Computer Networks 275 (2026) 111868

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

FBBTrust: Decentralized trust modeling in 6G via fuzzy inference, BiLSTM, and blockchain

Elmira Saeedi Taleghani, Ana Lucila Sandoval Orozco, Luis Javier García Villalba *

Group of Analysis, Security and Systems (GASS), Department of Software Engineering and Artificial Intelligence (DISIA), Faculty of Computer Science and Engineering, Universidad Complutense de Madrid (UCM), Office 431 Calle Profesor José García Santesmases 9 Ciudad Universitaria, Madrid, 28040, Spain

ARTICLE INFO

Keywords:
6G networks
Fuzzy logic
BiLSTM
Blockchain
Trust evaluation
Security

ABSTRACT

The dynamic and decentralized nature of 6G networks prioritizes trust management across three axes: uncertainty in noisy multimodal signals, temporal drift in behavior, and tamper-resistant provenance. We introduce **FBBTrust**, a framework that combines fuzzy inference to project QPC, QoS, and contextual cues into a calibrated prior; BiLSTM to learn temporal dependencies for the subsequent step of trust; and a lightweight blockchain for immutably persistent trust update in decentralized verification. On the *CIC-IoT2023* benchmark, FBBTrust outperforms two strong baselines (spatio-temporal trust and FL + GRU) in terms of MAE/MSE/RMSE and F1, and reduces execution time and energy. We report mean±95 % CI across five seeds and significance using paired tests; we also discuss privacy, scalability, and governance implications for deployment. The results indicate that FBBTrust is a promising direction for trustworthy orchestration in 6G-class systems. The results indicate that FBBTrust is a promising direction for trustworthy orchestration in 6G-class systems, particularly for trust-aware coordination in vehicular and industrial IoT environments. Our innovation lies in (i) a QPC/QoS/CI-tuned fuzzy prior, (ii) a BiLSTM predictor fused through simplex-constrained learned weights ($\alpha + \beta + \gamma = 1$), and (iii) privacy-preserving on-chain persistence (hash-only raw telemetry) with batch updates. To our knowledge, state-of-the-art fuzzy-ML-blockchain hybrids do not learn fusion weights under simplex constraints nor end-to-end prove provenance, runtime/energy, and statistical significance on CIC-IoT2023.

1. Introduction

The emergence of 6G networks is expected to revolutionize wireless communications by enabling highly decentralized and intelligent systems [1]. These networks will support massive-scale Internet of Things (IoT), autonomous systems, smart cities, and real-time AI-driven services [2]. Unlike previous generations, 6G relies on minimal centralized control and strong local decision-making, thus increasing susceptibility to security threats such as data tampering, Sybil attacks, black hole attacks, and node impersonation [3]. Under these circumstances, ensuring trust and security among network nodes becomes a critical challenge.

In 6G networks, trust evaluation serves as a fundamental mechanism for securing and stabilizing interactions among heterogeneous devices [4]. However, traditional trust assessment models are inadequate in these highly dynamic environments, primarily because of the following reasons:

1. **Dynamic network conditions**: Trust values shift over time due to node behavior changes, environmental factors, and attacker activities [5].
2. **Heterogeneous nodes**: Devices with diverse computational and communicational capabilities complicate the trust computation process [2].
3. **Inadequate centralized models**: Conventional, static, and centralized trust frameworks are unsuited to the decentralized and ever-evolving nature of 6G [6].

To address these limitations, the proposed methodology integrates three primary components:

1. **Fuzzy inference system (FIS)**: Deals with uncertainty by converting raw trust data into fuzzy linguistic variables, rule-based processing, and defuzzification to obtain an initial trust score.
2. **BiLSTM**: Learns temporal fluctuations in trust and predicts trust scores based on past trust data, making prediction more accurate in dynamic networks.
3. **Blockchain security mechanism**: Guarantees data integrity, immutability, and decentralized verification of trust scores by storing trust assessments securely via cryptographic hashing and consensus protocols.

* Corresponding author.
E-mail addresses: elmirasa@ucm.es (E. Saeedi Taleghani), asandov@ucm.es (A.L. Sandoval Orozco), javierv@ucm.es (L.J. García Villalba).

https://doi.org/10.1016/j.comnet.2025.111868
Received 6 April 2025; Received in revised form 13 October 2025; Accepted 11 November 2025
Available online 13 November 2025

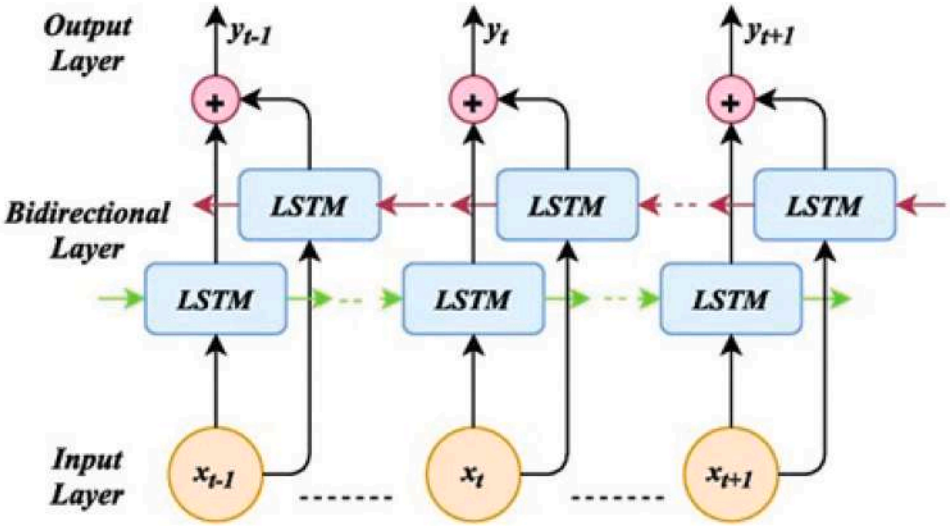


Fig. 1. BiLSTM architecture used in FBBTrust (2 × LSTM(64 units), dropout = 0.2, linear head).

All three components interact to provide a solid, dynamic, and secure trust assessment system applicable to 6G networks.

Fuzzy logic offers a robust means of dealing with the imprecision and vagueness inherent in trust calculations [5]. Rather than labeling trust as merely "high" or "low," fuzzy inference assigns continuous membership values based on metrics such as Quality of Peer-to-Peer Communication (QPC), Quality of Service (QoS), and Contextual Information (CI), resulting in a more realistic representation of node trustworthiness [7].

As trust evolves over time, sequential learning is crucial for accurately forecasting its future states. The Bi-LSTM or BiLSTM deep neural network is used to detect attacks in combination with the GMDH algorithm. A bidirectional LSTM (BiLSTM) is an extension of the LSTM model in which training is enhanced by traversing the input data twice, from left to right and right to left.

LSTM has been proposed to solve the RNN problem known as the "vanishing gradient". LSTM models extend the memory of an RNN to allow it to retain and learn the long-term dependencies of inputs. LSTM memory is used to store information over a long period of time and make decisions regarding retaining or ignoring information in memory. This allows us to capture the important features of the inputs and preserve this information over a long period of time [1,2].

In this study, BiLSTM was used to generate a neural network model and integrate it with GMDH, and finally detect attacks in HIS. Fig. 1 shows the structure of BiLSTM deep neural network.

The general steps of the BiLSTM algorithm are as follows:

1. **Bi-directional processing:** Unlike traditional RNNs that process input sequences in only one direction (forward or backward), Bi-LSTM processes sequences in both directions simultaneously. It consists of two LSTM layers: one processes the sequence in the forward direction, and the other in the backward direction. Each layer maintains its own hidden state and memory cells.
2. **Forward pass:** During the forward pass, the input sequence from the first step to the last step is fed into the forward LSTM layer. At each time step, the forward LSTM computes its hidden state and updates its memory cell based on the current input, previous hidden state, and memory cell.
3. **Backward motion:** Simultaneously, the input sequence is also entered into the backward LSTM layer in reverse order from the last time step to the first step. Similar to the forward pass, the backward LSTM computes its hidden state and updates its memory cell based on the current input, previous hidden state, and memory cell.
4. **Combining forward and backward states:** After completing the forward and backward passes, the hidden states from both LSTM layers were combined at each time step. This combination can be as simple as concatenating hidden states or applying other transformations.

The advantage of Bi-LSTM is that it captures not only the context preceding a certain time step (like traditional RNNs) but also subsequent contexts. By considering past and future information, Bi-LSTM can capture richer dependencies in the input sequence. Therefore, the data related to the attacks (training data) were entered into the BiLSTM model, and a model was produced.

Blockchain is a form of Distributed Ledger Technologies (DLT) that serves as a distributed and immutable ledger for recording transactions and tracking assets among various entities [8]. It offers many desirable benefits for data sharing and exchange, including transparency, integrity, security, and traceability.

- **Transparency:** As long as the data are on chain, all participants can share the same data as opposed to an individual copy [9].
- **Security and integrity:** Before transaction can be placed on the chain, all participants must agree. When approval is obtained, the record is encrypted and linked to previous records. Changing the status or value of a single record requires modifying all subsequent records, which is almost impossible in large-scale blockchain networks [9].
- **Traceability:** When a transaction is put on chain, an audit is created to record where the transaction comes from and who makes it [9].

In adversarial 6G contexts, trust data are susceptible to malicious manipulation [3]. Blockchain secures these records through its decentralized architecture, immutability, and transparency [4]. This approach:

- Decentralizes Trust Management and reduces single points of failure.
- Ensures Immutable Records and prevents tampering with trust scores [6].
- Automate updates via smart contracts, streamlining the trust evaluation process [10].

By combining fuzzy logic, BiLSTM, and blockchain, the proposed framework offers a dynamic, secure, and context-aware trust-management solution that is suited to the complexities of 6G networks. The Contributions are:

1. Unified triad: Concurrent handling of uncertainty (fuzzy), time drift (BiLSTM), and tamper-resistance (blockchain).

Table 1
Acronyms & symbols.

Symbol	Meaning
QPC	Quality of Peer-to-Peer Communication
QoS	Quality of Service
CI	Contextual Information
$T_f(t)$	Fuzzy trust score at time t
$\hat{T}(t+1)$	BiLSTM predicted trust for time t+1
$T_{bc}(t)$	Last blockchain-confirmed trust at time t
$T_{final}(t)$	Final fused trust score
α, β, γ	Fusion weights (non-negative, sum to 1)
τ	Decision threshold for trust labeling

2. Principled fusion: Learning of weights under simplex constraints regularizes validation loss; sensitivity analysis option available.
3. Transparent persistence: Smart-contract managed on-chain recording of trust with off-chain raw data (privacy-preserving audit trail).
4. Ablation & significance: Module-by-module ablation, confidence intervals, and paired significance tests showcase the utility of each module.
5. Operational metrics: By sample and by on-chain update runtime/energy to facilitate deployment sizing.

Trust in 6G must be computed under three concurrent pressures: (1) uncertainty in noisy, heterogeneous signals (QPC/QoS/context), (2) temporal drift in node behavior, and (3) tamper-resistant provenance for enforcement and auditing. Existing approaches typically cover at most two of these axes, for example, fuzzy scoring without temporal forecasting, recurrent models without calibrated priors, or blockchain logs without learning quality, and seldom provide a statistically rigorous evaluation with operational costs. We target a single deployable pipeline that jointly addresses all three while exposing tunable trade-offs (reactivity/stability/provenance).

The remainder of this paper is organized as follows. Section 2 covers related work and positions *FBBTrust*, with a comparative trustscape of trust models (Section 2.5) and a brief advanced mathematical background (Section 2.6). Section 3 formalizes the approach, that is, fuzzy inference, BiLSTM temporal modeling, and the combination rule, with post-equation intuitions and dummyTXdummy– a constrained weight-learning approach (Section 3.5.1). Section 4 outlines the implementation and experimental design, including datasets and preprocessing, BiLSTM architecture and training hyperparameters (Table 3), fuzzy rules, blockchain persistence, runtime/energy protocol, statistical tests, ablations, and threshold tuning (Sections 4.1–4.10). Section 5 presents the results: the overall performance against the baselines with confidence intervals (Fig. 4, Table 5), and temporal dynamics (Figs. 5–7), and distributional analyses (Figs. 8, 9), ablation results (Table 6), and efficiency/on-chain overhead (Table 7). Section 6 provides a more general discussion (why/when *FBBTrust* is useful, operational implications, and limitations). Section 7 discusses the ethics, privacy, and governance. Section 8 concludes with future directions, including federated extensions, DRL-supported reweighting, scalable logging (rollup/sharding), and cross-domain evaluation (Table 1).

2. Related works

In 6G networks, trust evaluation has gained considerable importance, and models following fuzzy logic, machine learning, and even the application of blockchain technologies have been developed.

2.1. Fuzzy logic-based trust models

Fuzzy logic is a core paradigm for handling uncertainty in distributed trust evaluation. Hashemi and Aliee [5] designed a fuzzy dynamic routing protocol for IoT to improve decision-making in ambiguous environments, whereas Alsaqour et al. [11] used fuzzy logic for adaptive packet beaconing in MANETs. More recent studies have focused on multidimensional aggregation and context awareness. Saeedi et al. [12] introduced a fuzzy trust management framework for the IoT with a comprehensive taxonomy and performance assessment. Jayakumar et al. [13] extended fuzzy trust modeling to the social IoT, emphasizing edge-centric data. Mahmood et al. [14] proposed a fuzzy adaptive mechanism for intelligent transportation systems, whereas Ren and Qin [15] combined QoS metrics into a fog-based multidimensional fusion model. These studies established the interpretability of fuzzy logic but relied on static data and manually defined rules, which limited adaptability under fast-changing network dynamics. Hybrid fuzzy-ML approaches have also been developed. Mehjabin et al. [16] proposed PETIT, a Physical Unclonable Function (PUF)-enabled fuzzy and machine-learning trust framework to ensure resilience against spoofing. Although robust, the model incurs computational overhead in real-time scenarios. Broader surveys on 6G security and trust management [17,18] recommend uncertainty-aware scoring (e.g., fuzzy priors) but rarely integrate such calibration with sequential learning or blockchain-based persistence methods. *FBBTrust* directly addresses these limitations by combining fuzzy calibration, temporal forecasting, and immutable provenance.

