AIMS Mathematics, 11(9): 28801–28828.
DOI: 10.3934/math.20261146
Received: 13 April 2026
Revised: 20 July 2026
Accepted: 24 July 2026
Published: 08 September 2026

https://www.aimspress.com/journal/Math

Research article

DPFT: A blockchain-enabled differentially private federated transformer for trust evaluation in vehicular networks

Wajahat Ali[1], Fahd Raza[1], Arshad Iqbal[1,*], Abdul Wadood[2,3], Herie Park[4,5,*], Hani Albalawi[2,3] and Byung O Kang[4,*]

[1] School of Computing Sciences, Pak-Austria Fachhochschule, Haripur 22620, Pakistan
[2] Zero Emission Technologies Innovation Center, University of Tabuk, Tabuk, Saudi Arabia
[3] Department of Electrical Engineering, Faculty of Engineering, University of Tabuk, Saudi Arabia
[4] Department of Electrical Engineering, Dong-A University, Busan, Republic of Korea
[5] Department of ICT Integrated Safe Ocean Smart Cities Engineering, Dong-A University, Busan, Republic of Korea

* **Correspondence:** Emails: arshad.iqbal@spcai.paf-iast.edu.pk; bakery@donga.ac.kr; bokang@dau.ac.kr.

Abstract: Intelligent transportation systems (ITS) rely on distributed data generated by connected vehicles, roadside infrastructure, and edge sensors. However, malicious participants may inject unreliable information that threatens transportation safety and operational stability. This study proposes a differentially private federated transformer (DPFT) framework for privacy-preserving node trust classification in vehicular networks. The framework combines Swin transformer-based feature learning, federated training, gradient clipping, Gaussian-noise-based differential privacy, and trimmed mean aggregation. A private blockchain records the resulting trust scores to provide tamper-resistant auditing and decentralized accountability. Experimental results show that DPFT achieves an area under the receiver operating characteristic curve (AUROC) of 0.90, an F1 score of 0.77, and more than 78% classification accuracy. It outperforms FedAvg-long short-term memory (LSTM) and FedAvg-Trans and maintains over 70% accuracy when 25% of the participating nodes are Byzantine. The blockchain component processes 55 transactions per second with a latency of 248 milliseconds, demonstrating practical scalability under the evaluated federated simulation setting.

Keywords: blockchain; differential privacy; federated learning; Swin transformer; vehicular networks; trust evaluation
Mathematics Subject Classification: 68P27, 68T07, 68W15

1. Introduction

Intelligent transportation systems are rapidly evolving into hyperconnected, cyber-physical infrastructures that rely on a continuous stream of data from vehicles, roadside units, traffic sensors, and external data providers [1]. These systems enable a range of safety-critical and efficiency-enhancing applications, including autonomous driving, collision avoidance, and adaptive traffic signal control [2, 3]. However, as these systems become more reliant on distributed, multisource data, they are increasingly vulnerable to adversarial behaviors and malicious injections that can compromise their integrity, security, and trustworthiness [4].

The inherent heterogeneity and decentralized nature of intelligent transportation systems (ITS) nodes raise unique challenges in identifying which data sources are reliable in real time [5]. Conventional trust evaluation mechanisms often rely on fixed thresholds, predefined scoring models, or central data fusion schemes that are neither scalable nor resilient to coordinated strategic adversarial attacks [6–8]. Furthermore, these approaches often fail to build trust and preserve user privacy, a critical requirement in the transportation domain, where behavioral and locational data are sensitive [9–11]. The evaluation of trust should not be limited to a classification problem in the field of machine learning for ITS. It also relates to the overall problem of information flow reliability in complex sociotechnical and networked systems. For infrastructures that are vulnerable to crises, erroneous or delayed trust decisions can impact safety, coordination, and the effectiveness of decision-making. It is recognized by complexity science and information-system perspectives that distributed infrastructures need reliable and trusted sensing, communication, and timely data validation to make safety-critical decisions [12].

Recent advances in federated learning and privacy-preserving computation provide promising directions to address these limitations [13]. Federated learning enables multiple edge nodes to train a shared model without centralizing their raw data. This reduces the risk of privacy leakage [14, 15]. Similarly, differential privacy adds strict mathematical conditions to deter inference attacks. Moreover, neural architectures based on transformers have proven to be better at learning complex dependencies in sequential and tabular data and hence are well-suited to the modeling of rich streams of behavioral telemetry patterns with telemetry data at hand to identify malicious and honest nodes [16]. The main contributions of this work are summarized as follows:

- We use a Swin transformer-based client-level representation and trust classification.
- We introduce a federated trust-evaluation framework for supporting trusted and malicious classification at a client level.
- We employ differential privacy to the training process directly by gradient clipping and injecting Gaussian noise into the training process to minimize the risk of leakage of private data from the model update sharing.
- We use trimmed mean aggregation for robustness against Byzantine or poisoned client updates when applying federated learning.
- We integrate a private blockchain-based trust ledger to store trust scores of the vehicles in a tamper-proof and auditable way.

These modules work together in a chain of sequential operations to transform local observations into privacy-preserving model updates, securely aggregated into a trust model in a blockchain, before generating auditable trust records in a blockchain.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 presents the system model. Section 4 introduces the proposed differentially private federated transformer (DPFT) framework and describes its major components. Section 5 describes the differential privacy, and Section 6 presents the experimental setup and evaluates the framework's effectiveness, scalability, robustness, and privacy-preserving capabilities. Finally, Section 7 concludes the paper and outlines directions for future research.

2. Related work

Trust management in intelligent and connected vehicular networks has been extensively studied over the past decade, particularly in the context of identifying data reliability and mitigating misinformation propagation in distributed environments [17, 18]. Early works predominantly utilized rule-based or reputation-based scoring mechanisms to estimate trustworthiness [19]. However, the study of trust and reputation mechanisms has also been extended to more complex networked systems in the broader literature. Adaptive reputation models have demonstrated how trust can be reinforced continuously with respect to observed behaviors and interactions in the network [20]. These types of methods can be helpful in comprehending the mechanisms underlying the generation of trust based on past behavior. Most models of reputation do not, however, directly consider privacy preserving federated training, Byzantine client actions, or tamper-proof trust histories [21]. Such heuristic approaches suffer from a lack of adaptability to novel attack patterns and fail to scale with the high dimensionality of modern vehicular sensor data.

In order to address these limitations, recent studies have investigated deep learning methods to identify nonlinear relationships and minor deviations in behaviors [22, 23]. Convolutional neural networks (CNNs) and long short-term memory (LSTM) networks are some of the models that have shown potential in anomaly detection, especially sequence-based sensor telemetry [24, 25]. The majority of such models are centrally trained, however, subjecting the system to privacy violations and failure points.

Federated learning has also become a privacy-conscious solution to allow distributed model training without raw data aggregation. Reference materials such as [26] on communication presented the basic federated averaging algorithm, and others further applied it to vehicular edge computing setting and scenarios [27]. Nevertheless, few such methods involve adversarial resilience or formal differential privacy.

The application of transformer architectures for trust modeling is relatively unexplored. Although transformers have revolutionized natural language processing and time-series prediction tasks [28], their integration with federated learning and differential privacy in the ITS domain remains a significant research gap. In Table 1, we summarize recent literature addressing trust evaluation, and privacy in ITS. For each study, we list the core contribution and highlight its limitations.

Despite these advancements, no existing framework combines transformer-based behavior modeling, formal differential privacy, and blockchain for trust classification within the ITS ecosystem. This paper addresses these gaps by proposing a framework that integrates these components for privacy-preserving and adversarially robust trust evaluation under simulated vehicular network conditions.

Table 1. Comparative analysis of recent work.

Reference	Contribution	Limitations
[19]	Rule-based or score-based trust evaluation based on direct interactions	Static models; lack scalability and adaptability to new threats
[21]	Nonlinear behavior modeling using deep learning	Requires centralized data; privacy risks remain high
[26]	Federated learning (FL) for distributed model training	No adversarial modeling or trust evaluation
[27]	FL for vehicular networks with communication efficiency	Ignores gradient poisoning or data integrity threats
[28]	Transformer architecture for attention-based learning	No application to trust scoring or federated ITS environments

3. System model

The proposed DPFT framework consists of a distributed set of connected vehicles $\mathcal{V}$ and roadside units $\mathcal{R}$ that collaboratively participate in a privacy-preserving federated learning environment. As illustrated in Figure 1, each camera-based client locally preprocesses its vehicle reidentification (VeRi) observations and applies the current global Swin transformer model to extract visual representations and generate a client-level trust prediction. The corresponding local training gradient is then clipped and perturbed with calibrated Gaussian noise before transmission, ensuring (ε, δ) differential privacy while keeping the raw images and sensitive data locally stored at each client.

