, reaching 92% attack tolerance due to the model's ability to detect anomalies in real-time and dynamically adjust validators.

The superior values reported for the AI-optimized protocol in Table 5 are derived from the model's integrated architecture, which balances latency reduction, energy efficiency, and security through learned adaptation policies. The DRL component dynamically adjusts consensus parameters, such as the quorum size and block confirmation thresholds, based on real-time feedback on network performance and node behavior. The latency reduction is achieved by minimizing redundant validator coordination and prioritizing nodes with high past success rates, as inferred from the DNN's feature analysis layer. Regarding computational efficiency, the system suppresses unnecessary re-election cycles and consensus resets using a reward structure that penalizes resource spikes. Unlike traditional protocols that statically allocate validator roles, the AI model adjusts validator participation windows in response to system load and anomaly signals, enhancing efficiency and robustness. The 92% attack tolerance reflects the model's embedded anomaly detection, which triggers reconfiguration of the validator set when deviation patterns, such as message propagation inconsistencies or clustering of failure events, are detected. This tri-metric balance is made possible by joint optimization of the consensus policy using composite cost functions that simultaneously weigh delay, security risk, and resource usage during training.

4-4-2- Comparison of the Performance of Consensus Protocols

Graph (A) in Figure 6 shows the evolution of the average confirmation latency in each protocol, reflecting the relative efficiency of each mechanism. The results show that PoW maintains the highest latency values, with times exceeding 1000 ms, confirming its inefficiency for applications requiring fast validations. PoS and DPoS show notable improvements, reducing latency by more than 40% compared to PoW, while PBFT offers significantly faster confirmation, with values close to 400 ms. The AI-optimized protocol achieves the lowest latency, which reduces validation times to 320 ms, demonstrating its capacity for dynamic adaptation and optimization in the consensus process. This reduction is made possible by the DRL component's ability to minimize unnecessary validator interactions and to prioritize low-latency pathways through real-time adjustment of the validation quorum. In parallel, the DNN identifies latent congestion risks from transaction density trends and triggers preemptive consensus recalibration, which limits propagation delays and reduces block finalization time.