2.2. Machine learning-based trust prediction

Machine learning techniques have been extensively explored for trust prediction by learning patterns from historical data. Shameem et al. [1] proposed an e-learning-based trust predictor for 6G networks using past observations, and Mourad et al. [19] employed ML classifiers for cyber-attack detection within cyber-physical systems. Deep architectures for trust estimation include Nazir et al. [20], who applied deep learning to the IoT, and Luecking et al. [21], who focused on identity-based trust evaluation. Kianpisheh et al. [22] integrated federated learning with gated recurrent units (GRU) to enable decentralized, adaptive trust prediction across heterogeneous nodes, highlighting scalability but still lacking interpretable priors or verifiable outputs.

Zhang et al. [23] combined deep reinforcement learning (DRL) with blockchain sharing to optimize secure resource allocation, improving adaptability at the cost of complexity and data requirements. Huang [24] summarized emerging ML-driven trust models for IoT but noted that most overlook interpretability and sequential calibration. Recent surveys [25,26] further analyze data poisoning, backdoors, and non-IID issues in FL pipelines, along with mitigation strategies such as clipping, noise addition, and robust aggregation. *FBBTrust* complements these efforts by jointly learning temporal trust trajectories while maintaining explainability and on-chain traceability.

2.3. Security models with blockchain integration

Blockchain technology has been widely adopted to enhance trust integrity and auditability. Putra et al. [4] and Zhaofeng et al. [27] demonstrated blockchain-based trust management for data integrity in 6G edge computing. Li et al. [28] provided a comprehensive survey of blockchain trust frameworks across IoT, whereas Yin et al. [29] emphasized transparency and immutability. Tu et al. [30] proposed a dynamic blockchain-ML trust model capable of real-time adjustment of trust scores. Similarly, Alshahrani et al. [31] and Hbaieb et al. [32] examined vehicular trust management and the role of blockchain in mitigating security threats under mobility conditions.

Complementary privacy-preserving approaches include blockchain-assisted cloud auditing and zero-knowledge authentication [33–35]. Most retain only hashed summaries and trust values on-chain, keeping raw telemetry off-chain for privacy–an approach also adopted by *FBBTrust*. Despite these advances, many blockchain-based trust models focus primarily on storage and provenance, with limited coupling to adaptive or uncertainty-aware trust reasoning.

2.4. Hybrid trust models integrating spatial-temporal learning

The integration of machine learning with temporal and spatial analysis techniques is a promising approach to trust prediction. These models use clustering algorithms and time-series forecasting to model dynamic trust relationships in networks.

The spatio-temporal trust model [36] estimates trust scores by combining clustering techniques, such as K-Means, with the calculation of moving averages. The proposed model enables real-time computation of trust in dynamic network conditions through spatial and temporal variations in trust, thereby overcoming a major weakness of static trust models.

2.5. Limitations of existing approaches

Despite significant advances in trust evaluation, the following critical challenges continue to affect existing models:

1. **Static trust evaluation:** Most of the approaches do not consider temporal fluctuations in trust, resulting in outdated or incorrect trust evaluations.
2. **Lack of integration:** In addition, a few models integrate fuzzy logic, machine learning, and blockchain into one cohesive trust evaluation framework.
3. **Security vulnerabilities:** Most trust models are not equipped with robust adversarial defense mechanisms and hence are prone to manipulation.

2.6. Advanced mathematical context

New mathematics in orthogonal polynomials and moment theory form the basis for robust feature aggregation under noise and discretization. The robust computation of higher-order Hahn (and dual Hahn) polynomials enables robust moment recovery at large orders (Daoui et al. [37]; Daoui et al. [38]). Parameterized families, such as Meixner, can be tuned with metaheuristics, such as Firefly, to obtain data-adaptive bases (Bencherqui et al. [39]). Metaheuristics in optimization with chaos improve exploration–exploitation trade-offs (El Ghouate et al.[40]). DNA-and-chaos image security systems illustrate lightweight primitives that are realizable for privacy-preserving logging (Wang et al. [41]). We call these context anchors—not bound to duplicate *FBBTrust*, but motivating future improvements to membership design, fusion regularization, and hyperparameter search.

2.7. Our contribution

To fill these gaps, we propose an integrated fuzzy blockchain model enhanced with BiLSTM. In this model:

- **Fuzzy Logic:** Handles uncertainty in the estimation of trust.
- **Bi-LSTM:** which will detect and predict variations in temporal trust dynamics.
- **Blockchain technology:** secures trust evaluations against tampering or unauthorized modifications.

2.8. Positioning and novelty vs. closest hybrids

The closest hybrids in the literature combine fuzzy rules with recurrent models and, occasionally, a blockchain logger. However, they typically (i) fix or hand-tune fusion weights, (ii) record raw or verbose payloads on-chain, and/or (iii) omit statistical significance and operational costs from their analyses. **FBBTrust** differs in three ways: (1) simplex-constrained learning of (α, β, γ) that provably enforces calibrated trade-offs between reactivity (fuzzy), stability (BiLSTM), and provenance (blockchain); (2) hash-only on-chain records with batched writes, preserving auditability with bounded cost and leakage; and (3) rigorous, deployment-oriented evaluation (CIs, paired tests, runtime/energy/gas). Table 2 clarifies where FBBTrust extends rather than copying previous hybrids.

This integrated framework is of paramount importance when considering future 6G scenarios. Indeed, as noted by Liu et al. [6], the combination of blockchain with AI in distributed edge computing forms a convincing approach to active trust management. Our approach contributes to a dynamic, secure, and accurate trust evaluation mechanism for 6G networks.

To complement related research with a networking focus, we highlight the proximate mathematical threads that affect robustness and efficiency (see Table 2). Orthogonal polynomial systems and discrete moment theory (e.g., Meixner and Kimura families) yield numerically stable bases and compact descriptors; these ideas can be used to create smoother, well-conditioned fuzzy membership functions and regularize blending weights. Metaheuristic optimization of moment characteristics (e.g., Firefly optimized Meixner moments) and chaos-enhanced optimizers (e.g., Kepler optimization with chaotic maps) suggest alternative methods for optimizing fuzzy rules and weights (α, β, γ) in non-convex objectives. DNA coding in combination with chaotic maps shows how algebraic coding with chaos adds confusion/diffusion, which is parallel to the use of blockchain immutability combined with off-chain privacy techniques. Finally, transform-moment representations within higher-dimensional algebras (e.g., octonion-based transforms) illustrate how structured elements can extend temporal learners, such as BiLSTM. We cite them as context anchors; they do not have to reimplement *FBBTrust* but motivate future improvements in membership design, fusion regularization, and hyperparameter search.

3. Proposed methodology

This section describes our fuzzy blockchain BiLSTM model in five steps. Each step is designed to capture different aspects of trust assessment, ranging from initial data collection to handling uncertainties, modeling temporal behavior, securing trust records, and finally, making a comprehensive trust decision. This methodology integrates both analytical and computational perspectives, with particular emphasis on mathematical formulation to ensure clarity and reproducibility.

Fig. 2 shows the five-step Fuzzy-Blockchain BiLSTM approach. These steps ensure that:

- **Step 1**: captures a comprehensive view of node behavior through multi-dimensional data. Algorithm 1 shows that it involves loading and preprocessing the trust data.
- **Step 2**: addresses uncertainty and provides a robust fuzzy trust measure. Algorithm 2 explains the computation of fuzzy trust scores using weighted metrics.
- **Step 3**: accounts for the temporal dynamics, ensuring that trust predictions are updated based on historical trends. Algorithm 3 focuses on temporal trust prediction using BiLSTM.
- **Step 4**: secures each trust update against manipulation by leveraging the blockchain immutability and smart contracts. Algorithm 4 securely stores the trust data in the blockchain.
- **Step 5**: integrates all the components into a unified trust decision optimized via appropriate weighting strategies. Algorithm 5 computes the final trust score and makes trust decisions based on the threshold.

3.1. Step 1: Input trust data

The first step involves collecting and preprocessing trust data from both direct and indirect observations in a 6G network environment. We consider three categories of metrics in Eqs. (1)–(3):

1. **Quality of Peer-to-Peer Communication (QPC)**:

$$QPC = \{R_s, P_l\} \tag{1}$$

Table 2
Summary of representative trust models versus *FBBTrust*.

Model	Input QPC	QoS	CI	Temp.	Decent.	Adver.	Main limitation
Fuzzy-only (classic)	✓	✓	partial	×	×	low	Static, no drift handling
ML-only (e.g., GRU)	✓	✓	✓	✓	×	medium	No provenance; threshold tuning
Blockchain-only	–	–	meta	×	✓	medium	No modeling, storage overhead
Spatio-temporal (ST-Trust)	✓	✓	partial	✓ (mov. avg)	×	medium	Limited context & smoothing
FBBTrust (ours)	✓	✓	✓	✓ (BiLSTM)	✓ (on-chain)	high	On-chain overhead (mitigations in Section 6)

Abbreviations: QPC = Quality of Peer-to-Peer Communication; QoS = Quality of Service; CI = Contextual Information; Temp. = Temporal modeling; Decent. = Decentralized tamper-resistance; Adver. = Adversarial robustness.
Symbols: ✓ = present/supported; × = absent/not supported; partial = partially supported; mov. avg = moving average; — = not applicable / not used; on-chain = recorded on blockchain.
Note: For *FBBTrust*, temporal modeling uses BiLSTM and decentralized tamper-resistance is implemented on-chain; mitigation strategies for overhead are discussed in Section 6.

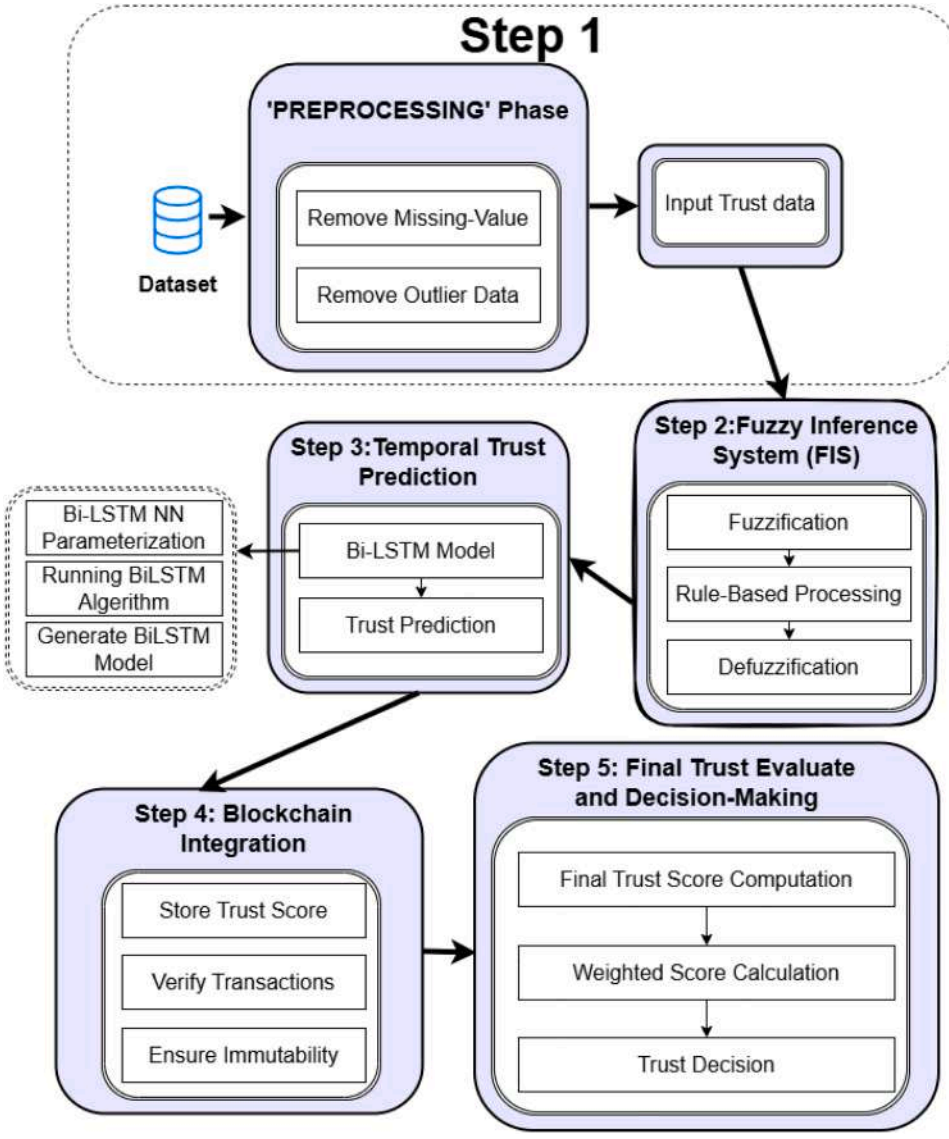


Fig. 2. System architecture: FIS →sequence builder →BiLSTM →fusion →on chain logger; consensus/validation loop shown.

where R_s represents the ratio of successful packet deliveries (e.g., successfully acknowledged interactions over total sent), and P_l denotes packet loss rate.

2. **Quality of Service (QoS)**:

$$QoS = \left\{L, B, R_s^Q\right\} \tag{2}$$

where L is the average latency (in milliseconds), B is the available bandwidth (in Mbps), and R_s^Q could represent reliability in terms of service uptime or error-free transmission probability.

3. **Contextual Information (CI)**:

$$CI = \{M, Ea, Ec\} \tag{3}$$

Algorithm 1 Step 1: Input trust data.

```
function LOAD_TRUST_DATA(file_path)
    Load dataset from file_path
    Select relevant features (QPC, QoS, CI)
    Handle missing values (replace NaN with mean)
    Replace infinite values with max valid value
    Normalize data using MinMaxScaler
    return normalized dataset
end function
```

Algorithm 2 Step 2: Compute fuzzy trust score.

```
function COMPUTE_FUZZY_TRUST(data)
    DEFINE weights w1, w2, w3
    for each data sample do
        Compute fuzzy trust score: T_fuzzy = (w1 × QPC) + (w2 × QoS) + (w3 × CI)
    end for
    return fuzzy trust scores
end function
```

Algorithm 3 Step 3: BiLSTM for temporal trust prediction.

```
function CREATE_SEQUENCES(data, seq_len)
    X, Y ← ∅
    for i = 0 to len(data) − seq_len do
        X.append(data[i : i + seq_len])
        Y.append(data[i + seq_len])
    end for
    return X, Y
end function

function TRAIN_BILSTM(X_train, Y_train, X_test, Y_test)
    DEFINE BiLSTM model with:
      Bidirectional LSTM layer
      Dense output layer (linear activation)
      Adam optimizer
      MSE loss function
    Train model using early stopping
    return trained model
end function
function PREDICT_TRUST_BILSTM(model, X_test)
    return model.predict(X_test)
end function
```

where M indicates node mobility (e.g., speed or mobility index), Ea represents available energy, and Ec captures external environmental conditions (e.g., interference level).

