This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



# Machine Learning-based Digital Forensics for Trust Assessment in Smart Grid IoT Using Blockchain

Wajahat Ali, Ikram Ud Din, *Senior Member, IEEE* , Ahmad Almogren, *Senior Member, IEEE*, Ayman Altameem, and Joel J. P. C. Rodrigues, *Fellow, IEEE*

***Abstract*—The smart grid is an advanced electrical infrastructure that integrates intelligent devices and applications to enhance efficiency and reliability. These devices include smart meters, transmission and distribution systems, and control centers. However, despite its advantages, many individuals remain hesitant to adopt smart grid technology due to concerns regarding privacy, data integrity, and trust, as their private information traverses the network. To address these concerns, we propose GridTrust, a digital forensics trust management framework that leverages machine learning and blockchain technology to build confidence among home users and facilitate the transition to smart grid adoption. GridTrust employs a Convolutional Neural Network and Bidirectional Long Short-Term Memory model to forensic real-time time-series data, while a private permissioned blockchain securely records and verifies the predicted patterns, ensuring transparency and security. To assess GridTrust performance, we evaluated it using key metrics such as accuracy, precision, recall, F1-score, ROC curve, transactions per second, average response time and computational resource utilization (memory and CPU). The results demonstrate that GridTrust outperforms existing models in all performance metrics, underscoring its effectiveness and efficiency in enhancing trust within smart grid systems.**

***Index Terms*—Blockchain, Control center, Convolution neural network, Long short term memory, Smart grid.**

## I. Introduction

THE journey of modern electricity began in 1831 when Michael Faraday discovered that an electric current could be induced by moving a conductor through a magnetic field. This groundbreaking discovery laid the foundation for converting mechanical energy into electrical energy on a large scale by 1839. While Faraday established the fundamental principles, it was Nikola Tesla, recognized in 1896, who is often regarded as the father of modern electricity, shaping the power systems that drive the world today [1].

The traditional electrical grid, originally developed over a century ago, was built around centralized power production and a passive delivery system for distributing electricity to consumers [2]. However, in today's era, the electric grid faces growing challenges concerning efficiency, reliability, and sustainability. The increasing global demand for electricity, coupled with the rising costs of power generation, underscores the need for sustainable and reliable alternatives to meet consumer demands [3], [4]. Due to these inefficiencies and system failures, global economic losses amount to approximately 96 billion dollars annually [5]. The United Kingdom incurs losses of 175 million dollars per year due to electricity-related issues [6]. In 2015 alone, India lost 16.2 billion dollars, Brazil 10.5 billion dollars, and Russia 5.1 billion dollars due to electricity inefficiencies [7]. Similarly, Peshawar electric supply company in Pakistan reported a staggering 36.2% energy loss in 2016-17 [8], [9]. These losses not only impact home consumers but also significantly affect national economies.

The smart grid emerges as a transformative solution, providing electrical companies with an advanced mechanism, enhanced by digital forensics, to combat inefficiencies and intrusions [10]. A smart grid is an intelligent electric system that integrates two-way communication among key elements, including home consumers, distribution gateways, control centers, power generation sources, and embedded smart devices. This advanced system also referred to as the intelligent grid or future grid enhances communication within the grid and between the grid and external networks [11].

Despite its advantages, smart grid technology faces critical challenges related to automation, privacy, security, and trust. Extensive research has been conducted to address privacy and security concerns, primarily focusing on protecting consumer energy consumption data through cryptographic techniques, external hardware solutions, and digital forensic methodologies [12]. However, trust management remains an underexplored area, particularly in smart grid environments [13]. Many consumers remain reluctant to adopt smart grid technology due to concerns about data privacy and information sharing. Therefore, it is crucial to first establish trust among consumers and all stakeholders involved in smart grid operations. This study introduces a novel hybrid trust management framework GridTrust by harnessing the synergy between blockchain technology and machine learning.

- To address the issue of class imbalance, the synthetic minority oversampling technique is employed, ensuring balanced representation of minority and majority classes in the dataset.
- Considering computational resource constraints, the extreme gradient boosting algorithm is utilized for effective extraction of the most informative features.
- A hybrid CNN-BiLSTM model is developed to perform forensic trust evaluation of devices operating within smart grid environments.

W. Ali and I.U. Din are with the Department of Information Technology, The University of Haripur, Pakistan (e-mail: wajahat.haripur@gmail.com, ikramuddin205@yahoo.com); A. Almogren and A. Altameem are with the Chair of Cyber Security, Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia (e-mail: ahalmogren@ksu.edu.sa, aaltameem@ksu.edu.sa); J. Rodrigues is with the Post-graduation Program in Electrical Engineering, Federal University of Piauí (UFPI), Teresina - PI, Brazil (e-mail: joeljr@ieee.org). Corresponding authors: Ahmad Almogren and Ikram Ud Din

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



Table I
COMPARATIVE ANALYSIS OF EXISTING SCHEMES.

| Ref | Contribution | Technique | Limitations |
|---|---|---|---|
| [14] | PTBiLSTM for theft detection | Random undersampling | Avoids underfitting using SMOTE. |
| [15] | STLF for Load Forecasting | Random forest | Ignore data balancing. |
| [16] | Intrusion Detection | Gradient boosting | Not consider data balancing. |
| [17] | IoT Trust Classification | LSTM model | No trust parameter selection. |
| [18] | Behavior-based Trust Detection | RNN-LSTM | Ignore data balancing and feature selection. |

- A private, permissioned blockchain is implemented to securely store and disseminate trust scores, ensuring data integrity and resistance to unauthorized modifications.
- An adaptive trust thresholding mechanism is incorporated to dynamically adjust trust levels, facilitating optimal and resilient trust management across the system.