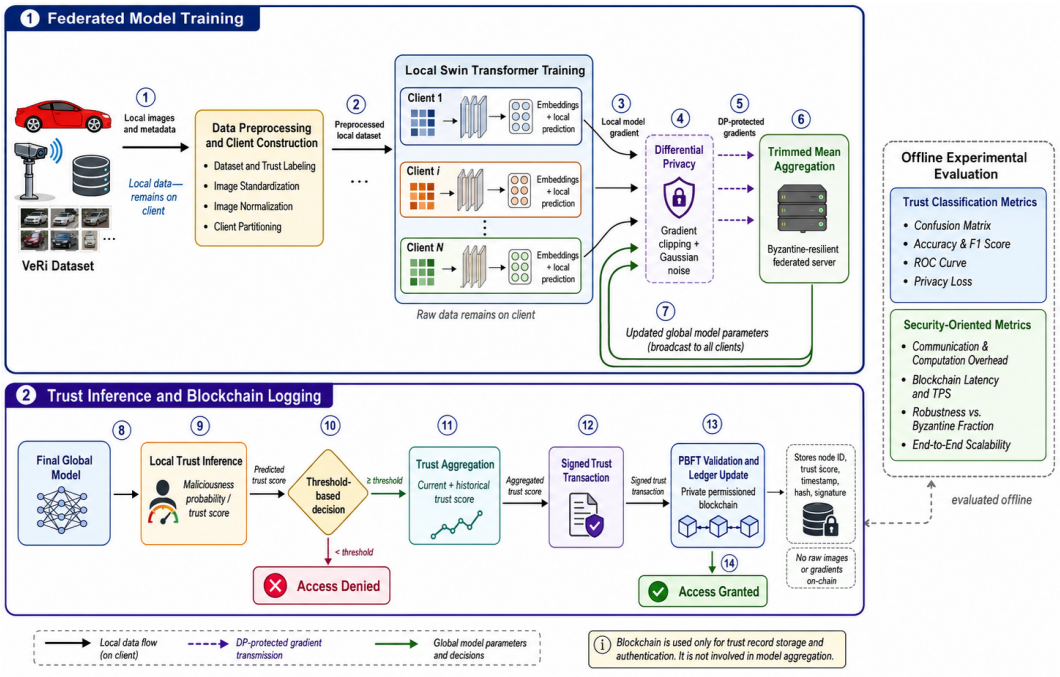

Figure 1. Proposed DPFT framework during FL, client-level trust inference, and blockchain-based trust-record management.

The protected gradients are transmitted to the federated coordinator $\mathcal{S}$, which applies Byzantine-resilient trimmed mean aggregation to reduce the influence of malicious or poisoned updates. The resulting global model is redistributed to the participating clients, and this process continues until convergence. During inference, the final global model produces a trust score for each participating node. A smart contract combines the newly predicted trust score with its historical trust value and submits the resulting trust record to a private permissioned blockchain. Authorized validators use practical Byzantine fault tolerance (PBFT) consensus to verify and record the transaction, providing tamper-resistant storage, decentralized accountability, and transparent auditing of node behavior within the ITS environment.

4. Proposed DPFT scheme

The proposed DPFT architecture consists of five key components: data preprocessing and client construction, learning on local trust, generation of differentially private model updates, Byzantine-resilient federated aggregation, and trust-record management on blockchain. The first four modules make up the federated training and inference pipeline, and the blockchain module is part of the after-inference component to ensure that trust decisions are captured in a secure and auditable history.

4.1. Data preprocessing

4.1.1. Dataset and trust labels construction

To evaluate the proposed framework, we utilize the publicly available VeRi dataset, which is widely used for vehicle analysis and intelligent transportation applications [29]. The VeRi dataset contains 49,357 vehicle images collected from 20 surveillance cameras deployed in a real-world urban traffic environment. The dataset includes 776 unique vehicle identities observed across multiple viewpoints and timestamps; see Table 2.

Table 2. VeRi dataset description.

Description	Value
Total vehicles	776
Total images	49,357
Number of cameras	20
Training samples	37,778
Testing samples	11,579

Each image sample is associated with annotations including vehicle ID, camera ID, and capture time, enabling the analysis of vehicle activity patterns across distributed monitoring points. However, the VeRi dataset does not explicitly provide labels indicating whether a vehicular node, camera node, or federated client is trusted or malicious. In this work, the annotations (VeRi) are not used as trust labels. On the contrary, VeRi is leveraged as a realistic observation source for vehicles, and the trusted/malicious labels are induced via a controlled federated attack simulation. The original VeRi data set is denoted by

$$D = \{(I_i, v_i, c_i, t_i)\}_{i=1}^{N}. \tag{4.1}$$

Here, I_i represents the image of the ith vehicle, v_i represents the identity label of the original vehicle, c_i represents the camera identifier, and t_i represents the timestamp. The label v_i is not intended to be

used as the trust label and is only for the vehicle's original annotation. The individual cameras and also roadside observation points are considered federated clients. Thus, the Local dataset of client k is given by

$$D_k = \{(I_i, v_i, c_i, t_i) \in D \mid c_i = k\}. \tag{4.2}$$

VeRi does not have any malicious-node labels, so malicious behavior is simulated at the client level. The Byzantine ratio is denoted by ρ_B, and at every communication round, a subset of clients is chosen to be malicious, according to the given ratio ρ_B. The rest of the clients are considered as trusted clients. The client-level trust label is as defined below:

$$y_k^{trust} = \begin{cases} 0, & \text{if client } k \text{ is trusted,} \\ 1, & \text{if client } k \text{ is malicious.} \end{cases} \tag{4.3}$$

Here, y_k^{trust} represents the experimentally obtained ground truth trust label. The trusted client trains on clean data; the malicious client manipulates labels/images/updates before sending gradients. The following two forms of malicious behavior are taken into consideration in this study; the first attack is when a malicious client flips a subset of its local training labels. The original vehicle identity label is denoted by v_i, and the manipulated label is denoted by v_i'.

$$\tilde{v}_i = \mathcal{F}(v_i), \tag{4.4}$$

where the function $\mathcal{F}(\cdot)$ is the flipping of the labels. This attack mimics the faulty semantic information provided in local training. Second, in the data-corruption attack, the malicious client corrupts its locally stored vehicle images prior to training. The corrupted image is

$$\tilde{I}_i = I_i + \alpha\xi_i. \tag{4.5}$$

Here, ξ_i denotes random perturbation added to image I_i, and α controls the corruption intensity. The attack mimics noisy sensors, visual sensor spoofing, or attacks on visual sensors in vehicular infrastructure. The vehicular observations are given in the original VeRi dataset, and the trusted and malicious classes are created by mimicking honest and adversarial client behavior in the federated learning setting. The model is assessed based on the comparison between the predicted trust decision by each client and simulated trust label y_k^{trust}.

Image standardization. I_i denotes the ith vehicle image captured by a surveillance camera. All images are resized to a fixed spatial resolution to ensure consistent input dimensions for the feature extraction network [30]. The resized image is represented as

$$I_i^{(r)} = \text{Resize}(I_i, H, W), \tag{4.6}$$

where H and W denote the target image height and width, respectively. The data preprocessing and trust-label construction for the VeRi dataset is shown in Algorithm 1.

Image normalization. To reduce illumination variation and improve training stability, pixel intensities are normalized using channelwise mean and standard deviation [31]. The normalized image $\tilde{I}_i$ is computed as

$$\tilde{I}_i = \frac{I_i^{(r)} - \mu}{\sigma}, \tag{4.7}$$

where μ and σ represent the mean and standard deviation of pixel values computed across the dataset.

Algorithm 1: Data preprocessing and trust-label construction for the VeRi dataset

Input: Raw VeRi dataset $\mathcal{D}^{raw} = \{(I_i, v_i, c_i, t_i)\}_{i=1}^{N}$, target image size (H, W), number of federated clients K, and Byzantine ratio ρ_B

Output: Normalized and client-partitioned datasets $\{\mathcal{D}_k\}_{k=1}^{K}$ and client-level trust labels $\{y_k^{trust}\}_{k=1}^{K}$

Initialize the preprocessed dataset: $\mathcal{D} \leftarrow \emptyset$

Compute the channelwise mean μ and standard deviation σ from the training data

for *each sample* $(I_i, v_i, c_i, t_i) \in \mathcal{D}^{raw}$ **do**

Resize the image: $I_i^{(r)} \leftarrow \text{Resize}(I_i, H, W)$

Normalize the resized image: $\tilde{I}_i \leftarrow \frac{I_i^{(r)} - \mu}{\sigma}$

if $\tilde{I}_i$ *contains missing or invalid pixel values* **then**

Apply interpolation or discard the sample **continue**

Apply optional training-time augmentation, such as flipping, rotation, or scaling

Append the normalized image and its original annotations:

$$\mathcal{D} \leftarrow \mathcal{D} \cup \{(\tilde{I}_i, v_i, c_i, t_i)\}$$

Partition $\mathcal{D}$ into K camera-based client datasets:

$$\mathcal{D}_k = \{(\tilde{I}_i, v_i, c_i, t_i) \in \mathcal{D} \mid c_i = k\}$$

Randomly select the malicious client set $\mathcal{M} \subseteq \{1, \ldots, K\}$ according to ρ_B

for *each client* $k = 1$ *to* K **do**

if $k \in \mathcal{M}$ **then**

Assign the malicious client-level label: $y_k^{trust} \leftarrow 1$

Apply a selected data-level attack to $\mathcal{D}_k$, such as label flipping or image corruption

else

Assign the trusted client-level label: $y_k^{trust} \leftarrow 0$

Keep $\mathcal{D}_k$ unchanged

return $\{\mathcal{D}_k\}_{k=1}^{K}$ and $\{y_k^{trust}\}_{k=1}^{K}$

Feature extraction. After normalization, each image is passed through the Swin transformer encoder used in the proposed DPFT framework. The Swin transformer extracts hierarchical visual representations using shifted-window self-attention instead of a conventional convolutional feature extractor. The visual feature embedding is defined as

$$z_i = f_\theta^{\text{Swin}}(\tilde{I}_i), \tag{4.8}$$

where $f_\theta^{\text{Swin}}(\cdot)$ denotes the Swin transformer encoder with trainable parameters θ, and $\mathbf{z}_i \in R^d$ represents the learned feature embedding.

Feature vector construction. The extracted embeddings are organized into structured feature vectors representing node observations within the ITS environment. Each processed sample is represented as

$$x_i = [z_{i1}, z_{i2}, \ldots, z_{id}]^{\mathsf{T}} \in R^d, \tag{4.9}$$

where d denotes the dimensionality of the learned visual representation. These feature vectors serve as the input to the transformer-based trust inference model.