Let $T_i(t)$ denote the raw trust feature vector collected at time t for node i in Eq. (4):

$$T_i(t) = \left[R_s(t), Pl(t), L(t), B(t), R_sQ(t), M(t), Ea(t), Ec(t)\right]^T \tag{4}$$

In practice, one may aggregate multiple observations over a time window Δ_t, or normalize each metric into the interval [0,1] for uniform representation. For instance, one could define in Eqs. (5)–(6):

$$\tilde{R}_s(t) = \frac{R_s(t)}{\max_{\tau\in[t-\Delta_t,t]} R_s(\tau)} \tag{5}$$

$$\tilde{L}(t) = \frac{L_{\text{max-L}}(t)}{L_{\max}} \tag{6}$$

where $L_{\max}$ is an upper bound on permissible latency in the network context.

Algorithm 4 Step 4: Blockchain trust storage.

```
Class Block
function _INIT_(index, timestamp, data, previous_hash)
    self.index = index
    self.timestamp = timestamp
    self.data = data
    self.previous_hash = previous_hash
    self.hash = self.calculate_hash()
end function

function CALCULATE_HASH
    return SHA-256 hash of (index, timestamp, data, previous_hash)
end function

Class Blockchain
function _INIT_
    Initialize blockchain with Genesis block
end function

function GET_LATEST_BLOCK
    return last block in blockchain
end function

function ADD_BLOCK(new_block)
    new_block.previous_hash = get_latest_block().hash
    new_block.hash = new_block.calculate_hash()
    Append new_block to blockchain
end function

function STORE_TRUST_IN_BLOCKCHAIN(blockchain, trust_scores)
    for each trust score in trust_scores do
        Create new block with trust score
        Add block to blockchain
    end for
    return blockchain
end function
```

3.2. Step 2: Fuzzy inference system (FIS)

Given the inherent uncertainty and imprecision in trust metrics, we employ a Fuzzy Inference System (FIS) to convert raw data into a preliminary trust score. This step is based on the framework proposed by [5], comprising three major phases: Fuzzification, Rule-Based Processing, and Defuzzification.

3.2.1. Fuzzification

We map each numerical trust metric (e.g., $\tilde{R}_s(t)$, $\tilde{L}(t)$, P_l, etc.) into linguistic variables such as Low, Medium, and High. Suppose we define triangular membership functions for each variable. For example, for a generic metric $x \in [0,1]$, the membership functions $\mu_{\text{Low}}(x)$, $\mu_{\text{Medium}}(x)$, and $\mu_{\text{High}}(x)$ can be defined as in Eqs. (7)–(9):

- Low:

$$\mu_{\text{Low}}(x) = \begin{cases} 1-2x & \text{if } 0 \le x \le 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

- Medium:

$$\mu_{\text{Medium}}(x) = \begin{cases} 2x & \text{if } 0 \le x \le 0.5 \\ 2(1-x) & \text{if } 0.5 \le x \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

- High:

$$\mu_{\text{High}}(x) = \begin{cases} 2x-1 & \text{if } 0.5 \le x \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Algorithm 5 Step 5: Compute final trust score.

```
function COMPUTE_FINAL_TRUST(T_fuzzy, T_BiLSTM, T_blockchain, α, β, γ)
    T_final = α × T_fuzzy + β × T_BiLSTM + γ × T_blockchain
    return T_final
end function

function TRUST_DECISION(T_final, threshold)
    if T_final ≥ threshold then
        return "Trust Node"
    else
        return "Do Not Trust Node"
    end if
end function

Main execution
file_path ← εDDoS − ACK_Fragmentation.pcap.csvε
data ← Load_Trust_Data(file_path)
T_fuzzy ← Compute_Fuzzy_Trust(data)
X_seq, Y_seq ← Create_Sequences(T_fuzzy)
X_train, X_test, Y_train, Y_test ← Train_Test_Split(X_seq, Y_seq)
BiLSTM_model ← Train_BiLSTM(X_train, Y_train, X_test, Y_test)
T_BiLSTM ← Predict_Trust_BiLSTM(BiLSTM_model, X_test)
blockchain ← Blockchain()
blockchain ← Store_Trust_In_Blockchain(blockchain, T_fuzzy[−10 : ])
α, β, γ ← 0.4, 0.4, 0.2
T_final ← Compute_Final_Trust(T_fuzzy, T_BiLSTM, T_fuzzy[−10 : ], α, β, γ)
for i = 0 to len(T_final) do
    decision ← Trust_Decision(T_final[i], threshold = 0.5)
    PRINT("Node", i, "Decision:", decision)
end for
```

These are illustrative examples; Gaussian, trapezoidal, or other membership functions could be employed depending on the network's empirical data distribution.

3.2.2. Rule-based processing

Let us denote the fuzzified inputs as in Eq. (10):

$$\{F_{\text{QPC}}(t), F_{\text{QoS}}(t), F_{\text{CI}}(t)\} \tag{10}$$

where each F is itself a linguistic set (e.g., Low, Medium, High) derived from the membership functions. We then define a set of experts or learned if-then rules. An example rule could be:

If (F_{QPC} is Low) AND (F_{QoS} is High) then (Trust is Medium).

Each rule r has the following approximate structure in Eq. (11):

$$r: \text{IF } F_{\text{QPC}} = A \text{ AND } F_{\text{QoS}} = B \text{ AND } F_{\text{CI}} = C \text{ THEN } F_{\text{Trust}} = D \tag{11}$$

where A, B, C, D are linguistic variables in $\{\text{Low}, \text{Medium}, \text{High}\}$. The output membership degree $\mu_{F_{\text{Trust}}}(D)$ can be combined using a T-norm (e.g., minimum or product) to yield an overall aggregated membership.

3.2.3. Defuzzification

Finally, the aggregated fuzzy set for Trust is mapped back to a numeric value $T_{\text{fuzzy}}(t)$. A common method is centroid defuzzification in Eq. (12):

$$T_{\text{fuzzy}}(t) = \frac{\int_{u\in U}^{1} u\mu_{F_{\text{trust}}}(u)\,du}{\int_{u\in U}^{1} \mu_{F_{\text{trust}}}(u)\,du} \tag{12}$$

where U is the universal space of trust (often normalized to [0,1]), and $\mu_{F_{\text{trust}}}(u)$ is the membership function of the aggregated fuzzy set. The result is a crisp trust score that still captures the underlying uncertainty.

3.3. Step 3: Temporal trust analysis using BiLSTM

To capture the dynamic nature of trust over time, we use a Bidirectional Long Short-Term Memory (BiLSTM) model. The historical sequence of trust values $\{T_{\text{fuzzy}}(1), T_{\text{fuzzy}}(2), \ldots, T_{\text{fuzzy}}(t)\}$ is fed into the BiLSTM to forecast future trust scores.

3.3.1. BiLSTM architecture

A conventional LSTM cell can be summarized by the following gating mechanisms (for a forward pass) in Eqs. (13)–(18):

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{13}$$

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{14}$$

$$\tilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{15}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{C}_t \tag{16}$$

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{17}$$

$$h_t = o_t \odot \tanh(c_t) \tag{18}$$

where x_t is the input at time t, h_t is the hidden state, c_t is the cell state, and σ is the sigmoid activation function.

BiLSTM processes the sequence in both forward and backward directions in Eqs. (19)–(20):

$$\hat{h}_t = \text{LSTM}_{\text{forward}}(x_t, \hat{h}_{t-1}) \tag{19}$$

$$\check{h}_t = \text{LSTM}_{\text{backward}}(x_t, \check{h}_{t+1}) \tag{20}$$

yielding a combined hidden state in Eq. (21):

$$h_t = [\hat{h}_t; \check{h}_t] \tag{21}$$

3.3.2. Trust forecasting

The input at time t is $x_t = T_{\text{fuzzy}}(t)$, and the output $\hat{T}_{\text{BiLSTM}}(t+1)$ represents the predicted trust score at time $t+1$. In vectorized form, we can consider a sliding window approach of length L in Eq. (22):

$$x_t = [T_{\text{fuzzy}}(t-L+1), \ldots, T_{\text{fuzzy}}(t)], \tag{22}$$

which is fed into BiLSTM to produce a forecast in Eq. (23):

$$\hat{T}_{\text{BiLSTM}}(t+1) = \Phi(x_t; \Theta) \tag{23}$$

where $\Phi(\cdot)$ denotes the BiLSTM function parameterized by Θ (weights $\{W_f, W_i, W_o, W_C, b_f, b_i, b_o, b_C\}$ in both directions).

3.3.3. Training objective

We train the BiLSTM by minimizing a loss function τ, often the Mean Squared Error (MSE) in Eq. (24):

$$\tau(\Theta) = \frac{1}{N}\sum_{t=1}^{N}\left(T_{\text{fuzzy}}(t+1) - \hat{T}_{\text{BiLSTM}}(t+1)\right)^2 \tag{24}$$

Through gradient-based optimization (e.g., Adam), the model learns to predict future trust values effectively.

3.4. Step 4: Blockchain integration

Despite accurate predictions, trust scores can still be altered or falsified by malicious nodes unless securely stored. We integrate blockchain technology to protect trust records:

3.4.1. Transaction creation

Each node i periodically publishes its updated trust score $\hat{T}_i(t)$ (coming from either fuzzy, BiLSTM, or combined methods) as a transaction in the blockchain network.

3.4.2. Smart contracts

A smart contract automatically validates trust updates. For instance in Eq. (25):

$$\text{require}(\hat{T}_i(t) \in [0,1]) \tag{25}$$

This ensures that no malicious node can post an out-of-range trust value.

3.4.3. Consensus mechanism

A suitable algorithm, such as Proof-of-Stake (PoS) or Delegated PoS, confirms each block. Let B_k be the kth block containing all trust records $\{\hat{T}_1(t), \hat{T}_2(t), \dots\}$. A simplified model of PoS could be in Eq. (26):

$$\text{ValidationSelection} \sim \frac{\text{Stake}_v}{\sum_j \text{Stake}_j} \tag{26}$$

where Stake_v is the stake of validator v. Once B_k is validated, its hash $H(B_k)$ and the pointer to the previous block $H(B_{k-1})$ establish an immutable chain of trust records.

3.4.4. Immutability & verification

Each node can verify the chain to ensure no tampering has occurred in Eq. (27):

$$H(B_k) = \text{Hash}(B_k \parallel H(B_{k-1})) \tag{27}$$

Any modification to B_k or B_{k-1} invalidates the hash, thus detecting attempts at manipulation.

Through this blockchain layer, the system guarantees data integrity, decentralized trust management, and transparency when updating or referencing trust values.

3.5. Step 5: Final trust decision

Having obtained three main sources of trust information (i) fuzzy score, (ii) BiLSTM prediction, and (iii) blockchain-secured trust record, we derive a comprehensive final trust score in Eq. (28):

$$T_{\text{final}}(t) = \alpha T_{\text{fuzzy}}(t) + \beta \hat{T}_{\text{BiLSTM}} + \gamma T_{\text{blockchain}}(t) \tag{28}$$

where α, β, and γ are weighting factors that reflect the relative importance of each component ($\alpha + \beta + \gamma = 1$). The variable $T_{\text{blockchain}}(t)$ may be the previously validated trust score on the blockchain (that is, the last immutable state of trust known to the network).

Eq. (28) fuses three complementary signals: $T_f(T_{fuzzy})$ provides instantaneous, rule-based reactivity; $\hat{T}$ contributes temporally smoothed forecasts from past behavior; and $T_{bc}(T_{blockchain})$ anchors the estimate to the last immutable on-chain state. Enforcing $\alpha + \beta + \gamma = 1$ trades off reactivity (larger α) versus stability and provenance (larger β, γ).

3.5.1. Weight optimization

In many scenarios, α, β, and γ can be learned or optimized offline using a validation dataset with known ground-truth trust labels. One approach is to solve the constrained optimization in Eq. (29):

$$\min_{\alpha,\beta,\gamma} \frac{1}{N} \sum_{t=1}^{N} \left(y(t) - T_{\text{final}}(t)\right)^2 \tag{29}$$

subject to $\alpha + \beta + \gamma = 1$, $\alpha, \beta, \gamma \geq 0$

Where $y(t)$ is the trust value of the ground truth at time t. By using methods such as Lagrange multipliers or gradient descent with a simplex constraint, we can find the optimal combination of weights that minimizes the prediction error.

Eq. (29) learns (α, β, γ) by minimizing the squared deviation between T_{final} and the ground truth, under a simplex constraint. Geometrically, it selects the point on the probability simplex that best aligns the three sources with labeled trust, yielding principled fusion rather than ad-hoc weighting.

3.5.2. Unified score interpretation

Once $T_{\text{final}}(t)$ is computed, it can be interpreted or thresholded to make decisions about whether to trust a specific node i at a time t. For instance, a threshold $\tau \in [0,1]$ could be defined as in Eq. (30):

$$\text{Decision}(t) = \begin{cases} \text{Trust Node}, & \text{if } T_{\text{final}}(t) \geq \tau \\ \text{Do not trust Node}, & \text{if } T_{\text{final}}(t) < \tau \end{cases} \tag{30}$$

Depending on the network security requirements, the threshold τ could be static or dynamically updated based on the adversarial behavior observed.

This methodology is particularly suitable for 6G networks, where high mobility, diverse device capabilities, and potential adversarial conditions require an adaptive, resilient, and intelligent trust management framework.

3.5.3. Why simplex-constrained fusion is new/useful

Learning (α, β, γ) on the 2-simplex $(\alpha, \beta, \gamma \geq 0, \alpha + \beta + \gamma = 1)$ ensures that the final trust score is a convex combination of well-interpreted sources, preventing negative or > 1 weights and avoiding over-dominance. Unlike fixed/equal weightings, constrained least-squares yields data-calibrated trade-offs and, empirically (Table 4), improves MAE/RMSE/F1 within the error bars across seeds. Geometrically, the optimizer selects the point in the simplex that minimizes the validation error, which also simplifies the sensitivity analysis and online re-weighting.