The remainder of the paper is organized as follows: Section II presents the related work, Section III elaborates on the proposed methodology, and Section IV discusses the performance of the proposed scheme.

## II. RELATED WORK

Recent studies have explored trust evaluation and intrusion detection in smart grid environments using various machine learning models. For instance, [14] utilized a parameter-tuned BiLSTM model with random undersampling to detect electricity theft, yet suffered from underfitting due to loss of vital data during undersampling. Similarly, [15] employed random forest for feature selection and a CNN-BiGRU architecture for short-term load forecasting, but lacked comparative analysis with alternative feature selection approaches and omitted data balancing strategies. In [16] a gradient boosting-based feature extraction scheme for power grid intrusion detection is applied, but did not benchmark it against other extraction techniques, limiting its comparative rigor.

In another direction, [17] and [18] focused on trust-based classification using LSTM and RNN architectures for IoT devices. However, their methods lacked comprehensive data preprocessing steps such as NaN handling, balancing, and proper trust parameter selection, see Table I. In [19], a CNN model targeted anomaly detection with focus on false data injection but did not include feature extraction or address data balancing.

## III. PROPOSED GRIDTRUST MODEL

The GridTrust framework consists of a network of smart meters ($S$), a distribution gateway ($G$), and a control center ($C$). The system evaluates trust levels periodically over a fixed time interval $T = 24$ hours and updates trust values on the blockchain. The trust evaluation function is defined for each entity, considering key trust parameters, namely honesty ($H$), cooperativeness ($C$), reactiveness ($R$), reliability ($R_y$), response time ($T_r$), packet loss ratio ($PLR$), and availability

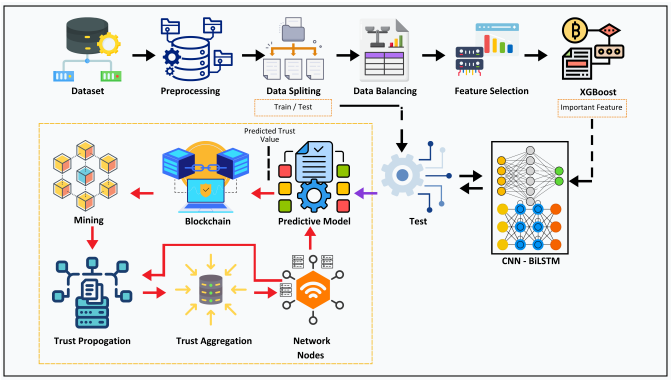


Figure 1. GridTrust system model.

($A$). GridTrust step by step explaination shown in Figure 1. The trust level of a smart meter $S_i$ at time $t$ is computed as:

$$T_i(t) = f(H_i(t), C_i(t), R_i(t), R_{y_i}(t), T_{r_i}(t), PLR_i(t), A_i(t)) \tag{1}$$

where $f$ is a weighted aggregation function, defined as:

$$T_i(t) = \sum_{j=1}^{7} w_j P_{ij}(t) \tag{2}$$

where $P_{ij}(t)$ represents the normalized value of the $j$-th trust parameter for the smart meter $i$, and $w_j$ is the weight associated with that parameter such that:

$$\sum_{j=1}^{7} w_j = 1, \quad 0 \leq w_j \leq 1 \tag{3}$$

A smart meter $S_i$ is allowed to communicate only if its trust value satisfies the threshold condition:

$$T_i(t) \geq \tau_S \tag{4}$$

where $\tau_S$ is the predefined trust threshold. If $T_i(t) < \tau_S$, the smart meter is suspended from communication.

Similarly, the trust level of the distribution gateway $G$ is evaluated by the control center using the same trust parameters:

$$T_G(t) = \sum_{j=1}^{7} w_j P_{G_j}(t) \tag{5}$$

where $P_{G_j}(t)$ represents the normalized trust parameter values for the distribution gateway. The gateway is permitted to communicate only if:

$$T_G(t) \geq \tau_G \tag{6}$$

where $\tau_G$ is the trust threshold for the gateway.

To ensure balanced training, data preprocessing involves handling class imbalance, using SMOTE, the dataset is balanced such that:

$$C_0' = C_1' = \max(C_0, C_1) \tag{7}$$

where $C_0'$ and $C_1'$ are the new counts of trusted and untrusted users after balancing.

Feature selection is performed using an XGBoost model. The selected features are then passed into the CNN-BiLSTM

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



trust computation model. The trust probability for a device $S_i$ is computed as:

$$P(T_i = 1|x_i) = \sigma(Wx_i + b) \tag{8}$$

where $x_i$ is the input feature vector of $S_i$, $W$ and $b$ are trainable parameters, and $\sigma(\cdot)$ is the sigmoid activation function:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{9}$$

The final trust decision is made as:

$$T_i = \begin{cases} 1, & P(T_i = 1|x_i) \geq \tau_S \\ 0, & P(T_i = 1|x_i) < \tau_S \end{cases} \tag{10}$$

After computing the trust level, the values are propagated and aggregated using the exponential moving average approach:

$$T_i'(t) = \alpha T_i(t) + (1 - \alpha)T_i(t - 1) \tag{11}$$

where $\alpha$ is the trust update factor ($0 < \alpha < 1$), and $T_i(t-1)$ is the previous trust value stored in the blockchain.