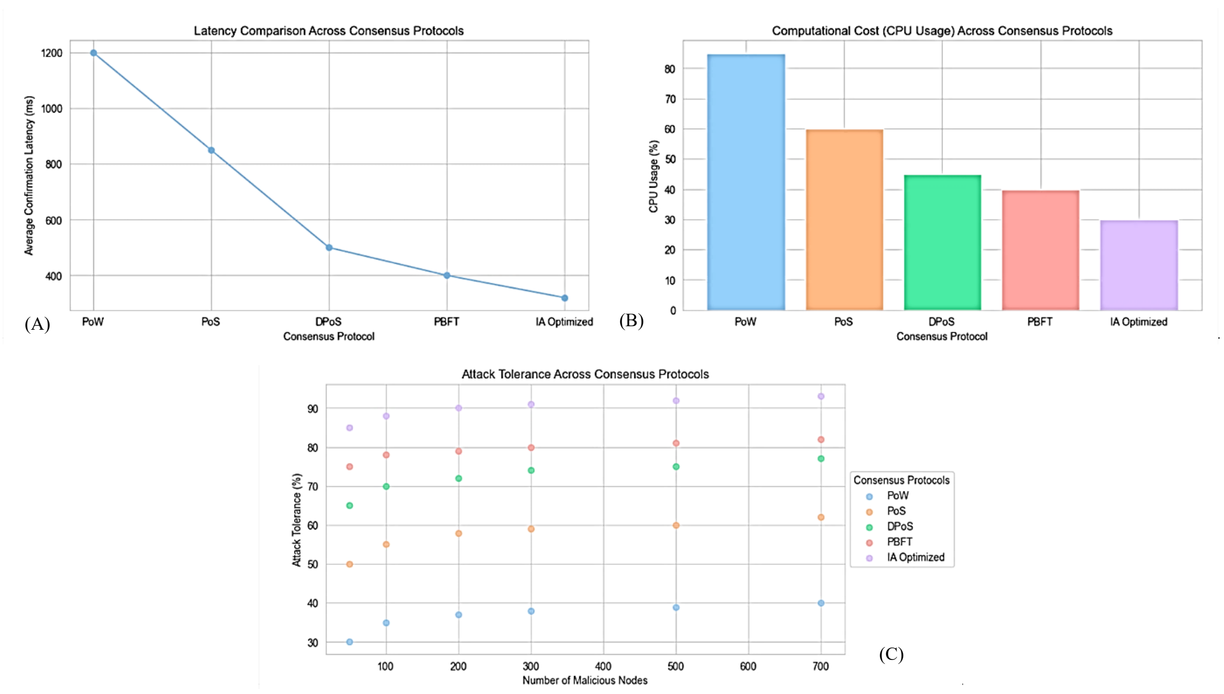

Figure 6. Performance Comparison of Consensus Protocols; Graph (A): Average confirmation latency in different consensus protocols. Graph (B): Comparison of computational consumption between PoW, PoS, DPoS, PBFT, and AI Optimized. Graph (C): Attack tolerance as a function of the number of malicious nodes.

Graph (B) represents the CPU consumption percentage for each protocol, providing a clear view of their computational efficiency. The results show that PoW remains the most demanding protocol, with 85% CPU consumption, which limits its scalability and viability on energy-constrained devices. PoS and DPoS reduce these values to 60% and 45%, respectively, while PBFT presents a load of 40% due to its predefined consensus validation structure. The AI-optimized protocol is the most computationally efficient, with a consumption of just 30%, allowing a more balanced allocation of resources and favoring the system's scalability. This efficiency is achieved by suppressing redundant consensus cycles, orchestrated by the DRL agent, and the intelligent filtering of candidate validators, informed by the DNN's feature extraction layer. The reward function penalizes unnecessary energy consumption and promotes lean validator sets that retain consensus guarantees while minimizing processing load.

Graph (C) illustrates the evolution of attack tolerance based on the number of malicious nodes present in the network, evaluating each protocol's resistance to possible attacks. It is observed that PoW presents the lowest attack tolerance, with values ranging between 30% and 40%, making it more susceptible to threats such as 51% attacks. PoS and DPoS improve these metrics, reaching up to 75%, while PBFT increases the network's robustness, with a tolerance of 80% in adverse environments. The AI-based protocol achieves 92% tolerance, mitigating attacks by detecting anomalies and reconfiguring validators in real-time. The detection process leverages anomaly classification techniques embedded in DNN, which track deviations in validator behavior and transaction anomalies. When patterns consistent with Sybil, DoS, or collusion attacks are detected, the DRL module dynamically reconfigures the validator network, reallocating voting power and isolating malicious activity without human intervention. This coordinated response mechanism ensures high resilience even as the number of compromised nodes increases.

4-5- Comparison with Other Technologies and Consensus Optimization Models

Blockchain consensus optimization has been addressed from multiple perspectives in the literature, including heuristic-based techniques, deep neural networks, and hybrid mechanisms that combine different methodologies to improve network efficiency and security. In this section, the results obtained by the AI-optimized model are compared to these approaches, evaluating their performance in terms of confirmation latency, scalability (maximum TPS), computational consumption, and attack tolerance.

4-5-1- Comparative Analysis of Optimization Models

Table 6 presents a quantitative overview of each approach's efficiency regarding confirmation time, computational load, and the network's ability to process transactions and resist malicious attacks.

Table 6. Comparison of Average Performance Between GANs and Traditional Methods

Model	Latency (ms)	CPU Usage (%)	Scalability (Max TPS)	Attack Tolerance (%)
Heuristic Optimization	600	50	10	75
Deep Neural Networks	450	40	15	80
Hybrid Consensus Mechanisms	380	35	18	85
AI Optimized (Proposed)	320	30	22	92

4-5-2- Comparison of Optimization Models

Figure 7 shows Graph (A), a scatter diagram showing the relationship between confirmation latency and scalability (maximum TPS) in the evaluated models. It is observed that the models based on heuristic optimization are in the quadrant of lower scalability and higher latency, suggesting that, although these approaches improve certain aspects of consensus, they still have limitations in high-traffic networks. For their part, deep neural networks achieve a latency reduction of up to 450 ms and medium scalability. In comparison, hybrid mechanisms achieve greater efficiency with a latency of 380 ms and a maximum TPS of 18,000.