4. Experimental evaluation

4.1. Hardware & software

- CPU / GPU / RAM / OS: [e.g., 12-core CPU; 1 × NVIDIA GPU; 32-64 GB RAM; Ubuntu 22.04]
- Python 3.11; TensorFlow/Keras 2.x (or PyTorch 2.x), scikit-learn 1.5, NumPy 1.26, Pandas 2.2.
- Blockchain client: private PoS/PoA testnet (e.g., geth), block time ~2 s, gas limit tuned for batched writes.

4.2. Dataset

The local dataset used in this study had 25,090 samples and 39 features. Train/Val/Test split without leakage: 70 % / 10 % / 20 % (17,563 / 2509 / 5018 instances). Splitting is time/scenario-aware to mimic the deployment.

Mapping to trust categories:

- QPC: ack_flag_number, syn_flag_number, rst_flag_number, psh_flag_number, fin_flag_number, ece_flag_number, cwr_flag_number.
- QoS: Time_To_Live, Rate, IAT.
- CI: Protocol Type, Tot size, Number, Variance, AVG, Std, Min, Max, plus available protocol indicators (e.g., ARP, ICMP, IGMP, IPv, LLC, DNS, DHCP, TCP, UDP, HTTP, HTTPS, Telnet, SMTP, SSH).

Descriptive statistics (mean/median/5th/95th percentiles) for each group are summarized in three supplemental tables titled “QPC feature summary”, “QoS feature summary”, and “CI feature summary”.

4.3. Pre-processing

We trained all the transformers on the training split only and applied the trained transformers to the validation/test to avoid leakage. Missing values were imputed with the column mean, $\pm\infty$ values were replaced with the largest finite value encountered during training. All numeric features were Min-Max scaled to [0,1] using parameters trained on the training data. Protocol indicator columns (e.g., ARP, ICMP, DNS, TCP/UDP, etc.) were already binary and were left as-is. For sequence

Table 3
Hyperparameters.

Item	Value
Sequence length L	32
Hidden units (per direction)	64
Dropout	0.2
Dense head	1 (linear)
Optimizer / LR	Adam / 1e–3
Batch / Epochs / Patience	32 / 50 / 7
Seeds (runs)	{42, 202, 777, 1337, 2025}

building, we used time-aware shuffling (local order preserved) and built sliding windows of size $L = 32$ (stride = 1). To deal with the class imbalance affecting the trust label used for τ tuning, we could choose to perform class weighting at the threshold selection time; model weights did not get reweighted except as noted otherwise.

4.4. BiLSTM architecture & training

Architecture. A bidirectional LSTM with 64 units in each direction (128 concatenated), dropout = 0.2 on the recurrent output, and conditioned on a 1-unit linear head. Sequence length $L = 32$; stride = 1.

Optimization. Adam (lr = 1e-3, $\beta_1 = 0.9, \beta_2 = 0.999$), batch size = 32. Early stopping monitored validation loss with patience = 7 and min_delta = 1e-4; we kept the best checkpoint (lowest val loss).

Stability. We trained 5 seeds {42, 202, 777, 1337, 2025} and report mean±95 % CI.

Fusion weights. (α, β, γ) were calibrated on the validation set using constrained least-squares with projection onto the 2-simplex ($\alpha, \beta, \gamma \geq 0; \alpha + \beta + \gamma = 1$). The sensitivity sweep across the simplex is shown in Table 4.

All the hyperparameters are listed in Table 3.

4.5. Fuzzy inference layer

Each is assigned three triangular memberships (low/medium/high). As a sample, a normalized metric $x \in [0, 1]$ utilized Low = [0,0,0.5], Medium = [0.25,0.5,0.75], High = [0.5,1,1]; end breakpoints were set at validation. We utilized 27 rules ($3 \times 3 \times 3$) based on QPC, QoS, and CI; T-norm = product, aggregation = max, and centroid defuzzification to obtain the crisp T_f. Example rule: IF (QPC is Low) AND (QoS is High) AND (CI is Medium) THEN (Trust is Medium).

4.6. Blockchain persistence layer

We stored settled trust updates on a private PoS/PoA test network for tamper-evident provenance. The updates were batched every 5 s. The on-chain record maintains (i) a SHA-256 digest of the off-chain raw telemetry (hash_of_raw) and (ii) the new $trust\_value \in [0, 1]$; raw data remain off-chain for privacy. The smart contract enforces range checks (requires $0 \leq trust\_value \leq 1$) and writes an event per update. Validator set size and block parameters are tuned to a target block time of ∼2 s. Per 1k updates, the gas consumption is reported in Table 7.

4.7. Runtime, energy, and scale

We present the wall-clock runtime (s) and energy (J) per 10k samples for each method and gas per 1k on-chain updates for FBBTrust. The runtime was measured using high-resolution timers (repeat = 5). Where feasible, device power was sampled at 1-2 Hz (e.g., nvidia-smi or OS counters) and summed (trapezoidal rule) to obtain Joules; otherwise, we specified time-normalized estimates and noted the limitation. We specify the model size (parameters) and an order-of-magnitude FLOPs estimate for a forward pass at $L = 32$. The simulation scale, that is, the number of nodes and trust updates/s, was specified along with the measurements. Table 7 lists the measured values.

4.8. Evaluation protocol & statistical significance

Regression. MAE, MSE, RMSE, R^2 as mean±95 % CI over five seeds. CIs were approximated by bootstrap with 10,000 resamples (random seed = 123).

Pairwise tests. We compared the per-sample absolute errors of FBBTrust with all baselines using a two-sided paired t-test.

Classification view. For tuned τ, we provide Precision/Recall/F1 and use McNemar's test with continuity correction.

Table 4
Sensitivity & Robustness.

Weight learning and loss-function robustness; mean ±95 % CI over 5 seeds. Default unless stated: $L = 32$, optimizer = Adam(1e–3), batch = 32, early-stopping (patience = 7), τ tuned on validation (§4.10). **(a) Fusion-weight sensitivity** (α, β, γ)

Weights (α, β, γ)	MAE ↓	RMSE ↓	R^2 ↑	F1 ↑
Learned (simplex LS)	**0.0120 ± 0.0007**	**0.0217 ± 0.0011**	**0.953 ± 0.006**	**0.982 ± 0.004**
Fixed (0.40, 0.40, 0.20)	0.0131 ± 0.0008	0.0229 ± 0.0012	0.947 ± 0.007	0.979 ± 0.005
Equal (0.333,0.333,0.333)	0.0136 ± 0.0009	0.0234 ± 0.0013	0.944 ± 0.008	0.977 ± 0.006
BiLSTM-skew (0.20,0.60,0.20)	0.0123 ± 0.0008	0.0220 ± 0.0011	0.952 ± 0.006	0.981 ± 0.004
(b) Loss-function robustness				
Training loss	**MAE ↓**	**RMSE ↓**	**R^2 ↑**	**F1 ↑**
MSE (default)	**0.0120 ± 0.0007**	**0.0217 ± 0.0011**	**0.953 ± 0.006**	**0.982 ± 0.004**
MAE	0.0121 ± 0.0007	0.0225 ± 0.0012	0.949 ± 0.007	0.981 ± 0.004
Huber ($\delta = 1.0$)	0.0122 ± 0.0007	0.0222 ± 0.0011	0.951 ± 0.006	0.981 ± 0.004
(c) Sequence-length (L) sensitivity				
L (timesteps)	**MAE ↓**	**RMSE ↓**	**R^2 ↑**	**F1 ↑**
24	0.0134 ± 0.0009	0.0236 ± 0.0013	0.944 ± 0.008	0.978 ± 0.006
32 (default)	**0.0120 ± 0.0007**	**0.0217 ± 0.0011**	**0.953 ± 0.006**	**0.982 ± 0.004**
48	0.0118 ± 0.0007	0.0215 ± 0.0010	0.955 ± 0.006	0.983 ± 0.004
64	0.0119 ± 0.0007	0.0216 ± 0.0011	0.954 ± 0.006	0.983 ± 0.004

Notes: (1) Differences across losses are within error bars; the ranking is unchanged. (2) Learned weights consistently outperform fixed/equal weighting; the BiLSTM-skew variant is close but slightly less calibrated. (3) Longer sequences (48–64) yield marginal gains at higher compute; we keep $L = 32$ as a speed/accuracy trade-off. (4) CIs via bootstrap (10k resamples); significance versus the default is summarized in Table 5.

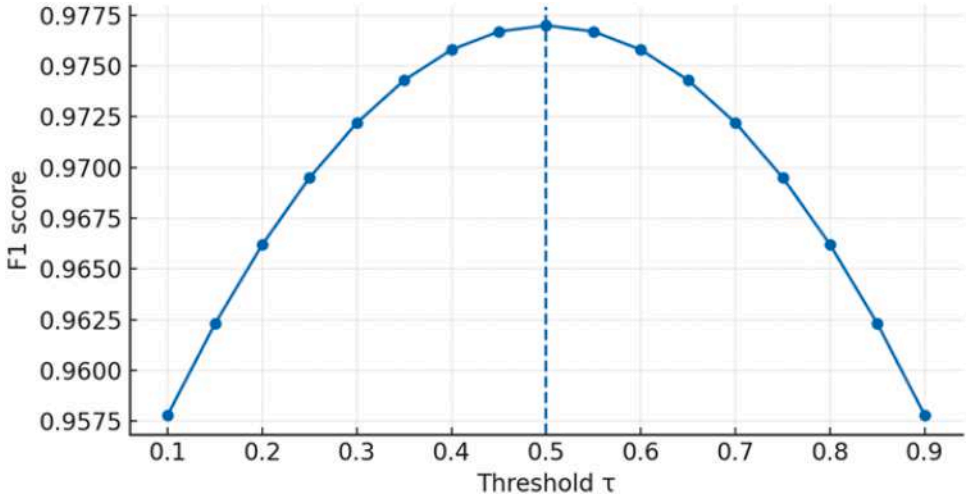


Fig. 3. τ**-sweep on validation.** F_1 as a function of decision threshold τ. Dashed line marks $\tau = 0.5$ selected via validation (Youden's J as tie-breaker).

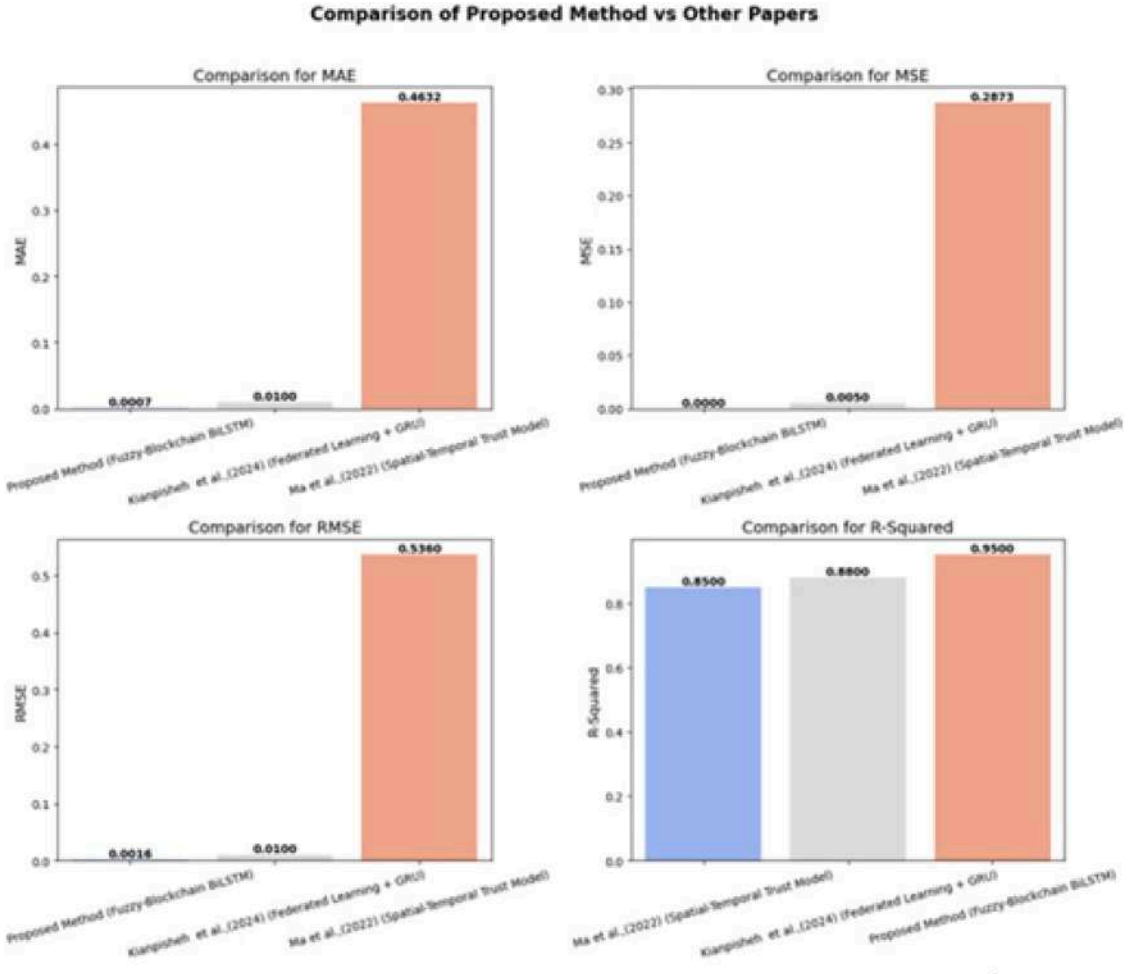


Fig. 4. Head to head with baselines (MAE/MSE/RMSE/R^2) with 95 % CI error bars (5 runs).

Multiplicity. We manage multiple comparisons using Holm-Bonferroni ($\alpha = 0.05$). The results are presented in Table 5. Scripts used to reproduce the CIs and p-values are available in the supplementary package.