Once an entity passes trust verification, its updated trust value is permanently stored in the blockchain:

$$B(t) = B(t - 1) \cup \{T_i'(t)\} \tag{12}$$

where $B(t)$ represents the blockchain state at time $t$. If any entity falls below the trust threshold, it is suspended, and the control center is notified:

$$S_i \in \mathcal{S}_{\text{suspended}} \quad \text{if} \quad T_i(t) < \tau_S \tag{13}$$

$$G \in \mathcal{G}_{\text{suspended}} \quad \text{if} \quad T_G(t) < \tau_G \tag{14}$$

CNN-BiLSTM model evaluates the trustworthiness level of the honest and dishonest users. The trust level is defined in range of 0 and 1. Detail description of each steps are explained in below subsections.

### A. Preprocessing

In GridTrust we have utilized Edge IIoT dataset, which consist of missing values and unscaled data. Missing values also known as NaN values (not a number). These missing value happens in the dataset due to network failure, device failure, or storage issue [20]. If we trained the GridTrust model on the same dataset, GridTrust produces biased and overfitted results.As Edge IIoT dataset consists of 157,800 data samples, with 63 different features. Among these, 133,499 samples belong to untrusted users and 24,301 samples represent trusted users [20]. Linear interpolate and min-max normalization techniques are used to remove NaN values, and for normalization. The step by step definition of preprocessing phase are elaborated in Algorithm 1.

### B. Class Balancing

The class distribution for trusted and untrusted samples is 84.6% and 16.4%, respectively [20]. Training the dataset in its current form would lead to biased results, as the minority class would be underrepresented and ignored. Thus, it is crucial to balance both classes before proceeding with the training and testing phases to ensure fair and effective learning.

**Algorithm 1** Data Preprocessing

Dataset $X$

Define dataset $X$ as a collection of tuples:

$$X = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$$

Initialize variables for minimum, maximum, and mean for each consumer $x_i$ **for** *each row* $i = 1$ *to row* **do**

each column $j = 1$ to col **if** $X(i, d-1)$ *and* $X(i, d+1)$ *are not NaN and* $X(i, d)$ *is NaN* **then**

$X(i, d) \leftarrow \frac{X(i,d-1)+X(i,d+1)}{2}$

$X(i, d-1)$ OR $X(i, d+1)$ is NaN

$X(i, d) \leftarrow 0$

**Min-max normalization: for** *each element* $X(i, j)$ **do**

Normalize using:

$$X_{\text{norm}}(i, j) = \frac{X(i, j) - X_{\min}}{X_{\max} - X_{\min}}$$

**return** $X_{\text{norm}}$

**Algorithm 2** Class Balancing

Input (X):

Identify minority class samples $T$ from $X$

**for** *(Each sample* $a$ *in* $T$*)* **do**

  Find $k$-nearest neighbors of $a$ in $T$

  **for** *(each neighbor* $b$ *of* $a$ *selected from* $k$*-nearest neighbors)* **do**

    Calculate difference diff $\leftarrow a - b$

    Generate a random number gap between 0 and 1

    Create a new synthetic sample $n \leftarrow a +$ diff $\times$ gap

    Add $n$ to $X$ $X_{\text{train\_resampled}}$ = X.

Output: Balanced Dataset $X_{\text{train\_resampled}}$

In GridTrust, we employ SMOTE for data balancing, as it provides higher accuracy and minimal loss [21]. The average execution time for each epoch using SMOTE is approximately 5 seconds. SMOTE synthetically generates new samples for the minority class by selecting a data sample from the minority class along with its K nearest neighbors. The mathematical representation of this interpolation method is given by Equation 15.

$$X_i = A_i + (B_i - A_i) * \delta \tag{15}$$

Here, $X_i$ represents the newly synthesized data sample, while $A_i$ and $B_i$ denote the values of the K nearest neighbor columns. The term $\delta$ is a random number that determines where the new sample will be positioned. Once both trusted and untrusted classes are balanced, the dataset is ready for the next stage of training, as illustrated in Figure 2. A step-by-step breakdown of the SMOTE process and its working structure is presented in Algorithm 2.

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



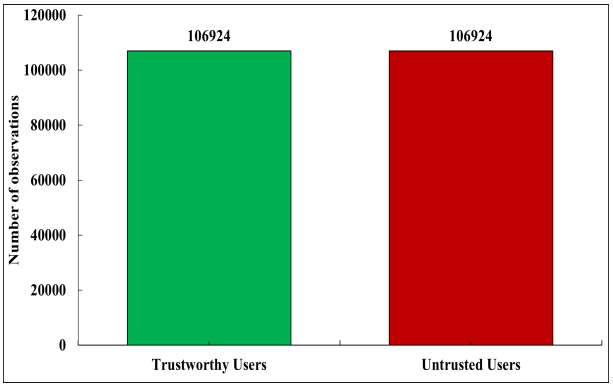


Figure 2. Distribution of trusted and untrusted data samples fter applying SMOTE.

### C. Feature Selection

Feature selection is the process of identifying the most significant and relevant input features from a large dataset [22]. Given computational constraints, it is infeasible to utilize all $d = 63$ features in the training and testing phases.

To optimize the model, feature selection is performed using extreme gradient boosting, which ranks features based on their importance scores. Let $F = \{F_1, F_2, \ldots, F_d\}$ be the full feature set, where $F_j$ represents the $j$-th feature. The importance score of each feature $F_j$ is computed as:

$$I(F_j) = \sum_{t=1}^{T} G_t(F_j) \tag{16}$$

where $G_t(F_j)$ denotes the gain contribution of feature $F_j$ in decision tree $t$, and $T$ is the total number of trees in the XGBoost model. The top $k = 25$ features with the highest importance scores are selected, forming the reduced feature subset:

$$F' = \{F_j \mid I(F_j) \text{ is among the top 25}\}, \quad F' \subseteq F \tag{17}$$