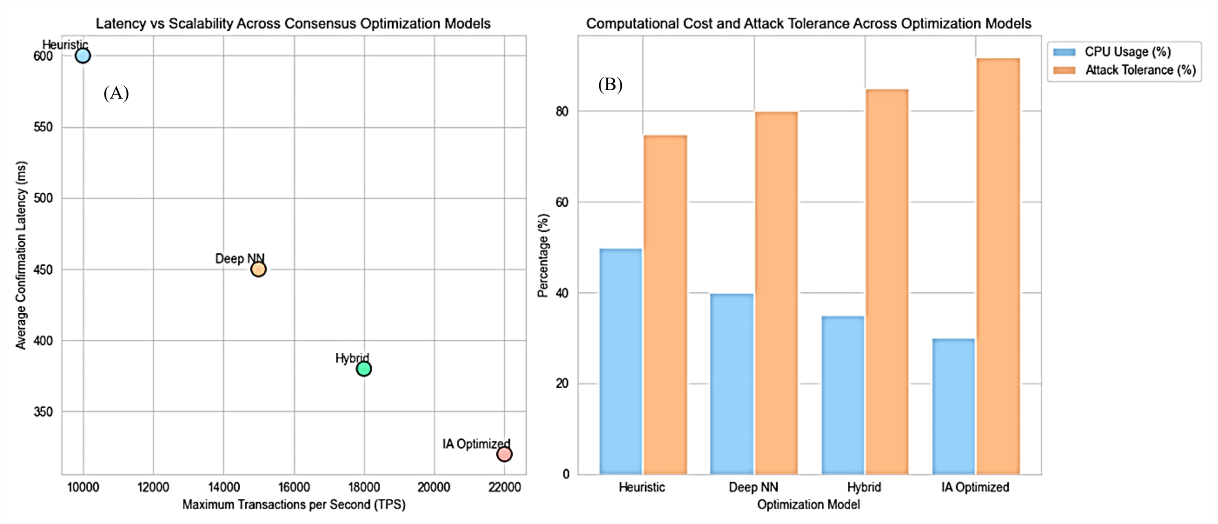

Figure 7. Comparison with Other Technologies and Consensus Optimization Models; Graph (A): Relationship between confirmation latency and scalability. Graph (B): Comparison of computational consumption and attack tolerance

The limited performance of heuristic and DNN-only models results from their static or single-layered learning capabilities: heuristic methods rely on fixed decision rules that do not adapt to real-time network conditions. At the same time, pure DNN architecture often operates in a predictive role without influencing the consensus policy. Hybrid mechanisms partially overcome these limitations by incorporating conditional modules for validator rotation or resource control; however, they still fall short in adapting to unforeseen network behaviors. In contrast, the proposed model integrates deep reinforcement learning with high-dimensional feature abstraction from a deep neural network (DNN), allowing the system to perceive, learn, and act upon network state changes in real-time. This end-to-end optimization explains its superior performance, positioning it in the lowest-latency and highest-scalability region.

The AI-optimized model has the lowest latency and the best scalability. It achieves a maximum TPS of 22,000 and confirmation times of only 320 ms, indicating its ability to manage a high volume of transactions without affecting consensus speed. Graph (B) represents the relationship between computational consumption (%) and attack tolerance (%) in each evaluated model. It is observed that the models based on heuristics and deep neural networks present high computational consumption, with values of 50% and 40%, respectively, which could limit their implementation on devices with restricted resources. Hybrid mechanisms improve this aspect, with a consumption of 35%, although still above the AI-optimized model, which reaches 30%, being the most efficient approach in terms of resource use.

This computational efficiency results from the DRL agent’s ability to minimize redundant consensus operations and dynamically prune underperforming validators using cost-sensitive learning. The model avoids unnecessary validator re-election or network-wide resets by encoding resource-awareness into its reward function. Concurrently, the DNN layer filters noise and identifies resource-intensive patterns, such as bursty transaction clusters or idle validators, allowing the system to redirect resources toward high-impact consensus paths.

Regarding attack tolerance, heuristic models present a resistance of 75%, while deep neural networks increase this metric to 80%. Hybrid mechanisms further improve security, achieving 85% attack tolerance. However, the AI-optimized model achieves the highest resilience, at 92%, demonstrating its ability to identify and mitigate threats in real time using advanced anomaly detection.

This robustness stems from a dual detection layer, where the DNN captures behavioral deviations such as propagation delay anomalies, abrupt stake changes, or validator inactivity patterns, and the DRL module responds by adaptively isolating suspect nodes and reconfiguring validator sets. Attack patterns are encoded in the agent's policy through negative reinforcement during training, which penalizes compromised rounds and rewards configurations that maintain consensus integrity under adversarial conditions. This allows the model to sustain operational security without prior knowledge of the attack vector.