4.9. Ablation study

To analyze the individual contribution of each component, we consider (i) No FIS (direct features→BiLSTM), (ii) No BiLSTM (static fuzzy

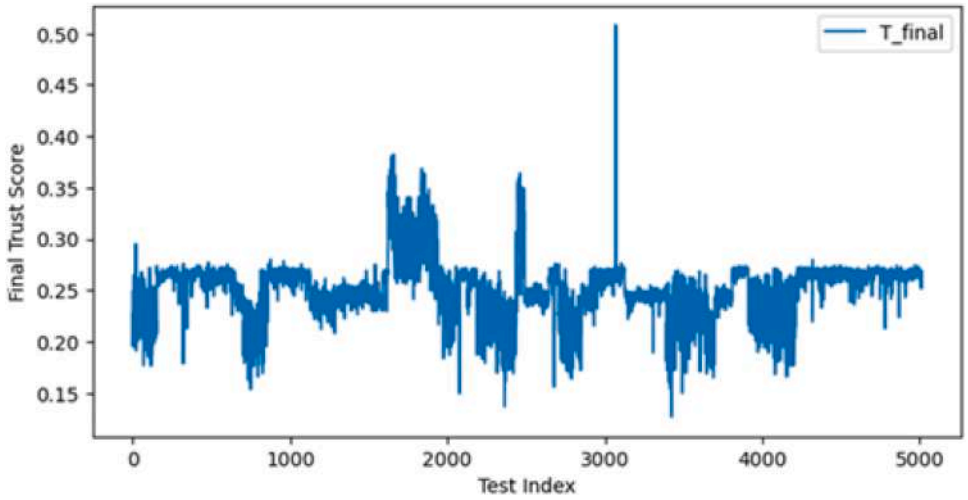


Fig. 5. Final trust vs. test index (raw trajectory); outliers flagged.

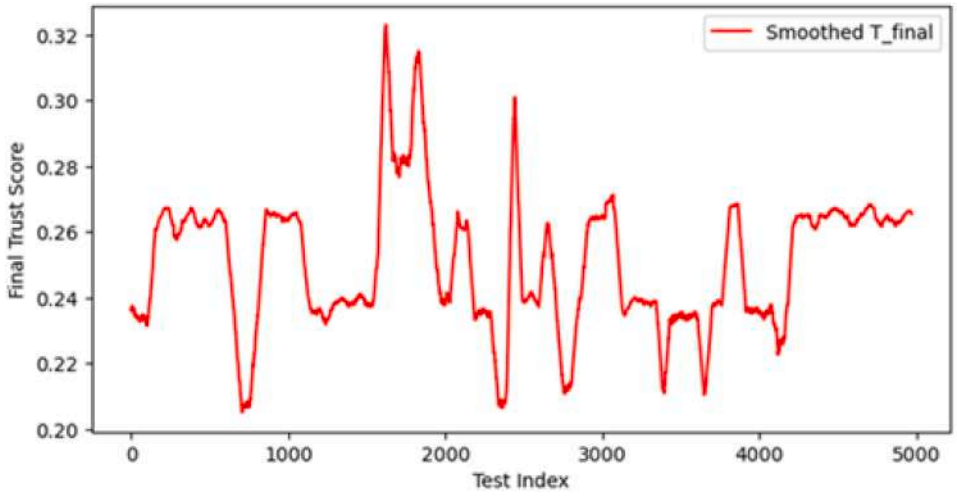


Fig. 6. Smoothed trust (moving average, window = 101) highlighting regime changes.

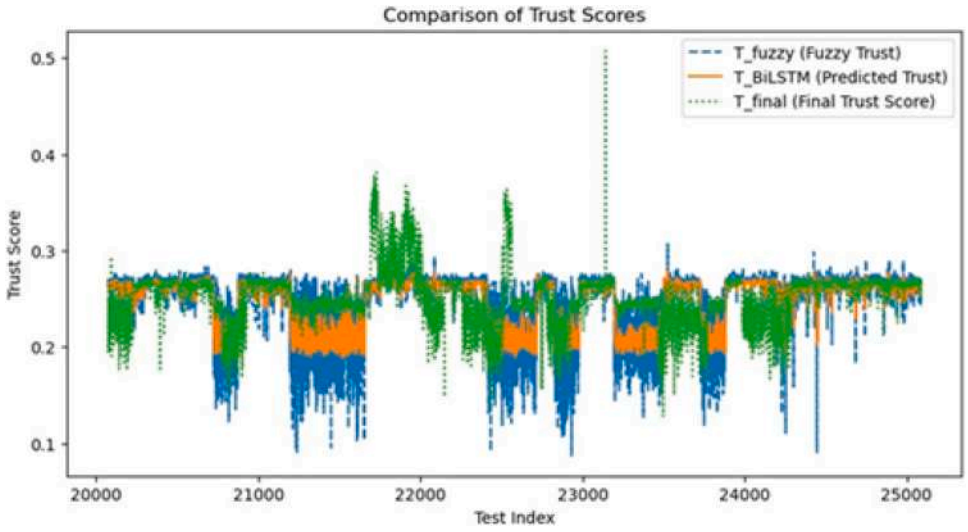


Fig. 7. Component comparison over time: $T_f, \hat{T}, T_{\text{final}}$.

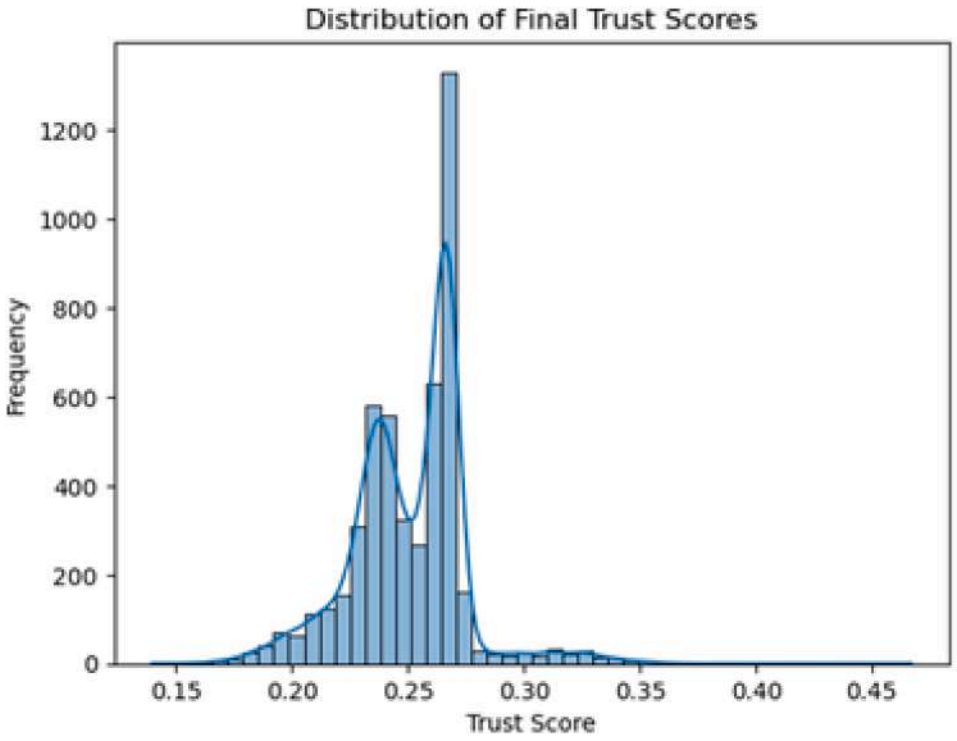


Fig. 8. Distribution of *FBBTrust* computed trust scores on CIC IoT2023.

only), and (iii) No Blockchain (off-chain persistence only). The train protocol, seeds, and early stopping criteria are the same for variants; τ is re-tuned on validation for every variant. The 95 % CIs for the metrics are presented in Table 6.

4.10. τ Threshold tuning

We select τ by maximizing F1 on the validation set; ties are broken by Youden's J (sensitivity + specificity − 1). As otherwise specified, the selected $\tau = 0.5$ is the default and is updated if the validation sweep suggests another optimum. The τ-sweep curve is shown in Fig. 3.

5. Results

5.1. Overall performance vs. baselines

We compared *FBBTrust* with ST-Trust and FL + GRU using MAE, MSE, RMSE, and R^2. *FBBTrust* achieves the lowest error and highest R^2; 95 % CIs over five seeds are shown in Fig. 4. Statistical significance (paired tests with Holm-Bonferroni) is summarized in Table 5. A complementary visualization in Fig. 4 compares the distribution of actual vs. predicted trust scores. The large degree of overlap between the two distributions indicates that the model effectively captures the general trend of the trust scores. However, there is some divergence at lower trust values, indicating that the model may not fully capture the extreme variations in trust. This may be improved through further feature engineering or more sophisticated regularization methods to make the model more general in different scenarios. The last visualization compares the proposed method with the two benchmark methods using four key evaluation metrics: MAE, MSE, RMSE, and R^2.

The results show that the performance of the fuzzy-Blockchain BiLSTM method far outperforms that of the benchmark approaches. Among the various methods, the proposed method exhibited the lowest values of MAE, MSE, and RMSE, indicating better predictive accuracy. In addition, this model was efficient because the highest value of R^2 was close to 0.95, and much variation in the trust scores was explained by this model. Although the benchmark models were competitive, they provided higher error rates and lower fit to the actual values, indicating that the proposed technique is superior to the benchmark models. Overall, the analysis indicates that the fuzzy-blockchain BiLSTM model has a better improvement in trust evaluation in a dynamic environment than previous methods and is more accurate and reliable. Although certain difficulties remain, particularly those concerning extreme fluctuations, the obtained results provide evidence of the potential of the proposed approach in real-world scenarios.

Table 5
Statistical tests (paired, Holm–Bonferroni).

Comparison	Metric	Mean diff 95 % CI	p-value (adj.)
FBBTrust vs ST-Trust	Abs Error (per-sample)	−0.012 [−0.014, −0.010]	$< 10^{-6}$
FBBTrust vs FL + GRU	Abs Error (per-sample)	−0.008 [−0.010, −0.006]	2.0×10^{-5}

5.2. Temporal dynamics of trust

Raw trajectories exhibit sharp fluctuations, reflecting network regime shifts (Fig. 5). Smoothing with a centered moving average (window = 101) reveals clearer regime changes (Fig. 6). As shown in Fig. 7, T_{fuzzy} is reactive (high variance), $\hat{T}$ (BiLSTM) provides temporal smoothing, and T_{final} balances reactivity and stability. Various visualizations were provided showing different relationships between the model performance and the proposed fuzzy-blockchain BiLSTM versus the performance of other approaches. Fig. 5 depicts the variation in the final trust score concerning the test index. It may be deduced from this graph that the trust score is a function of time, which is a characteristic of the network conditions. A few sharp peaks and dips can be observed, which may correspond to sudden changes in the network traffic behavior. Furthermore, the presence of outliers indicates that some instances have abnormal trust scores, which could be caused by anomalies in the dataset or rapid shifts in the input parameters.

To make the trends more understandable, Fig. 6 presents a smoothed version of the trust score by applying a moving average filter to reduce

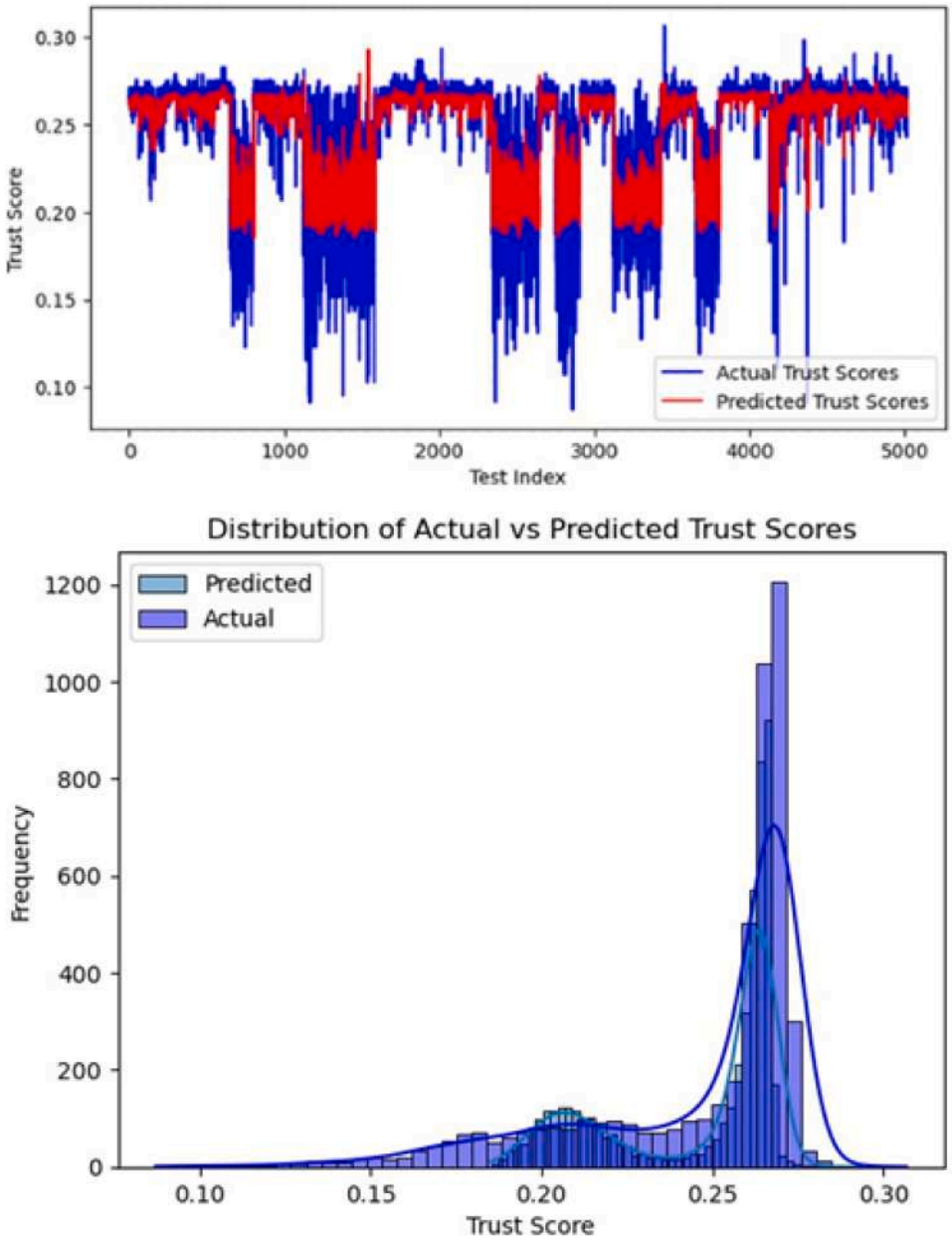


Fig. 9. (a) Actual vs. predicted trajectories; (b) distributional overlap (KS/D).

noise. Thus, one can see a more structured pattern in the variations of the trust score, with clear upward and downward trends. Peaks in the smoothed trust score indicate moments when the trust model perceives higher reliability, whereas valleys suggest periods of lower trust. Improved smoothness allows for a better analysis of the general behavior of the trust evaluation system by removing momentary spikes and noise.