Data partitioning. The preprocessed VeRi samples are partitioned into local client datasets for FL. The original vehicle identity label is denoted by v_i, and the client-level trust label is denoted by y_k^{trust}. These two labels have different meanings. The label v_i belongs to the original VeRi annotation, whereas y_k^{trust} is generated through controlled attack simulation. Each local client dataset is represented as

$$\mathcal{D}_k = \{(\mathbf{x}_i, v_i, c_i, t_i)\}_{i=1}^{n_k}, \tag{4.10}$$

where n_k is the number of local samples assigned to client k.

4.2. Local trust evaluation via Swin transformer

Each participating node i processes locally observed vehicle images collected from surveillance cameras. After preprocessing, each image sample is represented as a normalized visual input $\tilde{I}_i$. Instead of using conventional convolutional models, we employ a Swin transformer encoder to extract hierarchical visual representations that capture spatial relationships among vehicle appearance patterns.

Formally, the Swin transformer divides the input image into nonoverlapping patches and computes feature embeddings through shifted window-based self-attention. $f_\theta^{\text{Swin}}(\cdot)$ denote the Swin transformer encoder with parameters θ. The differentially private federated Swin transformer for trust classification is shown in Algorithm 2. The feature representation $\mathbf{z}_i$ of image $\tilde{I}_i$ is computed as

$$z_i = f_\theta^{\text{Swin}}(\tilde{I}_i), \quad \mathbf{z}_i \in \mathbb{R}^d, \tag{4.11}$$

where d represents the embedding dimension of the learned visual feature vector. For client-level trust prediction, the local feature representations are aggregated to obtain a client behavior embedding h_k. The trust score of client k is then computed as

$$\hat{y}_k^{trust} = \sigma(Wh_k + b), \tag{4.12}$$

where $\hat{y}_k^{trust} \in (0, 1)$ indicates the predicted probability that client k is malicious. The client-level representation h_k is obtained by aggregating the Swin transformer embeddings of the local observations of client k:

$$h_k = \frac{1}{|D_k|} \sum_{I_i \in D_k} f_\theta^{Swin}(\tilde{I}_i). \tag{4.13}$$

To train the model, binary cross-entropy loss is employed:

$$\mathcal{L}_k = -y_k^{trust} \log(\hat{y}_k^{trust}) - (1 - y_k^{trust}) \log(1 - \hat{y}_k^{trust}), \tag{4.14}$$

where $y_k^{trust} \in \{0, 1\}$ is the generated ground-truth trust label of client k, and $\hat{y}_k^{trust}$ is the predicted probability that client k is malicious:

$$\Delta\theta_k = \nabla_\theta \mathcal{L}_k. \tag{4.15}$$

To ensure (ε, δ)-differential privacy, calibrated Gaussian noise is injected before transmitting the update to the server:

$$\widetilde{\Delta\theta_k} = \Delta\theta_k + \mathcal{N}(0, \sigma^2 C^2 I). \tag{4.16}$$

Algorithm 2: Differentially private federated Swin transformer for trust classification

Input: Client datasets $\{\mathcal{D}_k\}_{k=1}^{K}$, labels $\{y_k^{trust}\}_{k=1}^{K}$, malicious set $\mathcal{M}$, rounds T, learning rate η, initial model Θ_0, clipping norm C, noise multiplier σ, aggregation rule $\mathcal{A}$, threshold θ_{mal}, tolerance τ

Output: Final model Θ_{final}, trust scores $\{\hat{T}_k\}_{k=1}^{K}$, and client decisions

Initialize $t \leftarrow 0$ and $\Theta_t \leftarrow \Theta_0$;

Broadcast Θ_0 to all clients;

while $t < T$ **do**

  Select participating clients $\mathcal{S}_t \subseteq \{1, \ldots, K\}$;

  foreach *client* $k \in \mathcal{S}_t$ *in parallel* **do**

    Sample mini-batch $\mathcal{B}_k \subseteq \mathcal{D}_k$;

    Extract embeddings $\mathbf{z}_{kj} \leftarrow f_{\Theta_t}^{\text{Swin}}(\tilde{I}_{kj})$;

    Compute $\mathbf{h}_k \leftarrow \frac{1}{|\mathcal{B}_k|} \sum_{\tilde{I}_{kj} \in \mathcal{B}_k} \mathbf{z}_{kj}$;

    Compute maliciousness probability $p_k^{mal} \leftarrow \text{sigmoid}(W\mathbf{h}_k + b)$;

    Compute $\mathcal{L}_k \leftarrow -y_k^{trust} \log(p_k^{mal}) - (1 - y_k^{trust}) \log(1 - p_k^{mal})$;

    Compute local gradient $\mathbf{g}_k \leftarrow \nabla_\Theta \mathcal{L}_k$;

    if $k \in \mathcal{M}$ *and Byzantine update manipulation is enabled* **then**

      $\mathbf{g}_k \leftarrow \text{Attack}(\mathbf{g}_k)$;

    end

    Clip gradient $\bar{\mathbf{g}}_k \leftarrow \frac{\mathbf{g}_k}{\max(1, \|\mathbf{g}_k\|_2 / C)}$;

    Add DP noise $\tilde{\mathbf{g}}_k \leftarrow \bar{\mathbf{g}}_k + \mathcal{N}(0, \sigma^2 C^2 I)$;

    Send only $\tilde{\mathbf{g}}_k$ to the federated server;

  end

  Collect protected gradients $\{\tilde{\mathbf{g}}_k\}_{k \in \mathcal{S}_t}$;

  Compute robust update $\Delta_t \leftarrow \mathcal{A}(\{\tilde{\mathbf{g}}_k\}_{k \in \mathcal{S}_t})$;

  Update global model $\Theta_{t+1} \leftarrow \Theta_t - \eta \Delta_t$;

  if $\|\Theta_{t+1} - \Theta_t\|_2 \leq \tau$ **then**

    $\Theta_{\text{final}} \leftarrow \Theta_{t+1}$; **break**;

  end

  Broadcast Θ_{t+1} to participating clients;

  $t \leftarrow t + 1$;

end

if $t = T$ **then**

  $\Theta_{\text{final}} \leftarrow \Theta_t$;

end

foreach *client* $k = 1$ *to* K **do**

  Compute p_k^{mal} using Θ_{final} and $\mathcal{D}_k$;

  Set $\hat{T}_k \leftarrow 1 - p_k^{mal}$ and classify client k as malicious if $p_k^{mal} \geq \theta_{\text{mal}}$; otherwise, classify it as trusted;

end

return Θ_{final}, $\{\hat{T}_k\}_{k=1}^{K}$, *and all client decisions*

The server $\mathcal{S}$ collects noisy updates from all participating nodes and constructs the gradient set

$$\mathcal{G}_t = \{\widetilde{\Delta\theta_1}, \widetilde{\Delta\theta_2}, \ldots, \widetilde{\Delta\theta_N}\}. \tag{4.17}$$

To mitigate the influence of malicious participants, the server applies a Byzantine-resilient aggregation rule $\mathcal{A}$, such as trimmed mean. The global model parameters are then updated as

$$\theta_{t+1} = \theta_t - \eta \cdot \mathcal{A}(\mathcal{G}_t), \tag{4.18}$$

where η denotes the learning rate. This iterative process enables collaborative model training across distributed ITS nodes while preserving data privacy and robustness against adversarial updates.

4.3. Byzantine-resilient aggregation module

In federated learning environments, some participating nodes may behave maliciously and attempt to poison the global model by sending manipulated gradients. Such participants are commonly referred to as Byzantine clients. To mitigate the influence of adversarial updates, the proposed DPFT framework employs a Byzantine-resilient aggregation strategy based on the trimmed mean rule.

$\mathcal{G}_t = \{\tilde{g}_1, \tilde{g}_2, \ldots, \tilde{g}_N\}$ denotes the set of noisy gradients received from N participating nodes at communication round t, where $\tilde{g}_i \in \mathbb{R}^d$ represents the gradient vector from node i. The trimmed mean operator removes extreme values in each coordinate dimension before computing the average update.

Formally, for the jth coordinate, the gradients are first sorted, and the largest and smallest B values are discarded. The aggregated value is then computed as

$$\mu_j = \frac{1}{N - 2B} \sum_{k=B+1}^{N-B} g_{(k)}^{(j)}, \tag{4.19}$$

where $g_{(k)}^{(j)}$ denotes the kth ordered gradient value in the jth dimension. The final aggregated update vector is defined as

$$\Delta_t = [\mu_1, \mu_2, \ldots, \mu_d]^{\mathsf{T}}. \tag{4.20}$$

This mechanism effectively filters out abnormal updates originating from malicious nodes while preserving contributions from honest participants. The robust update Δ_t is then used by the server to update the global model parameters; see Algorithm 3.

Trimmed-mean robustness. The gradients of honest clients are bounded by $\|\tilde{g}_i\|_2 \leq C$, and the number of Byzantine participants satisfies $B < N/4$. Then, the trimmed mean aggregation produces an estimate whose deviation from the mean of the honest gradients is bounded as

$$\left\| \Delta_t - \frac{1}{|\mathcal{H}|} \sum_{h \in \mathcal{H}} \tilde{g}_h \right\|_2 = O\left(\frac{CB}{N}\right), \tag{4.21}$$

where $\mathcal{H}$ denotes the set of honest clients.

Real-time inference logic: All participating nodes receive the final global model parameters θ^{final} after federated training coverages. The decision threshold θ_{dec} is determined using the optimal operating point of the receiver operating characteristic (ROC) curve obtained on the validation set. If the predicted score exceeds the threshold, the node is classified as malicious; otherwise, it is considered

trustworthy. Because inference is performed locally at each node, the system can scale efficiently to large ITS deployments without introducing centralized latency.