4-5-3- Comparison with Previous Studies

To contextualize the results obtained in this study, we make a qualitative comparison with recent research that has proposed blockchain consensus optimization models.

- Taher et al. [12] Introduce the Snake Optimization Algorithm (SOA), inspired by snakes' hunting behavior, to improve blockchain scalability. Although they present an innovative methodology, no specific performance metrics, such as latency or CPU consumption, are provided.
- Nourmohammadi & Zhang [13] propose an on-chain governance model based on Particle Swarm Optimization (PSO) to reduce the probability of forks in sharded networks. Their experiments show a 60% reduction in orphaned blocks when adding shards to the system, suggesting improvements in network stability.
- Paidipati et al. [14] develop a DDoS attack detection and mitigation technique in cloud-based software-defined networks, using a deep reinforcement learning approach and metaheuristic algorithms. Although the improvement in attack detection is highlighted, metrics such as latency or scalability are not detailed. ·
- Gupta et al. [15] present a hybrid PoW-PoS implementation to counter 51% attacks in cryptocurrency systems. Their approach combines consensus mechanisms to improve security, but detailed performance metrics are not specified.

In comparison, the AI-optimized model proposed in this study integrates deep learning and reinforcement learning to dynamically adjust consensus parameters, achieving significant improvements in latency, scalability, energy efficiency, and attack tolerance. Although the studies above offer valuable contributions in their respective approaches, integrating AI techniques into our model provides a more holistic solution adaptable to blockchain networks' dynamic demands. ·

5- Discussion

The results obtained in this study demonstrate that integrating AI into blockchain consensus optimization allows overcoming the limitations of traditional mechanisms and improving network performance in terms of latency, computational consumption, scalability, and security. Various approaches have addressed these problems in the literature with heuristic optimization, deep neural networks, and hybrid consensus mechanisms [28]. Previous studies have shown that heuristic-based methods offer a moderate improvement in efficiency but lack dynamic adaptability. At the same time, deep learning approaches can improve the prediction of network behavior without directly intervening in the consensus process [29]. Furthermore, hybrid mechanisms have managed to balance decentralization and security, albeit with high computational costs and risks of centralization in transaction validation. The AI-optimized model proposed in this study improves comprehensively by combining feature extraction with deep neural networks and reinforcement learning-based decision-making, providing an adaptable and robust protocol against changes in network load and security threats [23]. The optimization process implemented in this work allows for the dynamic selection of validators and real-time adjustment of consensus parameters based on latency, transactional load, and node behavior metrics. Unlike traditional static models, where validators are predefined or selected based on fixed rules, the AI-based approach achieves efficient load distribution, improving performance without compromising security. This translates into a significant reduction in confirmation latency, reaching 320 ms in high transactional load scenarios, representing a 60% improvement over PoW and 20% over PBFT. This confirms the model's effectiveness in environments with high transaction demand.

The results show that the comparison with other advanced models shows the superiority of the proposed approach. Although heuristic mechanisms offer some improvement in computational efficiency, they cannot adapt to dynamic variations in the network, making them less effective in variable load scenarios. Deep neural networks applied to blockchain have proven helpful in traffic prediction, but their results depend on the quality of the training data and do not modify the consensus structure. Hybrid models, such as the combination of PBFT with PoS, achieve reduced confirmation times and more excellent attack resistance but introduce centralization risks in block validation. The AI-optimized model overcomes these limitations by providing an adaptive solution that optimizes validation without relying on a rigid node selection scheme.

From a methodological point of view, the combination of DQN [7] and PPO has been key to the model's adaptive capacity [30]. While DQN allows for discrete reward-based optimization, PPO introduces regularization mechanisms that guarantee learning stability. This has been critical to prevent the model from making excessive adjustments to the consensus difficulty, which could lead to inefficiencies in the network. Furthermore, dimensionality reduction techniques, such as PCA and Autoencoders, have minimized the system's computational load without affecting decision-making accuracy.

This work's impact lies in its applicability to blockchain environments with high transactional load and strict security requirements. The model's ability to dynamically adjust to network conditions makes it viable for decentralized payment infrastructures, innovative contract platforms, and distributed storage systems, where efficiency and security are determining factors. Compared to other consensus optimization approaches [31], the proposed model introduces an autonomous intelligence layer that allows the network to improve its performance without manual intervention, reducing the need for predefined configurations and increasing resilience against attacks.