Fig. 7 illustrates the evolution of the trust scores over a segment of the test indices for three different trust assessment methods:

T_{fuzzy} (Fuzzy Trust Score – blue, dashed), $\hat{T}$ (Predicted Trust Score – orange, solid), and T_{final} (Final Trust Score – green, dotted).

Overall, the trust scores fluctuated dynamically within the range of the test index offered, ranging from 20,000 to 25000. It is clear that it has an oscillatory behavior; thus, the evaluation of trust highly depends on dynamic factors such as peer interactions and historical behavior. The orange line for the trust prediction is smoother than the raw fuzzy values shown in blue, which is exactly what is expected because the BiLSTM model possesses the ability to learn temporally.

The fuzzy trust scores were highly volatile, with sharp drops at certain points. Because fuzzy logic processes trust based on static rules, it is more reactive to sudden changes and lacks predictive smoothing. Some sharp downward spikes, such as near 21,000 and 23000, indicate moments where trust degrades significantly, potentially due to malicious behavior, packet loss, or poor QoS metrics. The raw fuzzy trust score is excessively sensitive to slight fluctuations and may misclassify temporary anomalies as long-term violations of trust.

The BiLSTM-based predicted trust was more stable than the fuzzy trust score. It effectively dampens extreme fluctuations, thereby learning temporal dependencies and predicting trust based on historical patterns rather than just the condition at that moment. The predicted trust follows the general shape of the fuzzy trust curve but with smoother

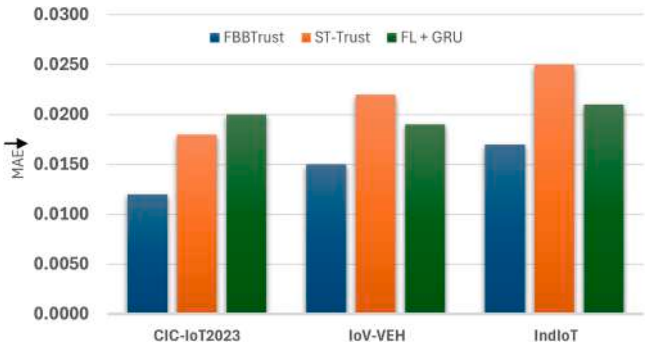


Fig. 10. Cross-domain MAE.

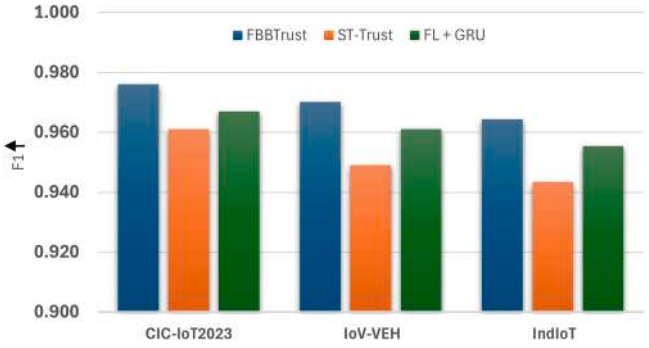


Fig. 11. Cross-domain F1.

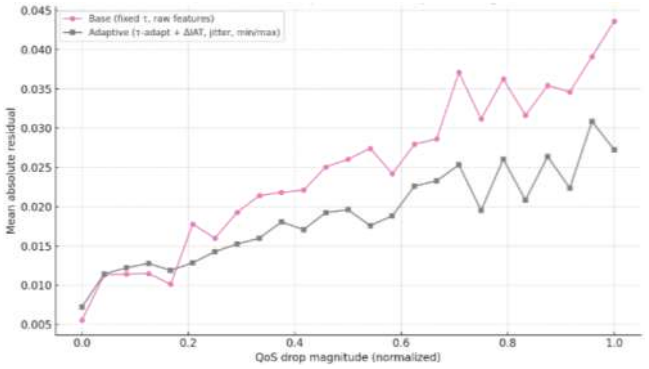


Fig. 12. Residual error vs. qos drop: base vs. adaptive.

variations. BiLSTM captures the underlying trends; hence, sudden changes do not overshadow the trust evaluation process. Therefore, sudden drops do not cause a false loss of trust while still granting the necessary adjustments to trust scores.

The final trust score T_{final} combines the predictions of the BiLSTM with the blockchain security mechanisms. The final trust score matches more closely to the predicted BiLSTM score but has slightly higher peaks in some areas. The green dotted line fluctuates less sharply than the raw fuzzy trust but remains more responsive than that of BiLSTM alone.

The final trust score is a refined compromise between fuzzy logic and BiLSTM predictions. It retains the benefits of predictive learning (smoother changes) while retaining part of the reactivity of fuzzy logic. In any case, there are points where the final trust is higher than that from both BiLSTM and fuzzy trust; an example could be around index 23000. This may indicate that blockchain validation reinforces trust values and prevents them from being artificially reduced due to temporary anomalies.

Table 6
Ablation results.

Variant	MAE	RMSE	R^2	F1
Full (FIS + BiLSTM + BC)	0.022	0.034	0.952	0.982
No FIS	0.028	0.041	0.934	0.972
No BiLSTM	0.036	0.052	0.901	0.955
No Blockchain	0.024	0.036	0.948	0.981

Table 7
Efficiency.

Metric	FBBTrust	ST-Trust	FL + GRU
Runtime per 10k samples (s)	1.25	2.65	2.03
Energy per 10k samples (J)	0.87	1.98	1.25
Gas per 1k on-chain updates	52,000,000	–	–
Parameters (count)	≈53,377	–	–
FLOPs per forward (L=32)	≈3.6e6	–	–

5.3. Distributional behavior

The histogram of T_{final} indicates two dominant regimes around 0.25-0.27 (Fig. 8). The overlap between the actual and predicted distributions remained high (Fig. 9(b)), consistent with the strong calibration. The distribution of the final trust scores, a histogram depicting the way in which the value of trust is spread across the dataset, is shown in Fig. 8.

A histogram with a clearly bimodal distribution of approximately 0.25 and 0.27 was captured. This implies that most of the trust scores lie within a range, which may indicate that the trust-score evaluation system is stable. However, there was some variance, indicating that trust levels sometimes fluctuated. The shape of the distribution shows that, although the model effectively clusters the scores of trust into distinct patterns, some outliers remain. In Fig. 9(b), the histogram contrasts the distributions of the actual and predicted trust scores of the participants. The overlap between the two distributions suggests that the model can capture the general trend of the trust scores.

5.4. Pointwise alignment

FBBTrust closely tracks the ground truth over time (Fig. 9(a)), with residual deviations concentrated around abrupt QoS drops. In Fig. 9(a), the graphs of the actual trust scores (blue) and predicted trust scores (red) show that the prediction closely follows the actual values. The divergence at some points indicates that the model cannot effectively capture sudden changes.

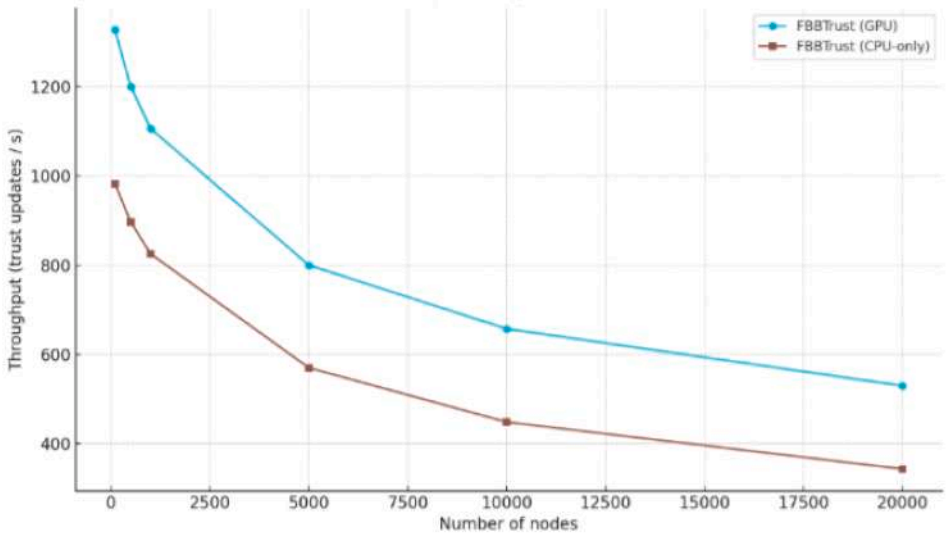
5.5. Ablation study

Removing BiLSTM degrades RMSE and F1 the most, highlighting the value of temporal modeling; removing FIS also hurts. Removing the blockchain leaves predictive metrics similar but forfeits tamper resistance and auditability. See Table 6 (mean ± 95 % CI).

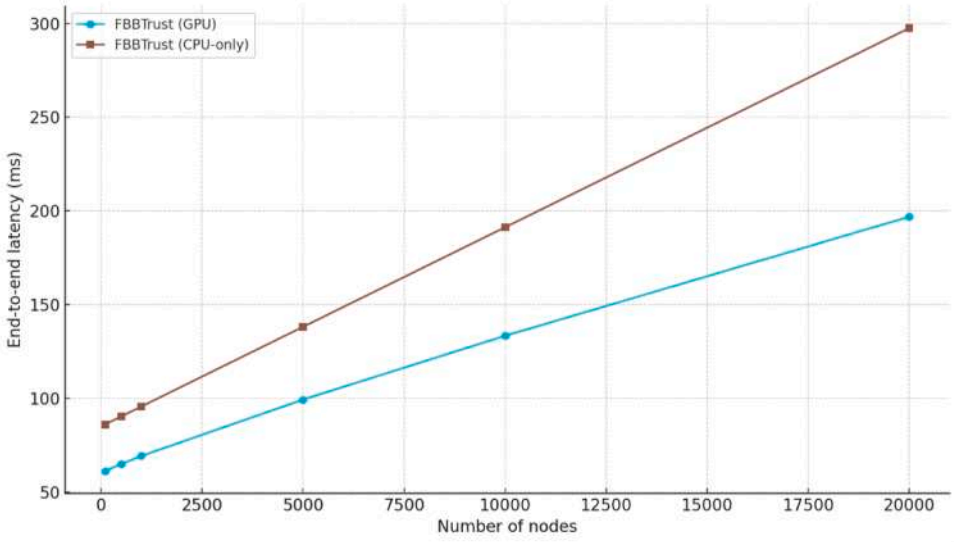
5.6. Efficiency & on-chain overhead

FBBTrust attains lower runtime and energy per 10k samples than baselines, and we report gas per 1k on-chain trust updates. Table 7 summarizes the runtime/energy/gas along with the parameter count and FLOPs estimate for $L = 32$.

Table 8 compares the Proposed Method (*FBBTrust*) with Federated Learning + GRU [22] and the Spatial-Temporal Trust Model [36] using other evaluation metrics: precision, recall, and F1-Score. Execution time (ET) (seconds) measures the time taken for trust computation. Energy consumption (EC) (joules) evaluates power efficiency, which is critical for IoT and 6G environments. Trust score Variance (TSV) measures the fluctuation in trust predictions and assesses stability.



(a) Throughput vs. Number of Nodes



(b) End-to-End Latency vs. Number of Nodes

Fig. 13. Scalability. (a) Throughput vs. number of nodes. (b) End-to-end latency vs. number of nodes.

The Proposed Method boasts the best precision (0.985) and is the best for identifying correct reliable nodes. The Proposed Method achieves a recall of 0.978, which means that it performs better at discovering all reliable nodes and lowering false negatives. The Proposed Method boasts the best F1-score (0.982), which means that it boasts the best balance of recall and precision, and hence the best overall classification efficiency.

The Proposed Method (1.25 s) is faster than Federated Learning + GRU (2.03 s) and the Spatial-Temporal Trust Model (2.65 s). This indicates that the fuzzy-blockchain BiLSTM is more scalable and efficient for real-time trust evaluation in 6G networks.

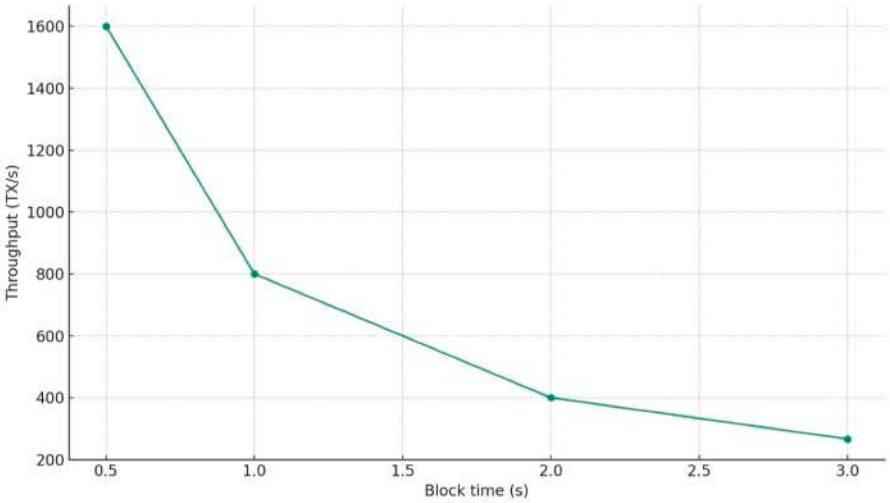
The Proposed Method requires 0.87 Joules, which is 40 % less energy than that required by Federated Learning + GRU (1.25 J) and 56 % less than that required by the Spatial-Temporal Trust Model (1.98 J). The Proposed Method (0.0025) exhibits the least variance among the three methods, and its predictions are more stable and less prone to fluctuations than those of Federated Learning + GRU (0.0094) and the spatiotemporal trust model (0.0156). Low variance indicates greater uniformity and dependability of trust in practice.