Algorithm 3: Trimmed mean Byzantine-resilient aggregation

Input: Gradient set $\mathcal{G}_t = \{\tilde{g}_1, \tilde{g}_2, \ldots, \tilde{g}_N\}$ received from N clients; Byzantine tolerance B.
Output: Robust aggregated update $\Delta_t \in \mathbb{R}^d$.
$d \leftarrow$ dimension of gradient vectors
Initialize $\Delta_t \leftarrow \mathbf{0}_d$
for $j \leftarrow 1$ **to** d **do**
  Extract coordinate values:

$$C_j = [\tilde{g}_1[j], \tilde{g}_2[j], \ldots, \tilde{g}_N[j]]$$

  Sort C_j in ascending order
  Remove the B smallest and B largest elements:

$$C_j^{trim} = C_j[B : N - B]$$

  Compute trimmed mean:

$$\mu_j = \frac{1}{|C_j^{trim}|} \sum_{x \in C_j^{trim}} x$$

  Set $\Delta_t[j] \leftarrow \mu_j$
end
return Δ_t

4.4. Blockchain-based trust aggregation and secure storage

The blockchain layer is used to store trust records in a verifiable, tamper-proof, and decentralized manner. A replicated database or a signed audit log can be used to store the trust score, but this typically relies on a trusted administrator and/or a centralized authority to manage the database. The proposed private blockchain, on the other hand, distributes the validation among authorized validators and stops any single compromised entity from making a secret alteration to historical trust records.

The proposed DPFT framework does not store vehicles images, local datasets, or model gradients on the blockchain. The blockchain stores only trust-related information: node ID, aggregated trust score, previous block hash, timestamp, and digital signature. This design not only reduces storage overhead but also helps maintain auditability and accountability. Thus, the blockchain layer is not applied for the transmission of raw data, but the trust-record management is applied for secure management. The blockchain network is a private permissioned network composed of trusted validators, for example, trusted roadside infrastructure units or edge servers. PBFT is used as the consensus protocol. When assuming the PBFT, a network of n validators can withstand up to f Byzantine validators if $n \geq 3f + 1$. The five validators used in the implemented configuration provide a degree of fault-tolerance against up to one faulty or malicious validator with still low confirmations latency; see Table 3.

Table 3. Blockchain configuration and threat model.

Component	Description
Blockchain type	Private permissioned blockchain
Consensus protocol	PBFT
Validators	Authorized roadside units or trusted edge servers
Number of validators	5
Byzantine tolerance	Up to one faulty or malicious validator
Stored data	Node ID, trust score, timestamp, previous hash, and signature
Raw data storage	Raw images, local datasets, and gradients are not stored on-chain
Threat model	Malicious federated clients and limited malicious validators
Security goal	Tamper-resistant and auditable trust-score logging
Latency	248 ms average block-confirmation delay
Throughput	55 TPS under the evaluated private network setting

After computing the real-time trust score of each participating node, denoted by $\hat{T}_i^{(t)}$, the proposed framework records the trust information in a private blockchain-enabled ledger. The blockchain layer ensures that trust records are tamper-resistant, auditable, and decentralized across the ITS infrastructure. This prevents malicious entities from manipulating trust histories and provides transparent verification of node behavior. To maintain a consistent trust history, a smart contract C_{trust} aggregates the newly predicted trust score with the previous trust value stored on the blockchain. The aggregated trust value is computed using a weighted update mechanism

$$T_i^{(t)} = \lambda T_i^{(t-1)} + (1-\lambda)\hat{T}_i^{(t)}, \tag{4.22}$$

where $T_i^{(t-1)}$ denotes the historical trust score recorded in the blockchain ledger, $\hat{T}_i^{(t)}$ is the newly predicted trust score obtained from the federated Swin transformer model, and $\lambda \in [0,1]$ is a weighting factor that balances past behavior and current observations. If the aggregated trust value $T_i^{(t)}$ falls below a predefined decision threshold θ_{dec}, the corresponding node is flagged as malicious, and its participation in the ITS communication network is restricted. Otherwise, the node continues operating as a trusted participant. The complete blockchain trust recording procedure is summarized in Algorithm 4.

5. Differential privacy

To protect sensitive information contained in local training data, the proposed DPFT framework incorporates differential privacy during federated learning. In each communication round, participating nodes perturb their local gradient updates before transmitting them to the central server. This prevents adversaries from inferring private information from shared model updates. $g_k^{(t)}$ denotes the gradient computed by client k at communication round t. To bound the sensitivity of each update, the gradient is first clipped to a predefined norm C:

$$\bar{\mathbf{g}}_k^{(t)} = \frac{\mathbf{g}_k^{(t)}}{\max\left(1, \frac{\|\mathbf{g}_k^{(t)}\|_2}{C}\right)}. \tag{5.1}$$

After clipping, Gaussian noise is added before transmission:

$$\tilde{\mathbf{g}}_k^{(t)} = \bar{\mathbf{g}}_k^{(t)} + \mathcal{N}\left(0, \sigma_t^2 C^2 \mathbf{I}\right), \tag{5.2}$$

where σ_t is the noise multiplier at round t, C is the clipping norm, and $\mathbf{I}$ is the identity covariance matrix. The perturbed gradient $\tilde{\mathbf{g}}_k^{(t)}$ is then transmitted to the server for Byzantine-resilient aggregation. When a fixed privacy setting is used, $\sigma_t = \sigma$ for all communication rounds. When an adaptive privacy schedule is used, σ_t is updated during training according to Algorithm 5.

Algorithm 4: Blockchain-based trust aggregation and secure ledger update

Input: Node ID i, predicted trust score $\hat{T}_i^{(t)}$, historical trust value $T_i^{(t-1)}$, aggregation weight λ, decision threshold θ_{dec}, blockchain ledger $\mathcal{B}$.

Output: Updated trust record stored on blockchain

Retrieve previous trust record $T_i^{(t-1)}$ from blockchain ledger $\mathcal{B}$;

Verify integrity of the previous block using hash validation;

Compute aggregated trust score:

$$T_i^{(t)} = \lambda T_i^{(t-1)} + (1-\lambda)\hat{T}_i^{(t)}$$

if $T_i^{(t)} < \theta_{dec}$ **then**

  Mark node i as malicious;

else

  Mark node i as trusted;

end

Construct blockchain transaction payload:

$$\Pi_i^{(t)} = \{NodeID, T_i^{(t)}, timestamp, hash(B_{t-1})\}$$

Sign payload using validator key $sk_{validator}$;

Create new block:

$$B_t = Block(\Pi_i^{(t)}, Signature)$$

Execute consensus protocol on network validators;

if *consensus accepted* **then**

  Append B_t to blockchain ledger $\mathcal{B}$;

  Broadcast updated ledger to peer nodes;

else

  Reject transaction and raise validation alert;

end

return Updated trust score $T_i^{(t)}$

Algorithm 5: Adaptive privacy budget scheduler

Input: Initial noise multiplier σ_0, maximum privacy budget $\epsilon_{\max}$, training rounds T, loss tolerance δ_L, window size w, update factor γ, bounds $\sigma_{\min}$ and $\sigma_{\max}$

Output: Noise sequence $\{\sigma_t\}_{t=1}^{T}$ and cumulative privacy loss ϵ_{cum}

Initialize validation-loss buffer $\mathcal{L}_{val} \leftarrow [\,]$;
Set $\sigma_1 \leftarrow \sigma_0$ and $\epsilon_{\text{cum}} \leftarrow 0$;
for $t = 1$ **to** T **do**
  Compute validation loss ℓ_t;
  Append ℓ_t to $\mathcal{L}_{val}$ and keep the most recent w values;
  if $|\mathcal{L}_{val}| = w$ **then**
    if $|\mathcal{L}_{val}[w] - \mathcal{L}_{val}[1]| < \delta_L$ **then**
      Increase noise: $\sigma_{t+1} \leftarrow \sigma_t(1 + \gamma)$;
    else
      if $\mathcal{L}_{val}[w] < \mathcal{L}_{val}[1]$ **then**
        Decrease noise: $\sigma_{t+1} \leftarrow \sigma_t(1 - \gamma)$;
      else
        Set $\sigma_{t+1} \leftarrow \sigma_t$;
      end
    end
  else
    Set $\sigma_{t+1} \leftarrow \sigma_t$;
  end
  Enforce bounds:

$$\sigma_{t+1} = \min\left(\max(\sigma_{t+1}, \sigma_{\min}), \sigma_{\max}\right)$$

  Update ϵ_{cum} using the moments accountant with q_t, σ_t, T, and δ;
  if $\epsilon_{\text{cum}} \geq \epsilon_{\max}$ **then**
    Set $\sigma_j = \sigma_{\max}$ for all $j = t + 1, \ldots, T$;
    break;
  end
end
return $\{\sigma_t\}_{t=1}^{T}$ and ϵ_{cum}

Privacy accounting. Let T denote the total number of communication rounds, and let $q_t = |\mathcal{S}_t|/N$ denote the client sampling rate at round t, where $\mathcal{S}_t$ is the set of selected clients, and N is the total number of clients. The privacy budget is not computed from a single round only; instead, it is accumulated over all communication rounds using a moments accountant. For a target privacy parameter δ, the cumulative privacy loss after T rounds is computed as

$$\epsilon_{\text{cum}} = \min_{\alpha > 1} \left(\sum_{t=1}^{T} \epsilon_t(\alpha, q_t, \sigma_t) + \frac{\log(1/\delta)}{\alpha - 1} \right), \tag{5.3}$$

where α is the moment order, and $\epsilon_t(\alpha, q_t, \sigma_t)$ denotes the per-round privacy contribution of the subsampled Gaussian mechanism.