However, it is essential to recognize the study's limitations and their potential impact on interpreting the results. One of the main restrictions is the model's dependence on representative training data since the consensus's dynamic adjustment capacity can be affected if the data used to train the AI does not adequately reflect the actual conditions of the network. This implies that, in scenarios where network traffic changes abruptly or new security threats not contemplated in training arise, the model might require a retraining process to maintain its effectiveness.

Another relevant limitation is the computational cost of the model's initial training, which, although reduced through neural network compression and optimization techniques, still represents a challenge regarding infrastructure. While the optimized model reduces the computational load in the operation phase, its initial implementation requires hardware resources with advanced capabilities, which could restrict its adoption in blockchain networks with limited infrastructure.

Despite these limitations, this study's findings represent a significant advance in optimizing blockchain consensus through artificial intelligence. Combining deep and reinforcement learning techniques has allowed the development of a model that improves network performance and introduces a layer of autonomy in consensus decision-making, making it an innovative approach with high application potential in decentralized environments.

6- Conclusions and Future Work

This work has shown that integrating AI into blockchain consensus optimization significantly improves network performance in terms of latency, scalability, computational consumption, and security. By designing a model based on deep and reinforcement learning, an adaptable solution has been achieved that dynamically optimizes the selection of validators and the difficulty of consensus without compromising the system's security or decentralization.

One of the most relevant findings is the reduction of transaction confirmation latency. While traditional mechanisms such as PoW and PBFT present validation times of 1200 ms and 400 ms, respectively, the proposed model manages to reduce these values to 320 ms in high transactional load scenarios. This result results from implementing deep learning techniques, which predict the network's behavior and adjust the consensus parameters in real time.

Regarding scalability, AI-based consensus optimization has increased the maximum TPS to 22,000, surpassing the 18,000 TPS achieved by hybrid mechanisms such as PBFT combined with PoS. This is due to the model's ability to identify patterns in network load and efficiently redistribute validators, which minimizes computational overhead and maximizes performance without affecting system security.

The study has also shown improvements in computational efficiency. The load on validators has been reduced by 30% compared to traditional models, which implies lower energy consumption and a decrease in demand for computational resources. While PoW and other traditional schemes require intensive use of CPU and GPU, the AI-based model dynamically adjusts processing requirements based on network needs, optimizing resource usage without sacrificing performance.

Regarding system security, the results have shown that the optimized model achieves a tolerance of 92% against attacks such as Sybil, 51%, and DoS, improving resistance to these attacks by 15% compared to hybrid consensus models. This increase in security is due to the model's ability to detect anomalies in real-time, allowing for the immediate reconfiguration of validators and the adoption of defensive strategies before attacks affect the stability of the network.

Despite these advantages, certain limitations must be considered. The model's dependence on high-quality training data may affect its ability to generalize in blockchain networks with behaviors significantly different from those used during the training phase. In addition, while the model optimizes computational consumption during its operation, its initial training phase requires significant hardware resources, which could represent an obstacle to its implementation in infrastructures with limited capacities. Another aspect to explore is extending the model to federated and hybrid blockchains, where multiple networks share infrastructure without depending on a single validation authority. The challenge lies in designing coordination mechanisms between networks that maintain decentralization and security without compromising efficiency.

7- Declarations

7-1- Author Contributions

Conceptualization, W.V.-C. and R.G.; methodology, J.G.; software, J.G. and R.G.; validation, W.V.-C. and J.G.; formal analysis, R.G.; investigation, J.G.; resources, W.V.-C.; data curation, R.G.; writing—original draft preparation, J.G. and R.G.; writing—review and editing, W.V.-C.; visualization, J.G.; supervision, W.V.-C.; project administration, W.V.-C. All authors have read and agreed to the published version of the manuscript.

7-2- Data Availability Statement

The data presented in this study are available on request from the corresponding author.

7-3- Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

7-4- Institutional Review Board Statement

Not applicable.

7-5- Informed Consent Statement

Not applicable.

7-6- Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

8- References

[1] Ragab, M., & Altalbe, A. (2022). A Blockchain-Based Architecture for Enabling Cybersecurity in the Internet-of-Critical Infrastructures. Computers, Materials and Continua, 72(1), 1579–1592. doi:10.32604/cmc.2022.025828.