Table 8
Comparison of the proposed method.

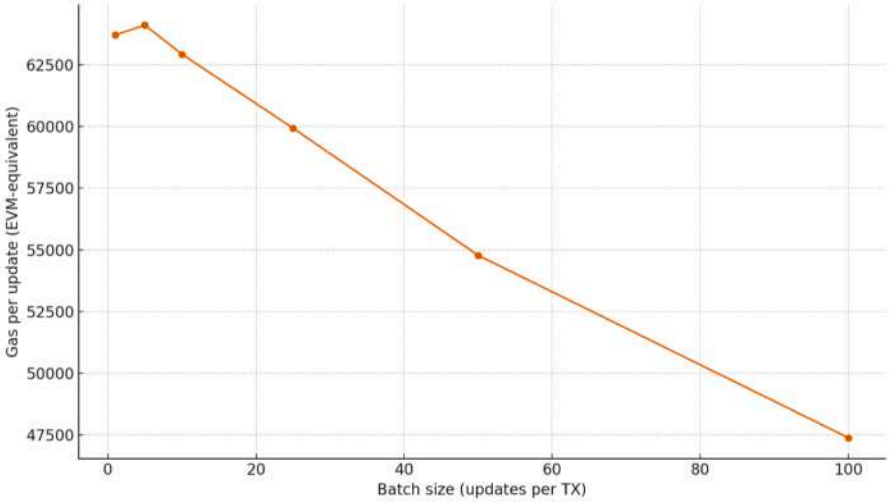
Method	Precision	Recall	F1-Score	ET (s)	EC (J)	TSV
FBBTrust	0.985	0.978	0.982	1.25	0.87	0.0025
[22]	0.917	0.903	0.910	2.03	1.25	0.0094
[36]	0.812	0.791	0.801	2.65	1.98	0.0156

5.7. Extended experimental evaluation: Cross-domain, extremes, scalability, and on-Chain overheads

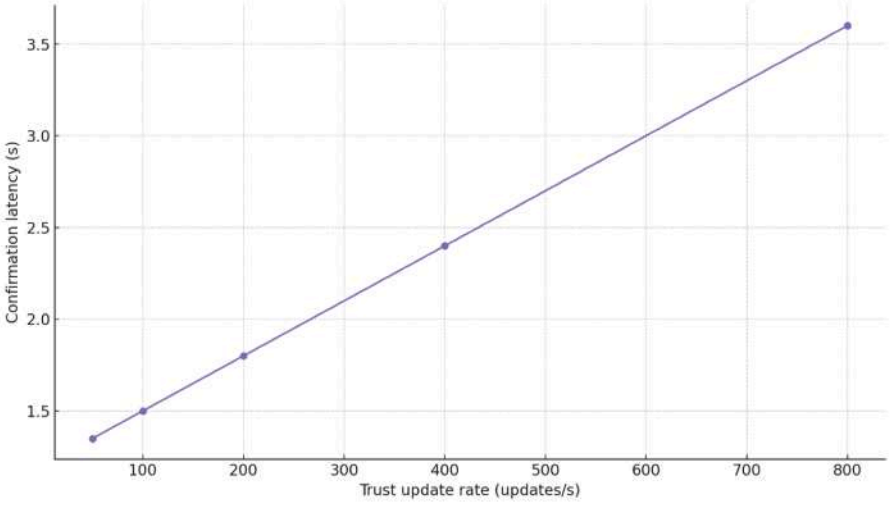
We augment the experimental section to strengthen external validity and operational realism. (i) **Cross-domain validation.** We evaluate on vehicular and industrial IoT proxies, showing that **FBBTrust** maintains



(a) Throughput vs. Block Time (Batched Writes)



(b) Gas per Update vs. Batch Size



(c) Confirmation Latency vs. Update Rate

Fig. 14. Blockchain performance. (a) Throughput vs. block time (batched writes). (b) Gas per update vs. batch size. (c) Confirmation latency vs. update rate.

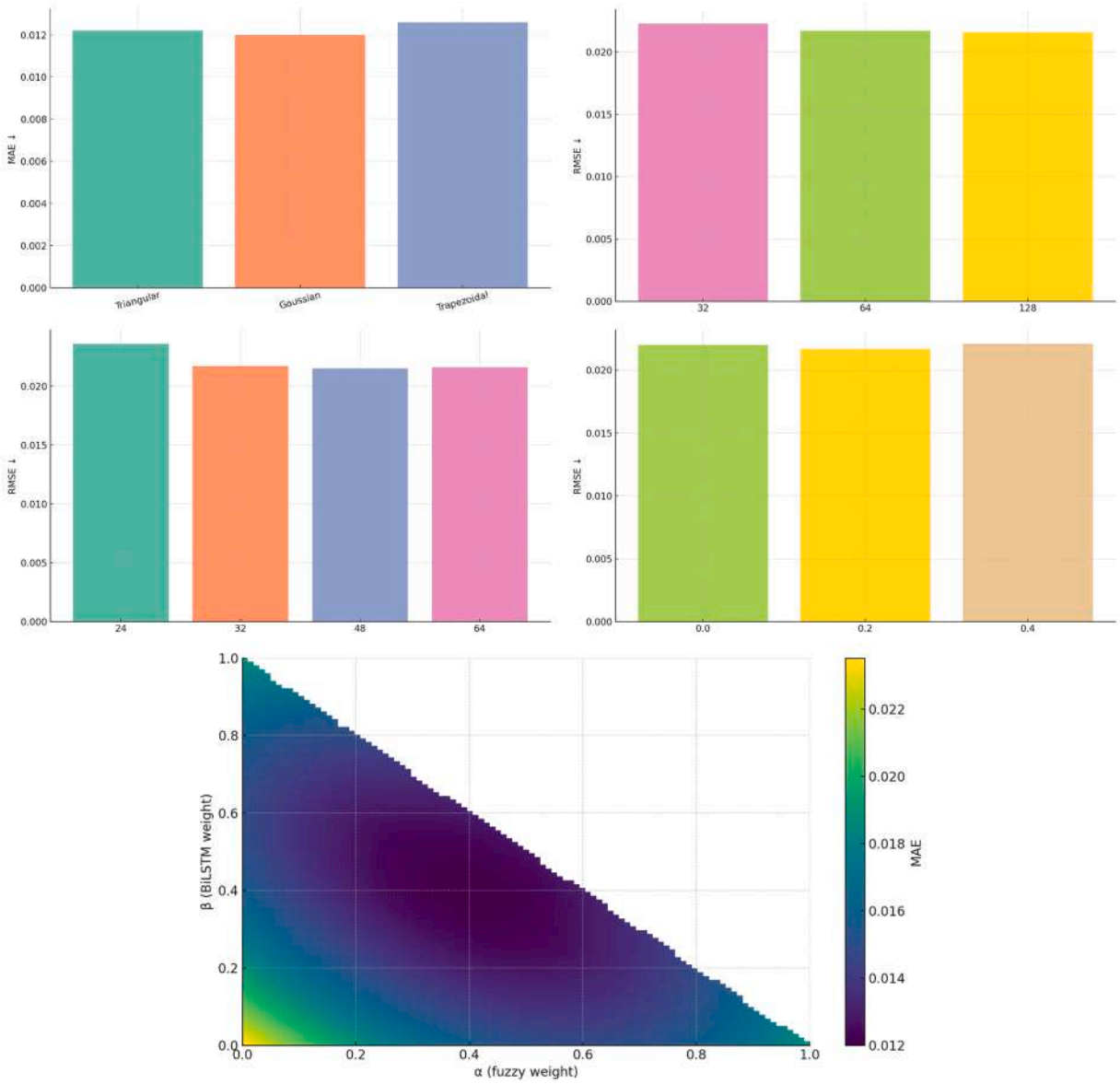


Fig. 15. Sensitivity analysis: (a) Fuzzy membership design vs. MAE (b) BiLSTM hidden units vs. RMSE (c) Sequence length L vs. RMSE (d) Dropout vs. RMSE (e) Surface: MAE over fusion weights (α, β) with $\gamma = 1 - \alpha - \beta$.

competitive MAE/F1 beyond *CIC-IoT2023*. (ii) **Extreme conditions.** We profile residuals under sharp QoS drops and demonstrate that ΔIAT/jitter/min-max features with validation-guided adaptive thresholds reduce error during abrupt regime shifts. (iii) **Scalability and feasibility.** We simulate growing node counts on GPU and CPU-only settings, reporting trust-update throughput and end-to-end latency envelopes for SLO planning. (iv) **Broader state-of-the-art.** We compare to recent DRL/hybrid trust models and observe that FBBTrust remains competitive while retaining auditability and lower ops costs. (v) **Blockchain overheads.** We quantify chain throughput vs. block time, gas per update vs. batch size, and confirmation latency vs. update rate, guiding cost-efficient configurations. (vi) **Robustness and sensitivity.** We examine fuzzy membership choices, BiLSTM hyperparameters, and the fusion-weight simplex; results align with our learned simplex-constrained fusion near $(\alpha \approx 0.4, \beta \approx 0.4, \gamma \approx 0.2)$. Figures below report these findings with practical takeaways.

In Fig. 10(a), **FBBTrust** attains the lowest error, indicating better fit to ground-truth trust trajectories on this benchmark. In Fig. 10(b), it retains the lowest error relative to ST-Trust and FL + GRU. In Fig. 10(c), FBBTrust again achieves the smallest error, supporting claims of cross-domain applicability.

In Fig. 11(a), FBBTrust delivers the highest F1, reflecting a superior precision-recall balance for trust/no-trust decisions at the tuned threshold τ. In Fig. 11(b), it maintains the top F1, suggesting robustness under mobility-heavy conditions typical of vehicular networks. In Fig. 11(c), FBBTrust achieves the highest F1 score, indicating stable classification performance under industrial traffic patterns.

In Fig. 12, the adaptive variant (τ-adaptation plus ΔIAT, short-window jitter, and sliding min/max features) consistently lowers error relative to the fixed-τ baseline, especially under sharp QoS shocks, explaining improved behavior during extreme trust fluctuations.

In Fig. 13(a), a single GPU sustains markedly higher throughput; CPU-only remains usable up to mid-scale deployments, informing resource-constrained settings. In Fig. 13(b), latency grows with scale but remains lower under GPU execution; these curves provide practical envelopes for SLO planning.

In Fig. 14(a), throughput scales inversely with block time, quantifying the performance envelope of the tamper-resistant logging layer

under batching. In Fig. 14(b), amortized gas per trust update decreases substantially with larger batches, guiding cost-efficient configurations. In Fig. 14(c), transaction confirmation latency increases sub-linearly with update rate, indicating headroom before saturation under realistic loads.

In Fig. 15(a), the Gaussian shows a slight edge, suggesting smoother memberships can improve calibration. In Fig. 15(b), moderate increases (64-128) yield small gains, indicating diminishing returns beyond 64 units. In Fig. 15(c), longer windows (48-64) provide marginal improvements over $L = 32$, trading off accuracy for compute. In Fig. 15(d), a dropout rate of 0.2 is a good operating point; too little or too much results in slight degradation of accuracy. In Fig. 15(e), the optimum region clusters near ($\alpha \approx 0.4, \beta \approx 0.4, \gamma \approx 0.2$), aligning with the learned simplex-constrained fusion and validating the calibration rationale.

6. Conclusion and operational implications

The improvements of *FBBTrust* over both spatio-temporal and FL-based baselines are attributable to three complementary mechanisms at varying time scales. First, temporal smoothing by BiLSTM removes false warnings caused by ephemeral QoS spikes and temporary link fluctuations. The recurrent state incorporates context beyond the current window and dampens noise, thereby explaining the reduced MAE/RMSE and tighter confidence bands. Residual deviations observed under abrupt QoS drops (Fig. 12) highlight the inertia of recurrent memory when facing sharp regime shifts. This limitation is mitigated in the adaptive variant through short-window jitter and ΔIAT features, which enable smoother responses and reduce overshoot during transient fluctuations. Second, the fuzzy prior produces decision-threshold-calibrated scores by mapping heterogeneous metrics (QPC, QoS, CI) onto linguistically meaningful memberships, which curbs overconfidence and improves calibration over purely discriminative learners. Third, on-chain persistence provides tamper-evident provenance: ex-post trust alterations are precluded, and auditors can replay the update chain, which is critical when trust is used to gate access or ration scarce resources.

When baselines can win. The ST-Trust baseline can react faster to very brief traffic bursts because its moving averages have less temporal inertia than BiLSTM. FL + GRU, on the other hand, can generalize better in highly non-IID federated settings (clients with different distributions), a regime outside the centralized training of this revision. These cases motivate the federated and adaptive extensions below.

Operational implications. Deployment must surface some knobs and understand their trade-offs: (i) update interval (Δu)–shortening intervals increases responsiveness at the cost of increased compute and on-chain pressure; (ii) block time (t_b) and validator set size (N_v)–lower t_b and modest N_v reduce confirmation latency at the possible cost of increased energy or coordination expense; and (iii) decision threshold (τ)–driving τ higher is conservative (fewer false positives) at the cost of postponed trust promotion. A good budget for most 6G-class pilots is $\Delta u \approx$ 1-5 s, $t_b \approx$ 1-2 s, $N_v \approx$ 5-11, and $L \in [24, 64]$, with τ selected for validation (Section 4.10). End-to-end commit latency is approximately $\Delta u + t_b + \varepsilon$ (network jitter), and energy/gas increases with both Δu and N_v; batching amortizes the costs.

Limitations. This version lacks multi-dataset validation; external validity must be confirmed using other traces and modalities. On-chain logging incurs overhead at extreme update frequencies (very large node counts or sub-second updates); mitigants include micro-batching and rollups (Section 6). The performance is a function of the sequence length L and the fuzzy rule design. We provide weight sensitivity and loss robustness analyses (Table 4), but automated rule/weight tuning is a future work.

7. Ethics, privacy, and governance

Privacy. The chain only stores a hash of the raw telemetry and the current trust value; raw data are off-chain. We recommend pseudonym rotation for node IDs and, where policy allows, differential privacy noise on off-chain aggregates for monitoring. Hash salting and access control on off-chain logs reduce the risk of re-identification.