### D. CNN-BiLSTM Model

The CNN model in GridTrust consists of three convolution layers (Conv1D) and three MaxPooling1D layers. The first convolution layer comprises 8 filters, a 3×3 kernel size, and uses the ReLU activation function. The second convolution layer consists of 16 filters, a 3×3 kernel size, and also employs the ReLU activation function. Similarly, the third convolution layer has 32 filters with the same kernel size and activation function as the previous layers, see Algorithm 3. The primary purpose of the convolution layers is to forensically extract the most relevant features while reducing the input dimensions. The mathematical representations for the convolution operation and ReLU activation function are given below:

$$\text{output} = \text{n - f + 1} \tag{18}$$

$$\text{ReLU} = \max\ (0,\ \text{x}) \tag{19}$$

In Equation 18, n represents the size of the input, while f is the size of the filter (kernel). In Equation 19, x represents the output of the ReLU function. If the output value is less than 0, it is mapped to 0. Otherwise, if x ¿ 0, the output remains x.

Each convolution layer is followed by a pooling layer to further refine feature extraction and reduce input dimensions. In GridTrust, we use MaxPooling1D, which selects the maximum value within the selected convolution feature map. The output of the dataset after applying max pooling is computed using Equation 20:

$$\text{Output} = \frac{\text{Length of input} - \text{Feature map}}{\text{Stride}} + 1 \tag{20}$$

The output of the CNN model is then passed to the BiLSTM layer in the form of X. The BiLSTM model is an extended version of LSTM and consists of 64 neurons operating in both forward and backward directions. Since the problem is a binary classification task, the dense layer uses a single neuron with a sigmoid activation function. The mathematical representation of the sigmoid function is provided in Equation 21:

$$\text{Sigmoid (x)} = \frac{1}{1+e^{-x}} \tag{21}$$

Here, x denotes the input value, and e is Euler's mathematical constant, which has a value of 2.71828.

**Algorithm 3** CNN-BiLSTM Model

**Input:** $X \in \mathbb{R}^{T \times D}$ *(Input sequence with $T$ timesteps and $D$ features)*

Filters: $\mathbf{F}_1 \in \mathbb{R}^{k_1 \times k_1}$ (8 filters), $\mathbf{F}_2 \in \mathbb{R}^{k_2 \times k_2}$ (16 filters), $\mathbf{F}_3 \in \mathbb{R}^{k_3 \times k_3}$ (32 filters)

Pooling window size: 2

BiLSTM hidden state: $H_f, H_b$

Fully connected layer weight: $W_o \in \mathbb{R}^{d \times 1}$

**Initialization:**

Randomly initialize $\mathbf{F}_1, \mathbf{F}_2, \mathbf{F}_3$ and $W_o$.

**Convolution and Pooling Layers**

$$Z_1^{(f)} = \max(0, X * \mathbf{F}_1^{(f)}), \quad f = 1, \ldots, 8$$
$$P_1^{(f)} = \max_{i \in 2} Z_1^{(f)}$$
$$Z_2^{(f)} = \max(0, P_1 * \mathbf{F}_2^{(f)}), \quad f = 1, \ldots, 16$$
$$P_2^{(f)} = \max_{i \in 2} Z_2^{(f)}$$
$$Z_3^{(f)} = \max(0, P_2 * \mathbf{F}_3^{(f)}), \quad f = 1, \ldots, 32$$
$$P_3^{(f)} = \max_{i \in 2} Z_3^{(f)}$$

**BiLSTM Layer**

$$H_f = \text{LSTM}_f(P_3)$$
$$H_b = \text{LSTM}_b(P_3)$$
$$H = [H_f, H_b]$$

**Dense Layer:** $Y_{\text{Pred}} = \frac{1}{1+e^{-HW_o}}$

**Output:** $Y_{\text{Pred}} \leftarrow$ (cnn_bilstm_model.predict(X_test_lstm))

**Trust Propagation and Aggregation in Blockchain:** A transaction represents the behavior and network interaction

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



of any device. A transaction consists of the probability of the predicted value and the trust score. A transaction can be initiated by a smart meter, distribution gateway, or control center. Within an individual block, each transaction hash is calculated using the device ID, trust value, and prediction result. Once the hash values for all transactions are computed, the block hash is generated, as detailed in Algorithm 4.

**Hashing function:** The secure hashing function, denoted as $H(x)$, where $H : \{0,1\}^{256} \rightarrow \{0,1\}^{256}$, is a cryptographic function that takes a 256-bit input and returns a 256-bit output. The function produces a message digest $D$, defined as:

$$D = H(M) \tag{22}$$

where $M$ represents the input string passed to the hashing function. The hashing process is irreversible, satisfying the property:

$$H^{-1}(D) \text{ does not exist} \tag{23}$$

In GridTrust, the hash of each transaction within a block, denoted as $H_T$, is computed as:

$$H_T = H(T_x) \tag{24}$$

where $T_x$ represents an individual transaction within the block. The overall block hash $H_B$ is determined using the block ID $B_i$, timestamp $t_i$, Merkle root $R_M$, and the previous block hash $H_{B_{i-1}}$:

$$H_B = H(B_i \parallel t_i \parallel R_M \parallel H_{B_{i-1}}) \tag{25}$$

where $\parallel$ represents concatenation. The Merkle root $R_M$ is computed as:

$$R_M = H(H_T^1 \parallel H_T^2 \parallel \cdots \parallel H_T^n) \tag{26}$$

where $H_T^1, H_T^2, \ldots, H_T^n$ are the hashes of all transactions within the block.