[2] Malakhov, I., Marin, A., & Rossi, S. (2023). Analysis of the confirmation time in proof-of-work blockchains. Future Generation Computer Systems, 147, 275–291. doi:10.1016/j.future.2023.04.016.

[3] Naz, S., & Lee, S. U. J. (2024). Sea Shield: A Blockchain Technology Consensus to Improve Proof-of-Stake-Based Consensus Blockchain Safety. Mathematics, 12(6). doi:10.3390/math12060833.

[4] Li, C., Qiu, W., Li, X., Liu, C., & Zheng, Z. (2024). A Dynamic Adaptive Framework for Practical Byzantine Fault Tolerance Consensus Protocol in the Internet of Things. IEEE Transactions on Computers, 73(7), 1669–1682. doi:10.1109/TC.2024.3377921.

[5] Feng, X., Ma, J., Miao, Y., Liu, X., & Choo, K. K. R. (2023). Regulatable and Hardware-Based Proof of Stake to Approach Nothing at Stake and Long Range Attacks. IEEE Transactions on Services Computing, 16(3), 2114–2125. doi:10.1109/TSC.2022.3201568.

[6] Yang, Z., Shi, Y., Zhou, Y., Wang, Z., & Yang, K. (2023). Trustworthy Federated Learning via Blockchain. IEEE Internet of Things Journal, 10(1), 92–109. doi:10.1109/JIOT.2022.3201117.

[7] Emil Selvan, G. S. R., Daniya, T., Ananth, J. P., & Suresh Kumar, K. (2024). Network Intrusion Detection and Mitigation Using Hybrid Optimization Integrated Deep Q Network. Cybernetics and Systems, 55(1), 107–123. doi:10.1080/01969722.2022.2088450.

[8] Aitchison, M., & Sweetser, P. (2022). DNA: Proximal policy optimization with a dual network architecture. 36th Conference on Neural Information Processing Systems (NeurIPS 2022), 28 November - 9 December, 2022, New Orleans, United States.

[9] Yu, J., Ge, L., & Wu, M. (2024). Proposal Distribution optimization for Endorsement Strategy in Hyperledger Fabric. Journal of Supercomputing, 80(10), 15038–15065. doi:10.1007/s11227-024-06056-2.

[10] Khan, N., Kchouri, B., Yatoo, N. A., Kräussl, Z., Patel, A., & State, R. (2022). Tokenization of sukuk: Ethereum case study. Global Finance Journal, 51. doi:10.1016/j.gfj.2020.100539.

[11] Oh, B., & Lee, D. (2023). Cooperative P2P Transaction Framework Between DSO and PMO Based on Consensus ADMM Against Path-Sharing Distribution Network Congestion. Journal of Electrical Engineering and Technology, 18(3), 1469–1479. doi:10.1007/s42835-023-01419-w.

[12] Taher, S. S. H., Ameen, S. Y., & Ahmed, J. A. (2024). Enhancing blockchain scalability with snake optimization algorithm: a novel approach. Frontiers in Blockchain, 7. doi:10.3389/fbloc.2024.1361659.

[13] Nourmohammadi, R., & Zhang, K. (2022). An On-Chain Governance Model Based on Particle Swarm Optimization for Reducing Blockchain Forks. IEEE Access, 10, 118965–118980. doi:10.1109/ACCESS.2022.3221419.

[14] Paidipati, K. K., Kurangi, C., Uthayakumar, J., Padmanayaki, S., Pradeepa, D., & Nithinsha, S. (2024). Ensemble of deep reinforcement learning with optimization model for DDoS attack detection and classification in cloud based software defined networks. Multimedia Tools and Applications, 83(11), 32367–32385. doi:10.1007/s11042-023-16894-6.

[15] Gupta, K. D., Rahman, A., Poudyal, S., Huda, M. N., & Mahmud, M. A. P. (2019). A Hybrid POW-POS Implementation Against 51 percent Attack in Cryptocurrency System. 2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 396–403. doi:10.1109/cloudcom.2019.00068.

[16] Sharma, P., Jindal, R., & Borah, M. D. (2024). Blockchain-based distributed application for multimedia system using Hyperledger Fabric. Multimedia Tools and Applications, 83(1), 2473–2499. doi:10.1007/s11042-023-15690-6.