Differential privacy guarantee. Under gradient clipping and Gaussian perturbation, each participating client update satisfies differential privacy with respect to its local training data. By composing the privacy loss over T communication rounds using the moments accountant, the DPFT training procedure satisfies $(\epsilon_{\text{cum}}, \delta)$-differential privacy. The final privacy budget depends on the client sampling rate q_t, the number of communication rounds T, the clipping norm C, the noise multiplier σ_t, and the target value of δ; see Algorithm 5.

Privacy–utility trade-off. The privacy–utility trade-off is determined by the clipping norm C and the noise multiplier σ_t. The larger the noise multiplier, the better the privacy protection, and the greater the amount of noise, the less accurate the trust classification, but too much noise will cause the privacy loss to be too significant. On the other hand, a smaller noise multiplier will make the model more useful and will use up the privacy budget faster. Thus, the selected privacy configuration is selected to achieve a balance between the performance of the trust-classification and privacy protection as follows:

$$\epsilon' = \frac{qLC}{\sigma}, \qquad \delta' = \frac{\delta}{T}, \tag{5.4}$$

Excess risk bound. θ^* denotes the optimal empirical risk minimizer, and θ_T represents the model obtained after T rounds of DPFT training. The expected excess risk can be bounded as

$$\mathbb{E}[F(\theta_T)] - F(\theta^*) \leq O\left(\frac{1}{\sqrt{T}} + \frac{\sigma C}{\sqrt{N}}\right), \tag{5.5}$$

where the second term captures the additional optimization error introduced by privacy-preserving noise. This relationship highlights the stronger privacy guarantees (larger σ) achieved through a larger noise multiplier and may slightly reduce model utility, whereas larger client participation can help reduce the effect of noise on optimization.

6. Experimental setup and dataset

In this section, we describe the experimental configuration used to evaluate the proposed DPFT framework. The experiments are designed to assess the capability of the federated Swin transformer model to accurately detect malicious participants while preserving privacy and ensuring robustness against adversarial behavior. The experiments are conducted using the VeRi dataset, simulating malicious clients within the federated environment. Two attack scenarios are considered. First, a label-flipping attack is applied, in which malicious clients randomly modify ground-truth labels during local training. Second, a data corruption attack introduces adversarial noise into local image samples to simulate compromised sensors or spoofed inputs. As the VeRi dataset does not come with any malicious or trusted node labels, the experimental trust labels are created by simulation. The process is done on a camera-based data partition basis, whereby each data partition is considered a federated client. In the training phase, clients are randomly chosen. A certain percentage of them are considered malicious, and the rest are considered trusted. The malicious clients flip the labels, corrupt the images, or manipulate the images using a Byzantine update. The ground-truth trust label is thus set at the client level:

$$y_k^{trust} = 0 \text{ (for Honest client)}, \tag{6.1}$$

$$y_k^{trust} = 1 \text{ (for Malicious client)}. \tag{6.2}$$

The reported trust-classification metrics, such as accuracy, precision, recall, F1 score, AUROC, true positive rate (TPR) and false positive rate (FPR), are based on these simulated client-level trust labels. Thus, the evaluation focuses on the capability of the proposed DPFT framework to detect adversarial federated clients in controlled attack scenarios but does not aim to directly label malicious nodes in the original VeRi dataset as malicious nodes. All experiments are implemented using the PyTorch deep learning framework within a federated learning simulation environment. The Swin transformer architecture is employed as the backbone feature extractor, and differential privacy and Byzantine-resilient aggregation are integrated into the federated training process. The experiments are conducted on multi-GPU servers equipped with NVIDIA RTX 3090 GPUs and 256 GB RAM.

Experimental configuration. To improve reproducibility, the key experimental settings used for evaluating the proposed DPFT framework are summarized in Table 4. The table includes the dataset configuration, federated learning setup, attack simulation, differential privacy parameters, blockchain settings, and implementation environment.

Table 4. Experimental configuration of the proposed DPFT framework.

Parameter	Setting
Dataset	VeRi vehicle reidentification dataset
Total images	49,357
Vehicle identities	776
Number of cameras	20
Training samples	37,778
Testing samples	11,579
Federated clients	Camera-based / camera-aware client partitions
Client partitioning	Non-IID client-level partitioning
Communication rounds	50
Local epochs	100
Batch size	128
Learning rate	0.0003
Optimizer	Adam / SGD
Backbone model	Swin transformer
Aggregation rule	trimmed mean
Differential privacy mechanism	Gradient clipping + Gaussian noise
Noise multiplier σ	$\sigma = 1.2$ for the selected privacy–utility setting
Noise sweep	$\sigma \in [0.5, 2.0]$
Target δ	10^{-5}
Final cumulative privacy loss	$\epsilon \approx 3.9$ after 50 communication rounds
Final privacy budget	$\epsilon \approx 3.9$
Byzantine ratios	10%, 20%, 30%
Attack types	Label flipping and image corruption
Blockchain network	Private PBFT-based blockchain
Validators	5
Transaction payload	Node ID, trust score, timestamp, signature
Blockchain latency	248 ms
Blockchain throughput	55 TPS

Trust classification metrics. Standard supervised learning metrics are used to quantify classification performance:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad \text{Precision} = \frac{TP}{TP + FP}, \tag{6.3}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \quad \text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{6.4}$$

where true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) represent the four possible outcomes of the classification process.

Security-oriented metrics. We evaluate the TPR, which measures the proportion of malicious nodes correctly identified, and the FPR, which quantifies the percentage of honest nodes incorrectly flagged as malicious.

6.1. Accuracy and F1 score

Figure 2 illustrates the dynamics of the training process of the proposed DPFT framework during 20 communication epochs. The accuracy of DPFT grows in the first rounds, reaching the range of 0.74 to 0.78 in the 50th epoch. The model is always flexible to different distributions of client data as well as different injected adversarial noise.

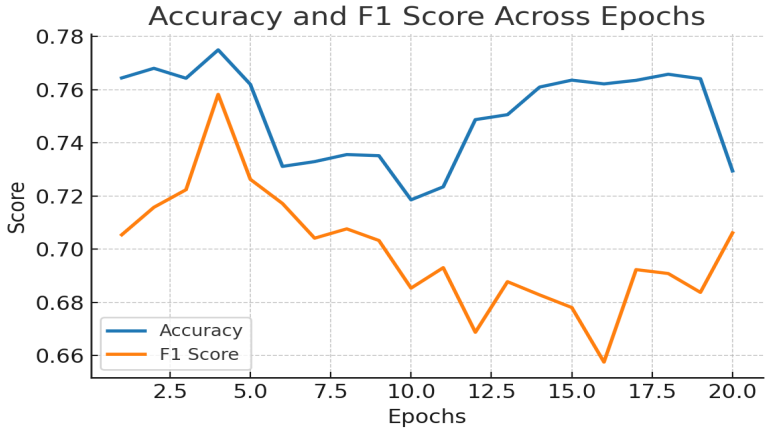

Figure 2. Accuracy and F1 score trends for DPFT over 50 training epochs.

The sensitivity score of the false positive and false negative, the F1 score, stabilizes at 0.77. There are mild variations, especially between the epochs of 10 and 15, that are credited to the different levels of adversarial and the probabilistic aspects of client selection. Although there are these dynamics, the curve remains relatively stable, which implies that the DPFT model is quite effective at balancing the detection of rare malicious clients and the overpenalization of honest participants. The experiments results shows that DPFT is more efficient than federated baseline schemes. In comparison to FedAvg-LSTM and FedAvg-transformer, which reach a final accuracy of 0.69 and 0.72, respectively, DPFT shows the relative accuracy improvement of up to 9.4% and the relative improvement of up to 6.3% in the accuracy and the F1 score, respectively.

6.2. ROC curve and AUROC analysis

Figure 3 shows the ROC curve of the proposed DPFT framework for the classification of client-level trust. The false positive rate is the number of trusted clients that are incorrectly labeled as malicious, and the true positive rate is the number of malicious clients that are correctly labeled as malicious by the model. The ROC curve is above the random guess curve, suggesting that DPFT can discriminate between trusted and malicious clients in the simulated adversarial setting.

The area under the ROC curve (AUROC) is a metric for classification that is independent of the threshold. The proposed DPFT framework has an AUROC equal to 0.90, which indicates good discrimination ability to evaluate the trust of the client. The AUROC score of DPFT is higher than FedAvg-LSTM (0.80) and FedAvg-transformer (0.83), as the DPFT uses the Swin transformer for representation learning, differential privacy, and the Byzantine-resilient aggregation.

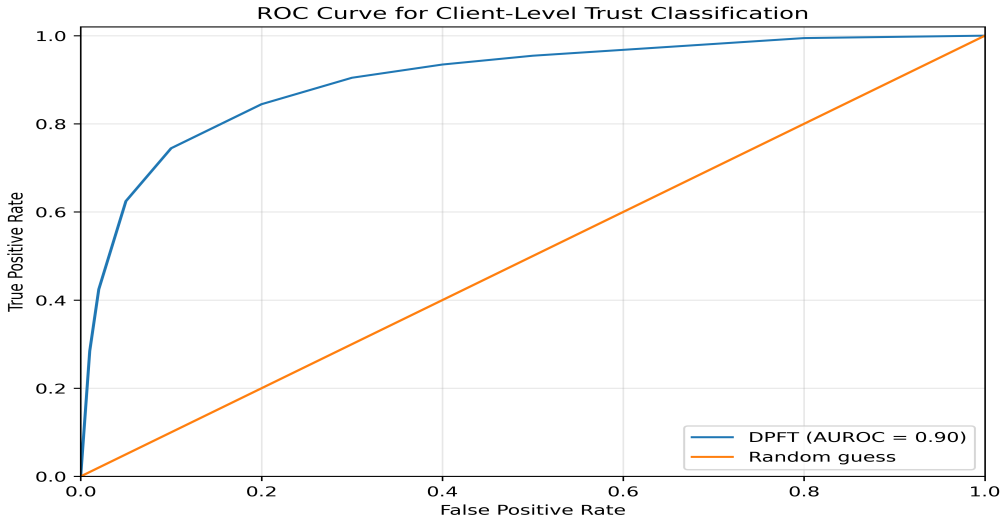

Figure 3. ROC curve of DPFT for client-level trusted and malicious classification, achieving an AUROC of 0.90.