Each block contains the hash of its predecessor, ensuring the integrity and immutability of the blockchain. The first block, termed the **genesis block**, lacks a predecessor, leading to:

$$H_{B_0} = \text{default} \tag{27}$$

The hashing function provides security by ensuring that any alteration in the data results in a significantly different hash value, satisfying the property:

$$\forall M_1 \neq M_2, \quad H(M_1) \neq H(M_2) \tag{28}$$

**Decision Phase:**

The blockchain executes the trust propagation and aggregation phase for the respective device. If a prior trust value exists for the device, the CNN-BiLSTM model overwrites it with the newly computed value and assigns a trust score to the corresponding device. Based on the predicted value and trust score ( Updated aggregated prediction $Y_{\text{Pred,agg}}$, Trust score $S_{\text{trust}}$), the device is categorized as either trustworthy or untrusted and is granted or restricted access to network communication.

The classification of a device as trusted or untrusted is based on the new predicted probability value and trust score (Updated aggregated prediction $Y_{\text{Pred,agg}}$, Trust score $S_{\text{trust}}$) generated by the CNN-BiLSTM model. If the predicted value exceeds 0.8, the device is categorized as untrusted, assigned a trust score of 0.2, and subsequently blocked from the network. Conversely, if the predicted value is below 0.5, the device is classified as trusted, assigned a trust score of 0.8, and grantedaccess to the network. The mathematical representation of this classification is given by:

$$S_{\text{trust}} = \begin{cases} 0.2, & \text{if } Y_{\text{Pred,agg}} > 0.8 \quad \text{(Untrusted, Blocked)} \\ 0.8, & \text{if } Y_{\text{Pred,agg}} < 0.5 \quad \text{(Trusted, Allowed)} \end{cases} \tag{29}$$

where: $S_{\text{trust}}$ represents the trust score assigned to the device.

**Algorithm 4 Blockchain Trust Propagation and Aggregation for Trust Prediction**

**Input:** New device prediction $Y_{\text{Pred,new}}$, Blockchain stored values $\mathcal{B}$, Trust level threshold $\tau$, Weight factor $\alpha$, Decay factor $\lambda$

**Output:** Updated aggregated prediction $Y_{\text{Pred,agg}}$, Trust score $S_{\text{trust}}$

**Initialization:** Retrieve stored aggregated prediction $Y_{\text{Pred,agg}}$, blockchain count $N$, and historical trust scores $\{S_{\text{trust},i}\}$ from blockchain $\mathcal{B}$;

**Exponential Weighted Aggregation:**
Compute time-decayed weight:
$W = e^{-\lambda N}$
Update weighted aggregated prediction:
$Y_{\text{Pred,agg}} \leftarrow (1-W)Y_{\text{Pred,agg}} + WY_{\text{Pred,new}}$
Update blockchain count: $N \leftarrow N+1$
Store updated $Y_{\text{Pred,agg}}$ into the blockchain $\mathcal{B}$;

**Deviation and Confidence-Based Trust Score:**
Compute prediction deviation:
$\Delta Y_{\text{Pred}} = |Y_{\text{Pred,new}} - Y_{\text{Pred,agg}}|$
Normalize deviation:
$D_{\text{norm}} = \frac{\Delta Y_{\text{Pred}}}{Y_{\text{Pred,agg}}}$
Compute trust confidence:
$\gamma = \frac{1}{1+e^{-N}}$
Compute trust score:
$S_{\text{trust}} = \max(0, 1 - \gamma D_{\text{norm}})$

**Previous Trust Score Adjustment:**
Compute moving average trust score:
$\overline{S_{\text{trust}}} = \frac{1}{N}\sum_{i=1}^{N} S_{\text{trust},i}$
Adjust trust score with history:
$S_{\text{trust}} \leftarrow \beta S_{\text{trust}} + (1-\beta)\overline{S_{\text{trust}}}$

**Trust Level Verification: if** $S_{trust} \geq \tau$ **then**
Prediction is trusted and considered valid; **else**
Mark prediction as untrusted and trigger re-evaluation;

**Blockchain Update:**
Store $(Y_{\text{Pred,agg}}, S_{\text{trust}}, N)$ into the blockchain $\mathcal{B}$;

**Output:**
Updated aggregated prediction $Y_{\text{Pred,agg}}$, Trust score $S_{\text{trust}}$

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



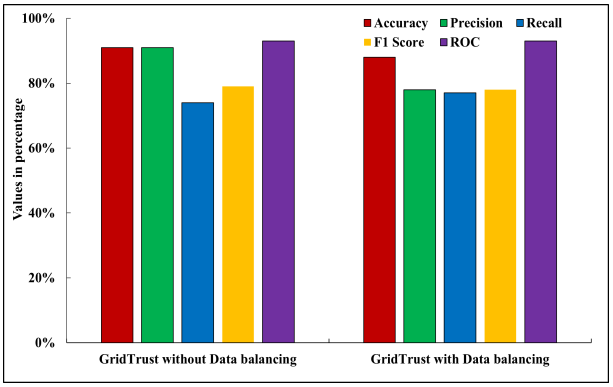


Figure 3. Model comparison with and without data balancing.

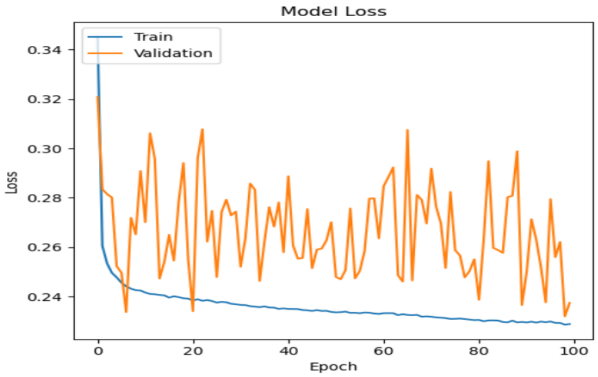


Figure 4. Loss of the proposed scheme after Data balancing.