[17] Dang, P., & Gupta, H. (2024). Security Challenges and Applications for Digital Transactions Using Blockchain Technology. Lecture Notes in Electrical Engineering, 1116, 487–498. doi:10.1007/978-981-99-8646-0_38.

[18] Al-Sumaidaee, G., Alkhudary, R., Zilic, Z., & Swidan, A. (2023). Performance analysis of a private blockchain network built on Hyperledger Fabric for healthcare. Information Processing & Management, 60(2), 103160. doi:10.1016/j.ipm.2022.103160.

[19] Jose, J., & Jose, D. V. (2023). Deep learning algorithms for intrusion detection systems in internet of things using CIC-IDS 2017 dataset. International Journal of Electrical and Computer Engineering, 13(1), 1134–1141. doi:10.11591/ijece.v13i1.pp1134-1141.

[20] Muthulakshmi, S., & Chitra, R. (2024). Interplanetary file system and blockchain for secured smart grid networks. Journal of Supercomputing, 80(5), 5900–5922. doi:10.1007/s11227-023-05680-8.

[21] Jagadeeswari, N., Mohanraj, V., Suresh, Y., & Senthilkumar, J. (2023). Optimization of virtual machines performance using fuzzy hashing and genetic algorithm-based memory deduplication of static pages. Automatika, 64(4), 868–877. doi:10.1080/00051144.2023.2223479.

[22] Suseno, T. R. D., Afrianto, I., & Atin, S. (2024). Strengthening data integrity in academic document recording with blockchain and InterPlanetary file system. International Journal of Electrical and Computer Engineering, 14(2), 1759–1769. doi:10.11591/ijece.v14i2.pp1759-1769.

[23] Liu, A., Chen, J., He, K., Du, R., Xu, J., Wu, C., Feng, Y., Li, T., & Ma, J. (2025). DynaShard: Secure and Adaptive Blockchain Sharding Protocol With Hybrid Consensus and Dynamic Shard Management. IEEE Internet of Things Journal, 12(5), 5462–5475. doi:10.1109/JIOT.2024.3490036.

[24] Muhammad, A., Shamshad, F., & Bae, S. H. (2023). Adversarial Attacks and Batch Normalization: A Batch Statistics Perspective. IEEE Access, 11, 96449–96459. doi:10.1109/ACCESS.2023.3250661.

[25] Adiban, M., Siniscalchi, S. M., & Salvi, G. (2023). A step-by-step training method for multi generator GANs with application to anomaly detection and cybersecurity. Neurocomputing, 537, 296–308. doi:10.1016/j.neucom.2023.03.056.

[26] Jiang, W., Wu, X., Song, M., Qin, J., & Jia, Z. (2023). A Scalable Byzantine Fault Tolerance Algorithm Based on a Tree Topology Network. IEEE Access, 11, 33509–33519. doi:10.1109/ACCESS.2023.3264011.

[27] Gurupriya, M., Dharshan, J. Y., Rohit, K. C., & Sridhar, S. K. (2025). Efficient Determent of Sybil Attacks in Blockchain. Proceedings of the International Conference on Multi-Agent Systems for Collaborative Intelligence, ICMSCI 2025, 2025, 268–272. doi:10.1109/ICMSCI62561.2025.10894178.

[28] Venkatesan, K., & Rahayu, S. B. (2024). Blockchain security enhancement: an approach towards hybrid consensus algorithms and machine learning techniques. Scientific Reports, 14(1). doi:10.1038/s41598-024-51578-7.

[29] Al-Marridi, A. Z., Mohamed, A., & Erbad, A. (2024). Optimized blockchain-based healthcare framework empowered by mixed multi-agent reinforcement learning. Journal of Network and Computer Applications, 224. doi:10.1016/j.jnca.2024.103834.

[30] Akrasi-Mensah, N. K., Agbemenu, A. S., Nunoo-Mensah, H., Tchao, E. T., Ahmed, A. R., Keelson, E., Sikora, A., Welte, D., & Kponyo, J. J. (2023). Adaptive Storage Optimization Scheme for Blockchain-IIoT Applications Using Deep Reinforcement Learning. IEEE Access, 11, 1372–1385. doi:10.1109/ACCESS.2022.3233474.

[31] Ipchi Sheshgelani, M., Pashazadeh, S., & Salehpoor, P. (2023). Cooperative hybrid consensus with function optimization for blockchain. Cluster Computing, 26(6), 3565–3576. doi:10.1007/s10586-022-03746-5.