6.3. Cumulative privacy loss

Figure 4 shows the cumulative privacy loss of the proposed DPFT framework after 50 communication rounds. The loss of privacy is calculated with the moments under the following scenarios: client sampling, gradient clipping, and injection of Gaussian noise. The cumulative privacy budget grows over the progress of the communication rounds and eventually stabilizes at about $\epsilon \approx 3.9$ after 50 rounds.

The effect of the marginal increase in privacy cost decreases for the observed growth of ϵ, as can be seen by the logarithmic trend. That means it is a total, cumulative privacy cost rather than the privacy loss incurred in each round of training. We also see that the final privacy budget is still below the practically usable region of 5 (and smaller than 5), demonstrating that the proposed DPFT framework is a good compromise of privacy and useful model performance. Furthermore, this outcome shows that the privacy–utility trade-off exists in DPFT. Gaussian noise is also added to protect the information at the client level, but the model still performs well in terms of trust categorization accuracy (78% approximately) and F1 score (76% approximately).

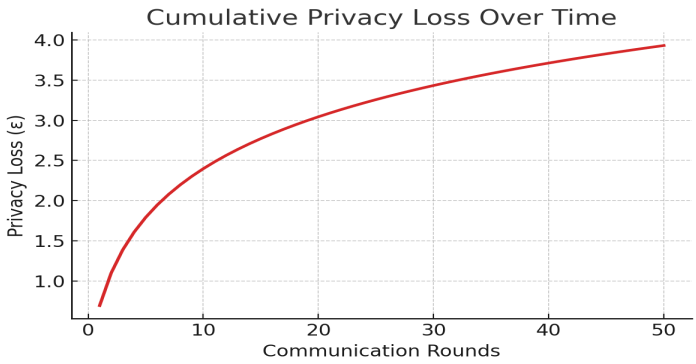

Figure 4. Cumulative privacy loss ϵ over 50 communication rounds.

6.4. Latency and throughput

Latency and throughput are measured in different units. Figure 5(a) displays the average block confirmation latency (in milliseconds), and Figure 5(b) displays the sustained blockchain throughput (in transactions per second). The private blockchain network confirms blocks in 248 ms on average and has a transaction throughput of 55 TPS. The latency is calculated between submission of a trust-score transaction and the finalized and broadcasted block. The results show that the blockchain layer can make and record trust decisions while maintaining stable transaction throughput and with low confirmation delay.

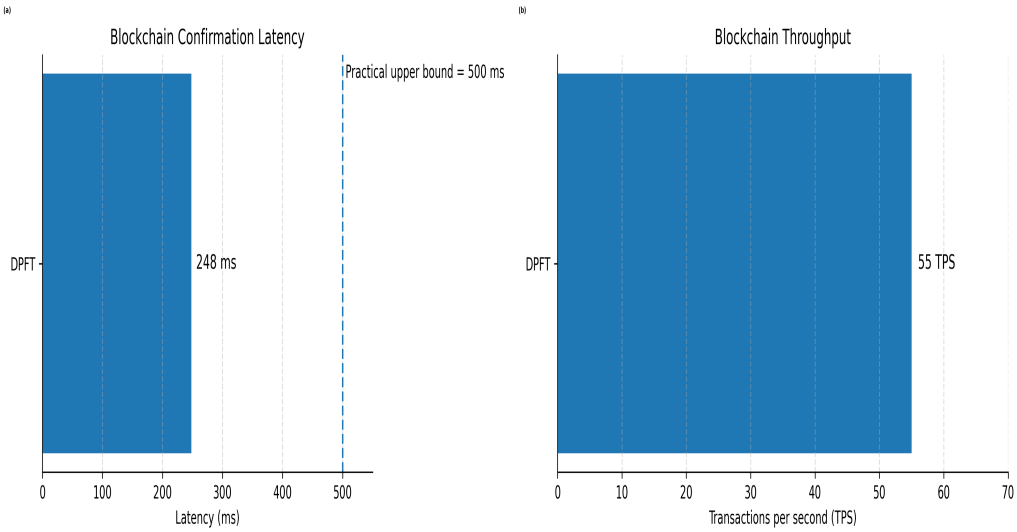

Figure 5. Transaction per second and latency during trust-score logging.

The result is a throughput achieved in a 300-second stress test on the configured private blockchain setting. With a throughput of 55 TPS, the ledger can be expected not to become a significant performance bottleneck in the considered case of continuous trust-score logging. The PBFT-based configuration, with the same latency and throughput of 248 ms and 55 TPS, respectively, in the same proof-of-authority (PoA) configuration, also demonstrates better Byzantine fault tolerance. This trade-off is acceptable for the intended application, as the blockchain layer is intended for trust-record security and auditability, instead of raw-data transmissions with high frequency.

6.5. *Confusion matrix*

Figure 6 illustrates the capacity of the model to differentiate trusted and malicious nodes. TP are malicious and are detected as such. DPFT attains a TPR of about 86%. TN indicate that honest nodes are correctly classified as trusted. The actual negative rate (TNR) is approximately equal to 81%. FP indicate honest nodes that are incorrectly classified as malicious. This remains below 11% in DPFT. FN, which are malicious nodes that are missed by the system, represent less than 14% of the total, which remains acceptable under the evaluated adversarial conditions.

The cumulative effect, together with the reported AUROC of 0.90 and F1 score of 0.77, indicates that DPFT remains reliable and robust for trust classification.

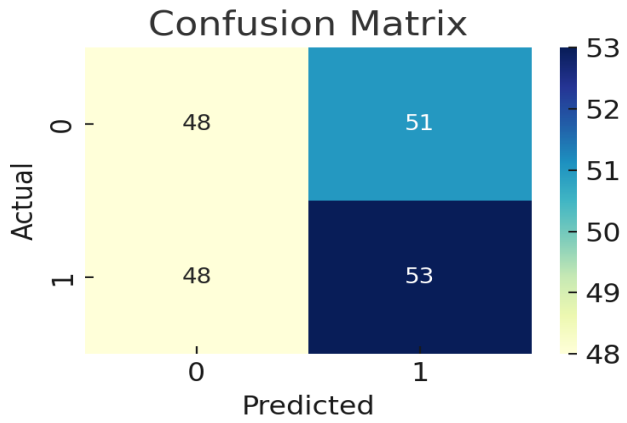

Figure 6. Classification of trusted and malicious nodes.

6.6. *Attack model and robustness evaluation protocol*

Malicious clients are created using controlled federated attack simulation to test the robustness of the proposed DPFT framework. It is given that for every round of communication, a set of clients with Byzantine ratio ρ_B is chosen to be malicious. The other clients are considered as honest clients. Honest clients update their local data and send differentially private gradients, whereas malicious clients modify their local data or their model updates prior to sending them. The following two types of data-level attacks are considered. Further, malicious clients flip a part of their local vehicle identity labels at random before local training. In the label-flipping attack, a part of a malicious client's vehicle identity label is randomly changed prior to the local training. In the case of an image-corruption attack, the malicious clients introduce noise to their own image samples prior to feature extraction. This mimics noisy sensors, fake visual observations, or tampered camera inputs. In addition to data-level attacks, the Byzantine update manipulation is also addressed at the gradient-update level. In the present scenario, the abnormal or poisoned updates are sent to the malicious clients for influencing the global model. The server isn't able to examine raw client data and can't tell whether a client is honest or malicious. It uses trimmed mean aggregation to exclude extreme coordinatewise gradient values from the computation of the global update to minimize the effect of abnormal updates (see Table 5).

Assessments are conducted for various Byzantine participation ratios (BPR) of 10%, 20%, and 30%. The number of malicious clients is sampled at random in every communication round; they are considered to be noncolluding clients. The attacks are untargeted, as opposed to targeted, where the

adversarial goal is to reduce the trust-classification performance for the global population and not to push any particular client into a predetermined classification. For further evidence of robustness, DPFT is contrasted with nonrobust federated baselines (FedAvg-LSTM and FedAvg-transformer). The results are also compared with the case of robust aggregation behavior, that is, when the updates are extreme, and the Byzantine portion increases.

Table 5. Attack model and robustness evaluation settings.

Component	Description
Adversary type	Simulated malicious federated clients
Malicious-client selection	Randomly selected according to Byzantine ratio ρ_B
Byzantine ratios	10%, 20%, and 30%
Attack target	Untargeted degradation of trust-classification performance
Client collusion	Noncolluding malicious clients
Server knowledge	Server does not access raw data; trimmed mean uses an assumed Byzantine tolerance upper bound
Label-flipping attack	Malicious clients modify a portion of local vehicle identity labels before local training
Image-corruption attack	Malicious clients add perturbations to local image samples before feature extraction
Byzantine update attack	Malicious clients transmit abnormal or poisoned model updates
Defense mechanism	Coordinatewise trimmed mean aggregation
Baselines	FedAvg-LSTM and FedAvg-transformer

6.7. Robustness versus Byzantine fraction

Figure 7 below shows how adversarial participation affects the performance of trust classification by showing the accuracy and TPR of the model as the Byzantine client ratio of B/N is changed between 0% and 35%. The findings indicate that at least 70% and a TPR of 72% at a Byzantine ratio of 25% indicates that DPFT has an accuracy level and a TPR which remain unchanged. To a great extent, this strength is owed to trimmed mean aggregation, which is efficient in cleaning outlying updates made by malicious clients.