$Y_{\text{Pred,agg}}$ is the predicted probability value from the model. If $Y_{\text{Pred,agg}} > 0.8$, the device is deemed untrusted and blocked. If $Y_{\text{Pred,agg}} < 0.5$, the device is considered trusted and permitted to communicate within the network. Both the trust score ($S_{\text{trust}}$) and the prediction probability ($Y_{\text{Pred,agg}}$) are securely stored in the blockchain for future reference and verification. This ensures a transparent and immutable record of device classifications within the smart grid network.

## IV. Performance Evaluation

In this section, we have evaluated the GridTrust model using machine learning performance metrics and blockchain indicators.

### A. Impact of Data Imbalance on GridTrust Performance

The ratio of untrusted nodes to trustworthy nodes is significantly higher. The accuracy of the proposed scheme without data balancing is higher compared to the balanced minority and majority classes due to the generation of false and biased results. The overall accuracy of GridTrust without data balancing is 91%. Precision, recall, and F1 score without data balancing are 91%, 74%, and 79%, respectively, see Figure 3. In contrast, the accuracy of GridTrust with data balancing is 88.30%, while the loss of the proposed scheme is 22.94% for training and 23.75% for validation testing after 100 epochs, as shown in Figure 4.

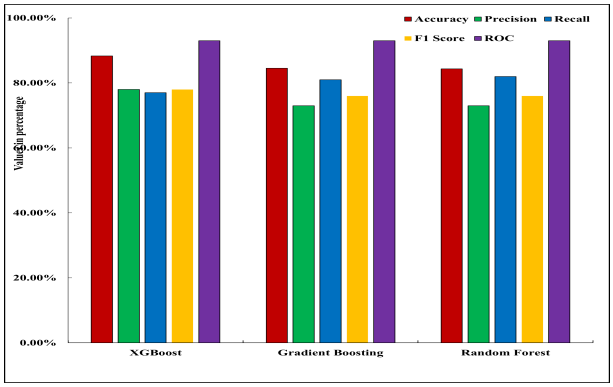


Figure 5. Comparison of various Feature selection schemes.

### B. Comparison of XGBoost with Other Feature Selection Schemes

To forensic the performance of our proposed GridTrust XGBoosting scheme against traditional feature selection methods, we compared XGBoost with random forest [15] and gradient boosting schemes [16]. The results indicate that XGBoost outperforms these methods in terms of precision, recall, F1 score, accuracy, and ROC. The overall accuracy of XGBoost is 88.30%, with precision, recall, F1 score, and ROC values of 78%, 77%, 78%, and 93%, respectively. In contrast, the accuracy of Random Forest and Gradient Boosting is 84.34% and 84.55%, respectively. The precision, recall, F1 score, and ROC for Random Forest are 73%, 82%, 76%, and 93%, while for Gradient Boosting, these values are 73%, 81%, 76%, and 93%, as illustrated in Figure 5.

### C. Transaction Per Second

TPS refers to the number of transactions GridTrust handle per second through the distribution gateway or control center. In this analysis, we examined the impact on TPS as the number of devices increases. A higher time duration indicates that GridTrust is underperforming.

To evaluate the system's efficiency and scalability, GridTrust assesses the proposed scheme by gradually increasing the number of devices. Initially, the TPS of a single device is measured, followed by assessments for 100, 300, and 500 devices. The upper limit of these tests is constrained by the computational resources available in our system. Observations indicate that the TPS for a single device is 0.8031 per second. When scaled to 100 devices, TPS decreases to 0.6916, whereas for 300 devices, it increases to 0.8675, and for 500 devices, it reaches 0.87322. These results suggest that GridTrust experiences a decline in TPS up to 100 devices, but beyond this point, the system performs efficiently in terms of scalability. After 100 devices, TPS shows a slight increase, which remains within an acceptable range. Figure 6 illustrates the average TPS for 100, 300, and 500 devices.

### D. Response Time

Average response time refers to the duration from the submission to the completion of a transaction. During this

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



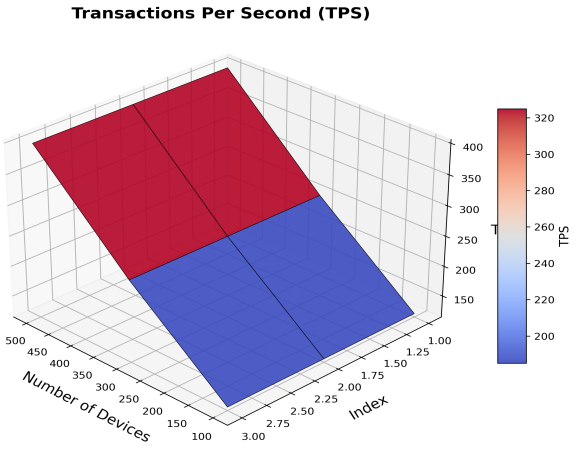


Figure 6. Execution of various transactions.

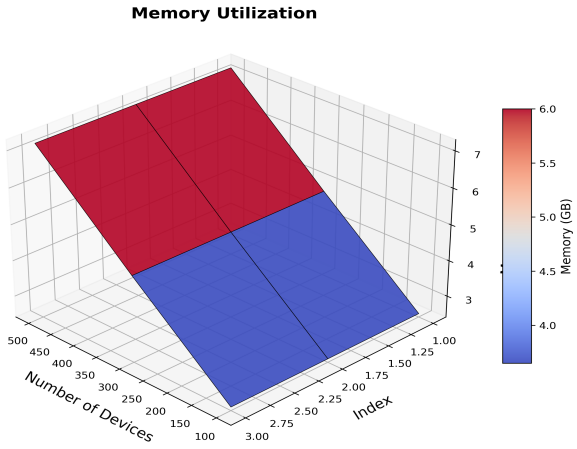


Figure 8. Resource utilization in context of memory.