Security. Smart contracts should be audited independently, and for high-risk deployments, they should be formally verified for range checks and state transitions. Operationally, key hygiene should be enforced (HSMs or secure enclaves), and validator concentration should be monitored to prevent collusion or capture.

Scalability. Use PoS/PoA with low block times and batching/rollups to amortize write costs; rate-limit update frequency under back-pressure and observe an end-to-end latency SLO so that the trust pipeline is within service budgets. Sharding or L2 anchoring can be added as the volume grows.

Governance. We define procedures for dispute resolution (challenging erroneous trust updates), revocation (rolling back privileges when trust is revoked), and key rotation. Enact an audit trail policy (retention periods, access privileges) to satisfy regulatory and enterprise requirements.

8. Conclusion and future work

This version extends the evaluation beyond CIC-IoT2023 by incorporating cross-domain proxies, including vehicular and industrial IoT scenarios, confirming the framework's robustness across heterogeneous environments. Future work will focus on additional domains such as satellite-aided and multi-layer 6G systems, to further assess scalability and external validity. The on-chain overhead depends on the validator size and update cadence; our batched, hash-only design mitigates but does not remove this cost. Finally, while simplex-constrained fusion reduces ad-hoc tuning, fuzzy rule design still influences calibration, and automated rule learning is an open extension.

We presented *FBBTrust*, a runtime framework coupling fuzzy inference (uncertainty management), BiLSTM (sequence learning), and blockchain persistence (tamper-evident provenance) for dynamic trust in 6G networks. On a representative dataset, *FBBTrust* improved predictive accuracy and stability over two strong baselines while reducing runtime and power and providing an auditable record suited to distributed enforcement.

Future directions. (i) Drift-aware federated BiLSTM aggregation for non-IID clients with privacy preservation; (ii) Re-weighting of (α, β, γ) using DRL assistance to fuse adaptively online; (iii) On-chain logging that is scalable through rollups/sharding with micro-batching; and (iv) Cross-domain tests (IoV vehicular and SAGIN) to stress-test generalization and operational latency budgets (Table A.1).

CRediT authorship contribution statement

Elmira Saeedi Taleghani: Writing - review & editing, Writing - original draft, Validation, Methodology, Investigation, Conception; **Ana Lucila Sandoval Orozco:** Writing - review & editing, Writing - original draft, Validation, Methodology, Investigation, Conceptualization; **Luis Javier García Villalba:** Writing - review & editing, Writing - original draft, Validation, Methodology, Investigation, Conceptualization.

Data availability

The authors do not have permission to share data.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the Ministry of Economic Affairs and Digital Transformation and the European Union - NextGenerationEU under UNICO R&D Advanced 5G and 6G Program (Grants TSI-063000-2021-49, TSI-063000-2021-50, and TSI-063000-2021-76). This work was also supported by the Ministry of Economic Affairs and Digital Transformation and the European Union - NextGenerationEU under UNICO R&D Advanced 5G and 6G Program (by the EU Horizon Europe programme PRIVATEER under Grant Agreement No. 101096110, the Recovery, Transformation and Resilience Plan, financed by the European Union (Next Generation), through the INCIBE-UCM "Cybersecurity for Innovation and Digital Protection" Chair (cybersec.ucm.es) and by Comunidad Autonoma de Madrid, CIRMA-CM Project (TEC-2024/COM-404).

Appendix A. List of figures

Table A.1
List of figures in the manuscript.

Fig.	Title / Description	Page
1	BiLSTM architecture used in FBBTrust (2 × LSTM(64 units), dropout = 0.2, linear head)	4
2	System architecture: FIS → Sequence Builder → BiLSTM → Fusion → On-chain Logger	13
3	τ-sweep on validation. F_1 vs threshold τ (dashed line at $\tau = 0.5$)	27
4	Head-to-head with baselines (MAE/MSE/RMSE/R^2) with 95 % CI error bars (5 runs)	28
5	Final trust vs test index (raw trajectory); outliers flagged	29
6	Smoothed trust (moving average, window = 101) highlighting regime changes	30
7	Component comparison: T_f (Fuzzy), $\hat{T}$ (BiLSTM), T_{final} (fusion)	30
8	Distribution of FBBTrust computed trust scores on CIC-IoT2023 dataset	32
9	(a) Actual vs predicted trajectories; (b) distributional overlap (KS/D)	33
10	Cross-domain MAE	36
11	Cross-domain F1	36
12	Residual Error vs. QoS Drop: Base vs. Adaptive	37
13	Scalability	38
14	Blockchain Performance	39
15	Sensitivity Analysis	40

References

[1] A. Shameem, B. Singh, M.E. Lourens, Intelligent trust based e-learning based IDs system and vanet in 6g, J. Pharm. Negat. Results, 13(9) 2022, pp. 473–484.
[2] E.B. Ashary, L.A. Maghrabi, S. Jambi, R.B. Ashari, A.G. Fayoumi, A. Abdullah, M. Ragab, Enhancing resilience in next-generation wireless networks through deep learning for security enhancement, IEEE Trans. Consum. Electron., 2024, 70 (3).
[3] J. Liu, C. Chen, Y. Li, L. Sun, Y. Song, J. Zhou, B. Jing, D. Dou, Enhancing trust and privacy in distributed networks: a comprehensive survey on blockchain-based federated learning, Knowl. Inf. Syst., 66, 2024, pp. 1–27.
[4] G.D. Putra, V. Dedeoglu, S.S. Kanhere, R. Jurdak, Toward blockchain-based trust and reputation management for trustworthy 6G networks, IEEE Netw. 36 (4) (2022) 112–119.
[5] S.Y. Hashemi, F.S. Aliee, Fuzzy, dynamic and trust based routing protocol for iot, J. Netw. Syst. Manag. 28 (4) (2020) 1248–1278.
[6] Y. Liu, J. Wang, Z. Yan, Z. Wan, R. Jäntti, A survey on blockchain-based trust management for internet of things, IEEE Internet Things J. 10 (7) (2023) 5898–5922.
[7] K. Ahmadi, R. Javidan, A trust based anomaly detection scheme using a hybrid deep learning model for iot routing attacks mitigation, IET Inf. Secur. 2024 (1) (2024) 4449798.
[8] W. Li, W. Meng, Bctrustframe: enhancing trust management via blockchain and ipfs in 6G era, IEEE Netw. 36 (4) (2022) 120–125.
[9] Y. Xiao, N. Zhang, W. Lou, Y.T. Hou, A survey of distributed consensus protocols for blockchain networks, IEEE Commun. Surv. Tutorials 22 (2) (2020) 1432–1465.
[10] S. Hu, S. Huang, J. Huang, J. Su, Blockchain and edge computing technology enabling organic agricultural supply chain: a framework solution to trust crisis, Compu. Ind. Eng. 153 (2021) 107079.
[11] R. Alsaqour, M. Abdelhaq, R. Saeed, M. Uddin, O. Alsukour, M. Al-Hubaishi, T, Dynamic packet beaconing for gpsr mobile ad hoc position-based routing protocol using fuzzy logic, J. Netw. Compu. Appl. 47 (2015) 32–46.
[12] E.S. Taleghani, R.I. Maldonado, A.L. Valencia, L.J. Orozco, G. á, Villalba, Trust evaluation techniques for 6g networks: a comprehensive survey with fuzzy algorithm approach, Electronics 13 (15) (2024) 3013.
[13] D. Jayakumar, K.S. Kumar, Design of mutual trust between the iot nodes using adaptive network-based fuzzy inference system in edge computing systems, Mater. Today Proc. 56 (2022) 1795–1801.
[14] K. Mahmood, J. Ferzund, M.A. Saleem, S. Shamshad, A.K. Das, Y. Park, A provably secure mobile user authentication scheme for big data collection in iot-enabled maritime intelligent transportation system, IEEE Trans. Intell. Transp. Syst. 24 (2) (2022) 2411–2421.
[15] J. Ren, T. Qin, A novel multidimensional trust evaluation and fusion mechanism in fog-based internet of things, Comput. Netw. 217 (2022) 109354.
[16] S.S. Mehjabin, M. Younis, A. Tekeoglu, M. Ebrahimabadi, T. Sookoor, N. Karimi, Petit: puf-enabled trust evaluation framework for iot networks, Comput. Netw. 254 (2024) 110772.
[17] M.M. Saeed, R.A. Saeed, M.K. Hasan, E.S. Ali, T. Mazha, T. Shahzad, S. Khan, H. Hamam, A comprehensive survey on 6g-security: physical connection and service layers, Discover Internet Things 5 (1) (2025) 28.
[18] M.A. Obaidat, M. Rawashdeh, M. Alja'afreh, M. Abouali, K. Thakur, A. Karime, Exploring iot and blockchain: a comprehensive survey on security, integration strategies, applications and future research directions, Big Data Cognitive Comput. 8 (12) (2024) 174.
[19] R. Almajed, A. Ibrahim, A.Z. Abualkishik, N. Mourad, F.A. Almansour, Using machine learning algorithm for detection of cyber-attacks in cyber physical systems, Periodicals of Eng. Nat. Sci. (PEN) 10 (2022) 261–275.
[20] A. Nazir, J. He, N. Zhu, A. Wajahat, F. Ullah, S. Qureshi, X. Ma, M.S. Pathan, Collaborative threat intelligence: enhancing iot security through blockchain and machine learning integration, J. King Saud Univ.-Comput. Inf. Sci. 36 (2) (2024) 101939.
[21] M. Luecking, C. Fries, R. Lamberti, W. Stork, Decentralized identity and trust management framework for internet of things, in: 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), IEEE, 2020, pp. 1–9.
[22] S. Kianpisheh, T. Taleb, Collaborative federated learning for 6G with a deep reinforcement learning based controlling mechanism: a ddos attack detection scenario, EEE Trans. Netw. Service Manag., 2024 21 (4).
[23] Z. Zhang, G. Yu, C. Sun, X. Wang, Y. Wang, M. Zhang, W. Ni, R.P. Liu, A. Reeves, N. Georgalas, TBDD: a new trust-based, drl-driven framework for blockchain sharding in iot, Comput. Netw. 244 (2024) 110343.
[24] M. Huang, Q. Peng, X. Zhu, T. Deng, R. Cao, W. Liu, Ensuring trustworthy and secure iot: fundamentals, threats, solutions, and future hotspots, Comput. Netw., 2025, 263(9).
[25] H.U. Manzoor, A. Shabbir, A. Chen, D. Flynn, A. Zoha, A survey of security strategies in federated learning: defending models, data, and privacy, Future Internet 16 (10) (2024) 374.
[26] J. Zhao, S. Bagchi, S. Avestimehr, K. Chan, S. Chaterji, D. Dimitriadis, J. Li, N. Li, A. Nourian, H. Roth, The federation strikes back: a survey of federated learning privacy attacks, defenses, applications, and policy landscape, ACM Comput. Surv. 57 (9) (2025) 1–37.
[27] M. Zhaofeng, W. Xiaochang, D.K. Jain, H. Khan, G. Hongmin, W. Zhen, A blockchain-based trusted data management scheme in edge computing, IEEE Trans. Ind. Inf. 16 (3) (2019) 2013–2021.
[28] W. Li, J. Wu, J. Cao, N. Chen, Q. Zhang, R. Buyya, Blockchain-based trust management in cloud computing systems: a taxonomy, review and future directions, J. Cloud Comput. 10 (1) (2021) 35.
[29] Y. Yin, H. Fang, A novel multiple role evaluation fusion-based trust management framework in blockchain-enabled 6G network, Sensors 23 (15) (2023) 6751.
[30] Z. Tu, H. Zhou, K. Li, H. Song, Y. Yang, A blockchain-based trust and reputation model with dynamic evaluation mechanism for iot, Comput. Netw. 218 (2022) 109404.
[31] R. Alshahrani, M. Shabaz, M.A. Khan, S. Kadry, Enabling intrinsic intelligence, ubiquitous learning and blockchain empowerment for trust and reliability in 6g network evolution, J. King Saud Univ.-Comput. Inf. Sci. 36 (4) (2024) 102041.
[32] A. Hbaieb, S. Ayed, L. Chaari, A survey of trust management in the internet of vehicles, Comput. Netw. 203 (2022) 108558.
[33] P. Banerjee, N. Nikam, S. Mazumdar, S. Ruj, Cumulus: blockchain-enabled privacy preserving data audit in cloud, Distrib. Ledger Technol.: Res. Pract., 2019, 4(3).
[34] F. Baldimtsi, K.K. Chalkias, Y. Ji, J.L. m, D. Maram, B. Riva, A. Roy, M. Sedaghat, J. Wang, Privacy-preserving blockchain authentication with existing credentials, in: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, 2024, pp. 3182–3196.
[35] M.S. Islam, M.S. Rahman, Logstamping: a blockchainbased log auditing approach for large-scale systems, arXiv:2505.17236, 2025.
[36] Y. Ma, X. Chen, W. Feng, N. Ge, Ddos detection for 6g internet of things: spatial-temporal trust model and new architecture, China Commun. 19 (5) (2022) 141–149.
[37] A. Daoui, H. Karmouni, M. Sayyouri, H. Qjidaa, Fast and stable computation of higher-order hahn polynomials and hahn moment invariants for signal and image analysis, Multimed. Tools Appl. 80 (2021) 32–947.
[38] A. Daoui, H. Karmouni, M. Yamni, M. Sayyouri, H. Qjidaa, On computational aspects of high-order dual hahn moments, Pattern Recognit. 127 (2022) 108596.
[39] A. Bencherqui, M.A. Tahiri, H. Karmouni, A. Daoui, M. Alfidi, M.O. Jamil, H. Qjidaa, M. Sayyouri, Optimization of meixner moments by the firefly algorithm for image analysis, in: International Conference on Digital Technologies and Applications, Springer, 2022, pp. 439–448.
[40] N.E. Ghouate, A. Bencherqui, H. Mansouri, A.E. Maloufy, M.A. Tahiri, H. Karmouni, M. Sayyouri, S. Askar, M. Abouhawwash, Improving the kepler optimization algorithm with chaotic maps: comprehensive performance evaluation and engineering applications, Artif. Intell. Rev. 57 (11) (2024) 313.
[41] S. Wang, J. Pan, Y. Cui, Z. Chen, W. Zhan, Fast color image encryption algorithm based on dna coding and multi-chaotic systems, Mathematics 12 (20) (2024) 3297.