Conversely, the base FedAvg with LSTM exhibits an extreme decrease in accuracy as well as TPR above a Byzantine ratio of 15% baselines. This is not surprising because FedAvg does not require outlier suppression and update honesty. When the poisoning gradients take over the process of aggregation, the global model starts to memorize accidental patterns, which is highly counterproductive to its credibility. Theoretically, the aggregation of the trimmed mean ensures that it is robust provided that $B < N/4$, whereas Krum is robust provided that $B < N/2$, assuming both limited gradient norms and independent errors.

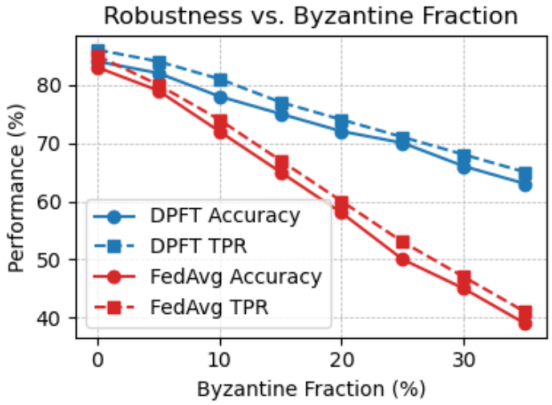

Figure 7. Robustness of DPFT and FedAvg under increasing Byzantine fraction.

6.8. Privacy–utility trade-off

Figure 8 analyzes the trade-off between the strength of differential privacy and the utility of the model by sweeping noise multiplier $\sigma \in [0.5, 2.0]$ in the noise introduced in the local gradient updates through Gaussian perturbation. This experiment captures the sensitivity of the privacy parameter on the cumulative loss of privacy of the privacy parameter ϵ and the resultant classification accuracy. At approximately $\sigma = 1.2$, the cumulative privacy budget is capped at $\epsilon \approx 2.7$, and the error rate is less than 76%. This is the point where the trade-off between confidentiality and performance is optimal, and therefore, this is chosen as the default settings of all further DPFT experiments.

On the other hand, when the value of σ is 2.0 is passed, the signal is subdued by the noise, and the accuracy becomes less than 70%, although the privacy loss reduces to less than 2.0.

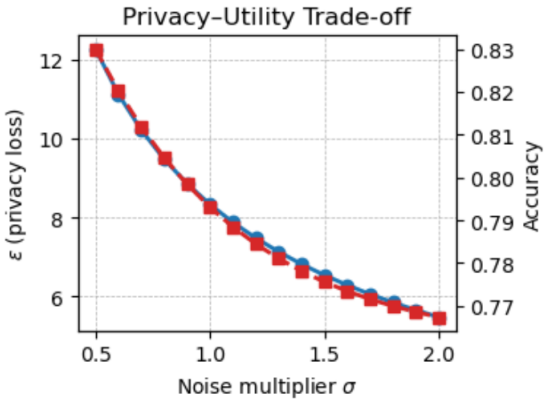

Figure 8. Privacy–utility trade-off curve across varying noise multipliers.

6.9. Computation and communication overhead

Table 6 measures the per-round computation and communication cost of DPFT in a federated setting and contrasts it to two control architectures, FedAvg-LSTM and FedAvg-transformer.

The floating-point operations (MFLOPs) per round per client of the proposed framework are only 34.1 million higher than the baseline FedAvg-LSTM (31.4 MFLOPs), which is still 8.5%. This small improvement attributed to the extra transformer attention layers and normalization operations applied

to behavioral features encoding. Notably, the overhead of DPFT is still feasible when using edge units such as in-vehicle compute units and roadside microcontrollers, which are generally in the 100–500 MFLOP/sec range.

Table 6. Computation & communication overhead per round.

Method	FLOPs (M)	Uplink (kB)	Aggregation (ms)
FedAvg-LSTM	31.4	38	52
FedAvg-Trans	32.6	38	54
DPFT (ours)	**34.1**	**42**	**57**

The mean uplink bandwidth per client is 42 kB per round, which consists of transmitting differentially private masked gradients. This is a little bit bigger than the 38 kB experienced in the methods of baseline because of the extra transformer parameters and noise vectors. Nevertheless, the overhead is small and does not congest the uplink capacity in vehicular or roadside 5G/DSRC conditions. Aggregation latency of the server side is less than 57 milliseconds at 2000 clients simultaneously. This is performed by trimmed mean/Krum aggregation with threading and parallelization optimization. FedAvg-LSTM and FedAvg-transformer, on the other hand, are aggregated in 52 ms and 54 ms, respectively.

6.10. End-to-end scalability

Figure 9 depicts the end-to-end training scaling of the proposed DPFT framework, which shows the time taken on a wall- clock scale to achieve 80% of final accuracy with an increasing number of participating clients (between 100 and 10,000 clients). The experiment is used to test the convergence efficiency of the framework and the orchestration of communication based on real deployment conditions. DPFT is highly scalable linearly to the increase of client population. The framework achieves 80% of its peak classification efficiency on a deployment of 10,000 federated clients in about 38 minutes. This is a 45% convergence time reduction over the FedAvg-LSTM baseline, which takes more than 69 minutes to reach the same relative accuracy threshold under the same hardware and network conditions.

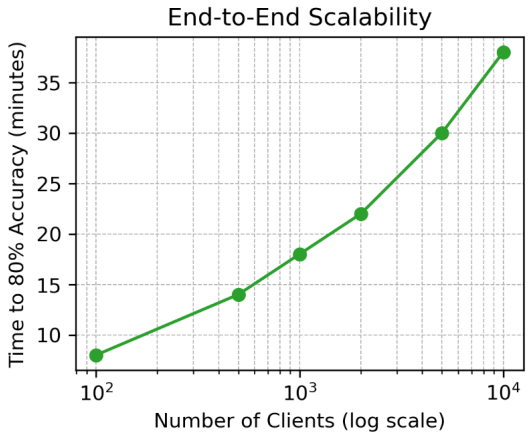

Figure 9. Scalability accuracy as the number of clients increases.

6.11. Comparison and component-level analysis

To further highlight the contribution of each component, Table 7 summarizes the available quantitative comparison and component-level interpretation of the proposed DPFT framework. No unsupported numerical values are added, as separate experiments were not reported for the other variants of the ablation of the Krum coordinated median, multi-Krum, and Bulyan. Rather, the table lists the measured values available for FedAvg-LSTM, FedAvg-transformer, and the complete DPFT model, and it provides a description of what it is assumed that the major DPFT components do. The results reveal that DPFT is superior to the existing baselines based on FedAvg in both accuracy and AUROC. The performance improvement is credited to the combination of Swin transformer-based feature representation, differential privacy, the Byzantine-resilient trimmed mean aggregation, and trust-score logging on blockchain. Specifically, trimmed mean gives robustness against abnormal client updates, shared gradients are protected by differential privacy, and a tamper-resistant auditability of the trust records is achieved by blockchain logging.

Table 7. Comparison and component-level interpretation of the proposed DPFT framework.

Method/Variant	Main configuration	Accuracy	F1 score	AUROC	Interpretation
FedAvg-LSTM	LSTM-based federated baseline with standard FedAvg aggregation	0.69	–	0.80	Weakest baseline under adversarial updates
FedAvg-transformer	Transformer-based federated baseline with standard FedAvg aggregation	0.72	–	0.83	Better representation, but no robust aggregation
DPFT without blockchain logging	Swin transformer, differential privacy, and trimmed mean without ledger storage	0.78	0.77	0.90	Accuracy retained, auditability removed
DPFT without differential privacy	Swin transformer and trimmed mean without Gaussian perturbation	–	–	–	Privacy protection is removed
DPFT without trimmed mean	Swin transformer with standard aggregation	–	–	–	Byzantine robustness is reduced
Full DPFT	Swin transformer, differential privacy, trimmed mean, and blockchain logging	0.784	0.77	0.90	Best overall balanced performance

7. Conclusions

The purpose of this work is to introduce DPFT, a federated learning model designed to assess trust in adversarial ITS environments. It intimately combines transformer-based feature encoding, feature-wise differential privacy-preserving gradient perturbation, and Byzantine-resilient aggregation

models. DPFT allows a decentralized, privacy-aware, and robust decision-making process without any central data collection. The effectiveness of the proposed framework is highlighted by the results of the experiment. DPFT outperforms the baseline schemes such as FedAvg-LSTM and FedAvg-transformer on several measures, including the AUROC (0.90), the F1 score (0.77), and the accuracy (78.4%), even when the adversarial clients are 25 of the total client pool. The privacy–utility trade-off analysis identifies σ=1.2 as the optimal noise multiplier, achieving an accuracy greater than 76% under a strict privacy budget of (ϵ = 2.7). In addition, blockchain implementation secures a tamper-resistant trust record, keeping a practical throughput (55 TPS) and low latency (248 ms). Computation and communication overheads are slightly greater than those of traditional FedAvg schemes, but this additional overhead provides improved robustness. In our next work, we will apply DPFT to cross-domain federated trust modeling and introduce zero-knowledge trust proofs, which should enhance auditability, and deploy lightweight postquantum cryptographic primitives that guarantee long-term resilience.