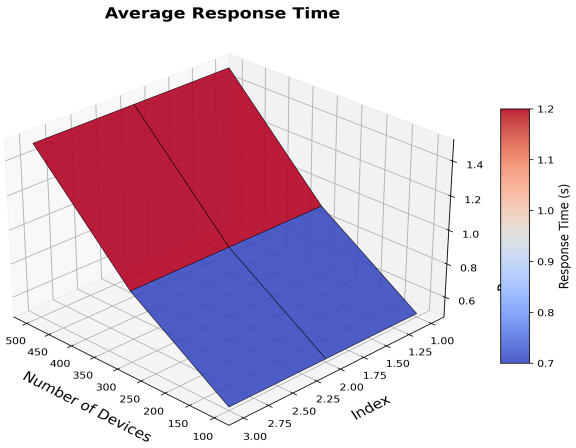


Figure 7. Average response time.

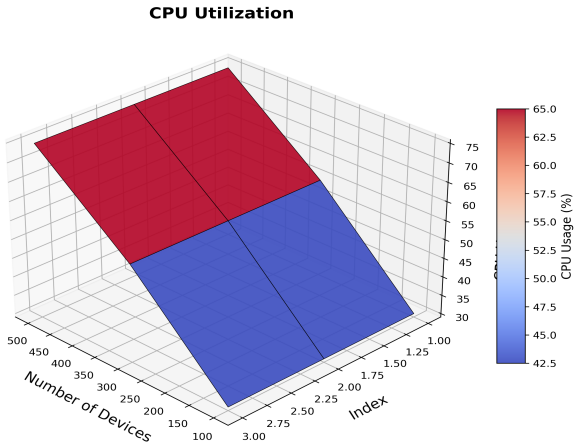


Figure 9. Resource utilization in context of CPU.

process, the transaction undergoes prediction, validation, block creation, addition, and consensus mechanisms. A lower average response time indicates higher system efficiency. However, efficiency also depends on the network's scalability. If the number of devices increases and the response time rises significantly, it suggests a lack of improvement in the model's performance.

To evaluate this aspect, we compared the GridTrust scheme under different network conditions with 100, 300, and 500 devices. The observed average response time for 100 devices is 0.7231 seconds. As the number of devices increases, the response time decreases, measuring 0.1467 seconds for 300 devices and 0.1384 seconds for 500 devices. This indicates that GridTrust maintains efficiency as scalability increases. Figure 7 illustrates the average response time across these scenarios.

### *E. Memory Allocation*

Average memory utilization refers to the total memory required for all transactions divided by the total number of transactions, representing the memory resources consumed by GridTrust during the prediction and storage processes. In GridTrust, memory usage is measured in bytes. If the system occupies a high amount of memory while handling a low number of transactions and devices, its performance is degraded. Conversely, if the system efficiently manages memory while processing a higher number of transactions and devices, it demonstrates optimal performance.

To evaluate memory utilization, we simulated the GridTrust model with 100, 300, and 500 devices. For a single device, memory utilization is recorded as 7.3 bytes while handling 0.83 transactions per second. For 100 devices, the average memory utilization is 6.89 bytes, decreasing to 6.39 bytes for 300 devices and 6.19 bytes for 500 devices. The results indicate that memory utilization is slightly higher at 100 devices but decreases as scalability increases. Specifically, memory usage reduces from 7.3 bytes to 6.39 bytes and 6.19 bytes for 300 and 500 devices, respectively. This optimization in memory consumption suggests that the GridTrust model is well-suited for large networks and high-load environments. Figure 8 illustrates the average memory utilization across different device configurations.

### *F. CPU Utilization*

To assess the system's performance under scalability and high workload conditions, we measured the average CPU utilization across different device configurations. As the number

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2025.3581699



of devices increases, CPU utilization either increases slightly or remains stable. The evaluation was conducted with device limits ranging from 100 to 500, as shown in Figure 9.

Specifically, CPU utilization for a single device is recorded at 6.209%. As the number of devices increases, the average CPU utilization decreases to 3.2% for 100 devices, 2.7% for 300 devices, and 2.2% for 500 devices. These results highlight the GridTrust model's ability to efficiently manage computational resources under increasing workloads.

## V. Conclusion

In the preprocessing phase, missing values and outliers are removed. Missing values are handled using a linear imputation method, while data normalization is performed using min-max scaling. The dataset is split in an 80:20 ratio, with 80% allocated for training and 20% for testing. Due to an imbalance where untrusted samples significantly outnumber trusted ones, the SMOTE is applied to equalize the classes. SMOTE generates 87,608 new samples for the minority class, resulting in a balanced dataset. After balancing the dataset, feature selection is performed using XGBoost, which forensically identifies the top 25 most important features out of 63 based on their assigned importance scores. For trust computation and model development, a CNN combined with a BiLSTM model is employed. The CNN-BiLSTM architecture consists of three convolutional and pooling layers, a BiLSTM layer, and an output layer. The model predicts a probability value and trust score to determine whether an entity should be allowed in the network.

Experimental results demonstrate that the GridTrust model excels in scalability and high-load scenarios, utilizing minimal computational resources while efficiently predicting intrusions. The proposed GridTrust framework introduces computational overhead due to the cryptographic puzzle-solving process and faces increased storage demands as the number of devices grows. In future work, we aim to evaluate our scheme using alternative consensus mechanisms such as Proof of Stake and Practical Byzantine Fault Tolerance to improve security while reducing computational latency.