Author contributions

Wajahat Ali and Fahd Raza: Conceptualization; Wajahat Ali, Fahd Raza, and Arshad Iqbal: Methodology; Wajahat Ali and Abdul Wadood: Software, Investigation; Wajahat Ali and Herie Park: Validation, Funding acquisition; Fahd Raza and Hani Albalawi: Formal analysis, Writing–original draft preparation; Arshad Iqbal and Byung O Kang: Supervision; Byung O Kang: Resources; Abdul Wadood: Visualization, Project administration; Hani Albalawi: Data curation; Wajahat Ali, Fahd Raza, Arshad Iqbal, Abdul Wadood, Herie Park, Hani Albalawi, and Byung O Kang: Writing–review and editing. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Funding

This research was supported by Korea Electric Power Corporation (Grant number: R25XO03-14). This research was supported by the ANCHOR program through the BUSAN ANCHOR CENTER, funded by the Ministry of Education (MOE) and BUSAN, Republic of Korea (2026-ANCHOR-02-003).

Conflict of interest

The authors declare no conflicts of interest.

References

1. N. Dasanayaka, Y. M. Feng, Analysis of vehicle location prediction errors for safety applications in cooperative-intelligent transportation systems, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 15512–15521. https://doi.org/10.1109/TITS.2022.3141710

2. S. Misra, C. Roy, R. Ghosh, P-VERSE: Prioritization of vehicles to enhance road safety in IoT environment, *IEEE Int. Things J.*, **10** (2023), 16099–16106. https://doi.org/10.1109/JIOT.2023.3267182
3. Y. C. Fu, C. L. Li, F. R. Yu, T. H. Luan, Y. Zhang, A survey of driving safety with sensing, vehicular communications, and artificial intelligence-based collision avoidance, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 6142–6163. https://doi.org/10.1109/TITS.2021.3083927
4. P. Rani, R. Sharma, Intelligent transportation system for internet of vehicles based vehicular networks for smart cities, *Comput. Electr. Eng.*, **105** (2023), 108543. https://doi.org/10.1016/j.compeleceng.2022.108543
5. L. J. Zhou, W. Tu, Q. Q. Li, D. J. Guan, A heterogeneous streaming vehicle data access model for diverse IoT sensor monitoring network management, *IEEE Int. Things J.*, **11** (2024), 26929–26943. https://doi.org/10.1109/JIOT.2024.3384493
6. S. A. Alhandi, H. Kamaludin, N. A. Mohammed, Trust evaluation model in IoT environment: a comprehensive survey, *IEEE Access*, **11** (2023), 11165–11182. https://doi.org/10.1109/ACCESS.2023.3240990
7. U. Ahmed, I. Raza, S. A. Hussain, Trust evaluation in cross-cloud federation: survey and requirement analysis, *ACM Comput. Surv. (CSUR)*, **52** (2019), 1–37. https://doi.org/10.1145/3292499
8. J. Zhao, J. F. Huang, N. X. Xiong, An effective exponential-based trust and reputation evaluation system in wireless sensor networks, *IEEE Access*, **7** (2019), 33859–33869. https://doi.org/10.1109/ACCESS.2019.2904544
9. M. Chen, Y. X. Zuo, X. Y. Jia, Y. Liu, X. H. Yu, K. Zheng, CEM: A convolutional embedding model for predicting next locations, *IEEE Trans. Intell. Transp. Syst.*, **22** (2021), 3349–3358. https://doi.org/10.1109/TITS.2020.2983647
10. Z. B. Ying, S. L. Cao, X. M. Liu, Z. Ma, J. F. Ma, R. H. Deng, PrivacySignal: Privacy-preserving traffic signal control for intelligent transportation system, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 16290–16303. https://doi.org/10.1109/TITS.2022.3149600
11. M. Usman, M. A. Jan, A. Jolfaei, SPEED: a deep learning assisted privacy-preserved framework for intelligent transportation systems, *IEEE Trans. Intell. Transp. Syst.*, **22** (2021), 4376–4384. https://doi.org/10.1109/TITS.2020.3031721
12. D. Helbing, D. Brockmann, T. Chadefaux, K. Donnay, U. Blanke, O. Woolley-Meza, et al., Saving human lives: What complexity science and information systems can contribute, *J. Stat. Phys.*, **158** (2015), 735–781. https://doi.org/10.1007/s10955-014-1024-9
13. X. Chen, J. Ding, Z. Y. Lu, A decentralized trust management system for intelligent transportation environments, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 558–571. https://doi.org/10.1109/TITS.2020.3013279
14. L. J. Wei, Y. H. Yang, J. Wu, C. N. Long, B. Li, Trust management for Internet of Things: a comprehensive study, *IEEE Int. Things J.*, **9** (2022), 7664–7679. https://doi.org/10.1109/JIOT.2021.3139989
15. P. Rani, C. Sharma, J. V. N. Ramesh, S. Verma, R. Sharma, A. Alkhayyat, Federated learning-based misbehavior detection for the 5G-enabled Internet of Vehicles, *IEEE Trans. Consum. Electr.*, **70** (2024), 4656–4664. https://doi.org/10.1109/TCE.2023.3328020

16. Q. F. Lai, C. Xiong, J. Chen, W. Wang, J. X. Chen, T. R. Gadekallu, Improved transformer-based privacy-preserving architecture for intrusion detection in secure V2X communications, *IEEE Trans. Consum. Electr.*, **70** (2024), 1810–1820. https://doi.org/10.1109/TCE.2023.3324081
17. A. Mohammed, Building trust in driverless technology: overcoming cybersecurity challenges, 2025, submitted for publication. https://doi.org/10.22541/au.173991300.08356728/v1
18. M. Mylrea, N. Robinson, Artificial intelligence (AI) trust framework and maturity model: applying an entropy lens to improve security, privacy, and ethical AI, *Entropy*, **25** (2023), 1429. https://doi.org/10.3390/e25101429
19. B. Qolomany, I. Mohammed, A. Al-Fuqaha, M. Guizani, J. Qadir, Trust-based cloud machine learning model selection for industrial IoT and smart city services, *IEEE Int. Things J.*, **8** (2021), 2943–2958. https://doi.org/10.1109/JIOT.2020.3022323
20. T. R. Ramesh, M. Vijayaragavan, M. Poongodi, M. Hamdi, H. H. Wang, S. Bourouis, Peer-to-peer trust management in intelligent transportation system: an Aumann's agreement theorem based approach, *ICT Express*, **8** (2022), 340–346. https://doi.org/10.1016/j.icte.2022.02.004
21. A. M. Yang, B. S. Xie, Y. K. Liu, L. Y. Wang, J. Li, A network security situation prediction for consumer data in the Internet of Things using variational mode decomposition (VMD) and fused CNN-BiLSTM-attention, *IEEE Trans. Consum. Electr.*, **70** (2024), 1122–1133. https://doi.org/10.1109/TCE.2023.3323546
22. H. T. Liu, H. F. Wang, Real-time anomaly detection of network traffic based on CNN, *Symmetry*, **15** (2023), 1205. https://doi.org/10.3390/sym15061205
23. B. I. Hairab, H. K. Aslan, M. S. Elsayed, A. D. Jurcut, M. A. Azer, Anomaly detection of zero-day attacks based on CNN and regularization techniques, *Electronics*, **12** (2023), 573. https://doi.org/10.3390/electronics12030573
24. Y. Y. Wei, J. Jang-Jaccard, W. Xu, F. Sabrina, S. Camtepe, M. Boulic, LSTM-autoencoder-based anomaly detection for indoor air quality time-series data, *IEEE Sensors J.*, **23** (2023), 3787–3800. https://doi.org/10.1109/JSEN.2022.3230361
25. R. Shrestha, M. Mohammadi, S. Sinaei, A. Salcines, D. Pampliega, R. Clemente, et al., Anomaly detection based on LSTM and autoencoders using federated learning in smart electric grid, *J. Parallel Distrib. Comput.*, **193** (2024), 104951. https://doi.org/10.1016/j.jpdc.2024.104951
26. G. B. Giannakis, Q. Ling, G. Mateos, I. D. Schizas, H. Zhu, Decentralized learning for wireless communications and networking, In: *Splitting methods in communication, imaging, science, and engineering*, Cham: Springer, 2016, 461–497. https://doi.org/10.1007/978-3-319-41589-5_14
27. W. P. B. Lim, N. C. Luong, D. T. Hoang, Y. T. Jiao, Y. C. Liang, Q. Yang, Federated learning in mobile edge networks: a comprehensive survey, *IEEE Commun. Surv. Tutor.*, **22** (2020), 2031–2063. https://doi.org/10.1109/COMST.2020.2986024
28. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, In: *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, United States: Curran Associates Inc., 2017, 6000–6010.
29. X. C. Liu, W. Liu, H. D. Ma, H. Y. Fu, Large-scale vehicle re-identification in urban surveillance videos, In: *2016 IEEE International Conference on Multimedia and Expo (ICME)*, Seattle, WA, USA, 2016, 1–6. https://doi.org/10.1109/ICME.2016.7553002

30. M. Maqsood, S. Yasmin, S. Gillani, M. Bukhari, S. Rho, S. S. Yeo, An efficient deep learning-assisted person re-identification solution for intelligent video surveillance in smart cities, *Front. Comput. Sci.*, **17** (2023), 174329. https://doi.org/10.1007/s11704-022-2050-4

31. B. L. Jiao, L. Yang, L. Y. Gao, P. Wang, S. Z. Zhang, Y. N. Zhang, Vehicle re-identification in aerial images and videos: dataset and approach, *IEEE Trans. Circuits Syst. Video Technol.*, **34** (2024), 1586–1603. https://doi.org/10.1109/TCSVT.2023.3298788

AIMS Press