## Acknowledgments

This work was partially funded by the Deanship of Scientific Research, King Saud University through Vice Deanship of Scientific Research Chairs: Chair of Cyber Security; and partially funded by Brazilian National Council for Scientific and Technological Development - CNPq, via Grant No. 306607/2023-9.

## References

[1] I. Baldwin, "Discovery of electricity and the electromagnetic force: Its importance for environmentalists, educators, physicians, politicians, and citizens," *Advances in Social Sciences Research Journal*, vol. 7, no. 12, pp. 362–383, 2020.

[2] R. W. Baloh, "Electricity and magnetism: Two sides of the same coin," in *Brain Electricity: The Interwoven History of Electricity and Neuroscience*. Springer, 2024, pp. 93–123.

[3] A. M. Shaheen, R. A. El-Sehiemy, H. M. Hasanien, and A. Ginidi, "An enhanced optimizer of social network search for multi-dimension optimal power flow in electrical power grids," *International Journal of Electrical Power & Energy Systems*, vol. 155, p. 109572, 2024.

[4] G.-F. Fan, Y.-Y. Han, J.-W. Li, L.-L. Peng, Y.-H. Yeh, and W.-C. Hong, "A hybrid model for deep learning short-term power load forecasting based on feature extraction statistics techniques," *Expert Systems with Applications*, vol. 238, p. 122012, 2024.

[5] P. Jokar, N. Arianpoo, and V. C. Leung, "Electricity theft detection in ami using customers' consumption patterns," *IEEE Transactions on Smart Grid*, vol. 7, no. 1, pp. 216–226, 2015.

[6] K. M. Ghori, M. Imran, A. Nawaz, R. A. Abbasi, A. Ullah, and L. Szathmary, "Performance analysis of machine learning classifiers for non-technical loss detection," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16, 2023.

[7] L. J. Lepolesa, S. Achari, and L. Cheng, "Electricity theft detection in smart grids based on deep neural network," *Ieee Access*, vol. 10, pp. 39 638–39 655, 2022.

[8] N. Ding, H. Ma, H. Gao, Y. Ma, and G. Tan, "Real-time anomaly detection based on long short-term memory and gaussian mixture model," *Computers & Electrical Engineering*, vol. 79, p. 106458, 2019.

[9] Z. Hussain, S. Memon, R. Shah, Z. A. Bhutto, and M. Aljawarneh, "Methods and techniques of electricity thieving in pakistan," *Journal of Power and Energy Engineering*, vol. 4, no. 9, pp. 1–10, 2016.

[10] M. F. Ayub, X. Li, K. Mahmood, S. Shamshad, M. A. Saleem, and M. Omar, "Secure consumer-centric demand response management in resilient smart grid as industry 5.0 application with blockchain-based authentication," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 1370–1379, 2023.

[11] M. Z. Gunduz and R. Das, "Smart grid security: An effective hybrid cnn-based approach for detecting energy theft using consumption patterns," *Sensors*, vol. 24, no. 4, p. 1148, 2024.

[12] A. Kumar, G. Rathee, C. A. Kerrache, M. Bilal, and T. R. Gadekallu, "A secure architectural model using blockchain and estimated trust mechanism in electronic consumers," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 4, pp. 996–1004, 2023.

[13] A. Mohammadali and M. S. Haghighi, "A privacy-preserving homomorphic scheme with multiple dimensions and fault tolerance for metering data aggregation in smart grid," *IEEE Transactions on Smart Grid*, vol. 12, no. 6, pp. 5212–5220, 2021.

[14] M. Krishnamoorthy and J. R. Albert, "Electricity theft detection in iot-based smart grids using a parameter-tuned bidirectional lstm with pre-trained feature learning mechanism," *Electrical Engineering*, vol. 106, no. 5, pp. 5987–6001, 2024.

[15] Y. Xuan, W. Si, J. Zhu, Z. Sun, J. Zhao, M. Xu, and S. Xu, "Multi-model fusion short-term load forecasting based on random forest feature selection and hybrid neural network," *Ieee Access*, vol. 9, pp. 69 002–69 009, 2021.

[16] D. Upadhyay, J. Manero, M. Zaman, and S. Sampalli, "Gradient boosting feature selection with machine learning classifiers for intrusion detection on power grids," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 1104–1116, 2020.

[17] Y. Alghofaili and M. A. Rassam, "A trust management model for iot devices and services based on the multi-criteria decision-making approach and deep long short-term memory technique," *Sensors*, vol. 22, no. 2, p. 634, 2022.

[18] ——, "A dynamic trust-related attack detection model for iot devices and services based on the deep long short-term memory technique," *Sensors*, vol. 23, no. 8, p. 3814, 2023.

[19] I. Siniosoglou, P. Radoglou-Grammatikis, G. Efstathopoulos, P. Fouliras, and P. Sarigiannidis, "A unified deep learning anomaly detection and classification approach for smart grid environments," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1137–1151, 2021.

[20] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning," *IEEe Access*, vol. 10, pp. 40 281–40 306, 2022.

[21] P. Verma, J. G. Breslin, D. O'Shea, N. Mehta, N. Bharot, and A. Vidyarthi, "Leveraging gametic heredity in oversampling techniques to handle class imbalance for efficient cyberthreat detection in iiot," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 1940–1951, 2023.

[22] K. Abbas, A. Nauman, M. Bilal, J.-H. Yoo, J. W.-K. Hong, and W.-C. Song, "Ai-driven data analytics and intent-based networking for orchestration and control of b5g consumer electronics services," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 2155–2169, 2023.

Authorized licensed use limited to: Centro Universitário Fei. Downloaded on June 23,2025 at 19:07:24 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.