Robust Intrusion Detection System for Vehicular Networks: A Federated Learning Approach Based on Representative Client Selection

Chunyang Fan, Jie Cui, Hulin Jin, Hong Zhong, Irina Bolodurina, Debiao He

***Abstract*—The rapid development of network technology has allowed numerous vehicular applications to be deployed in vehicles, thereby enriching the driving experience of users. However, the openness of vehicular networks enables attackers to launch network attacks on vehicles through network ports, leading to the destruction of vehicular networks. To develop an intrusion detection system suitable for distributed vehicular networks, researchers have utilized federated learning to train detection models. Nevertheless, most federated learning-based vehicular intrusion detection systems seldom consider rapidly updating the detection model and fail to detect unknown attacks effectively. In this study, we propose a federated learning-based vehicular intrusion detection system that fully considers the traffic characteristics of multiple network regions and selects representative clients to participate in model aggregation, thereby accelerating the convergence of the global model. Furthermore, to enhance the robustness of the detection system, we utilize extreme value theory and multilayer activation vectors to construct an unknown attack discriminator that can determine whether a network flow is an unknown attack. Comprehensive experiments on three open datasets demonstrate that the proposed intrusion detection system can quickly update and effectively identify known/unknown attacks in open vehicular networks.**

***Index Terms*—Vehicular networks, Intrusion detection system, Federated learning, Extreme value theory, Unknown attack.**

I. Introduction

VEHICULAR networks have seen rapid growth with the large-scale deployment of the 5G infrastructure. 5G networks guarantee low-latency, high-bandwidth communication for vehicles, which has led to the widespread emergence of numerous vehicular safety/entertainment applications and a significant enhancement in the driving experience of vehicle users [1], [2]. Operating vehicular applications requires vehicles to open external ports to receive downlink data from roadside units (RSUs) or base stations [3], [4]. However, because of the open nature of wireless networks, malicious attackers can launch network attacks on moving vehicles, leading to severe traffic accidents [5]. Therefore, it is extremely important to develop a reliable and efficient intrusion detection system to secure vehicular networks.

To improve the security of vehicular networks, many researchers have proposed centralized vehicular intrusion detection systems for detecting possible network attacks [6]. These schemes train a centralized detection model using cloud servers and deploy the model on each vehicle and RSU to defend against network attacks. However, vehicular networks are large distributed networks with multiple vehicles and RSUs, and each region exhibits certain differences in network traffic characteristics [7]. Therefore, training a centralized detection system using cloud servers is difficult to apply to all network regions, and is prone to the biased-flow problem [8] (i.e., the detection system is more likely to identify attack traffic in a certain region and has difficulty identifying attack traffic in other regions).

To address the problems with centralized intrusion detection systems, researchers considered using federated learning to train collaborative intrusion detection systems [9], [10]. Federated learning allows multiple vehicles and RSUs to train detection models locally and then upload the model parameters/gradients rather than data to a cloud server for model aggregation. This approach improves the accuracy and generalization of the detection models while ensuring data privacy. Moreover, the continuous updating and learning capabilities of federated learning enable intrusion detection systems to adapt to changes in vehicular networks continuously [11], [12]. However, most federated learning-based vehicular intrusion detection systems select clients for model aggregation based on sample quantity. This may lead to the fixation of clients participating in model aggregation and hinder the rapid update of the detection model. Particularly, in dynamically changing vehicular networks, fast updating of the detection model enables more timely adaptation to network changes.

Furthermore, vehicular networks are open scenarios where attackers may continuously launch new types of network attacks on vehicles or RSUs [13]. Because intrusion detection systems have not learned the data features of these novel attacks, it is difficult for vehicles to effectively defend against these novel unknown attacks [14]. In particular, most vehicular intrusion detection systems use softmax as the final layer of the model output, resulting in the detection category being fixed in a predefined range, while unknown attacks do not belong. Consequently, it is difficult for these systems to guarantee robust detection.

C. Fan, J. Cui, H. Jin and H. Zhong are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China, and the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China (e-mail: cuijie@mail.ustc.edu.cn).

Irina Bolodurina is with the Faculty of Mathematics and Information Technologies, Orenburg State University, Orenburg, 460018, Russia (e-mail: prmat@mail.osu.ru).

D. He is with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China and the Shanghai Key Laboratory of Privacy-Preserving Computation, MatrixElements Technologies, Shanghai 201204, China (email: hedebiao@163.com).

To realize the ability of the detection system to quickly update and effectively identify unknown attacks in the distributed vehicular network, we first design a federated learning algorithm based on representative client selection. The algorithm uses gradient information to cluster clients and obtains multiple client clusters. As the clients within the clusters are extremely similar, a representative client from each cluster can be selected to participate in the global model aggregation, thereby ensuring the effectiveness of each aggregation and accelerating the model update. Secondly, to overcome the limitation of softmax and efficiently recognize unknown attacks, we design an unknown attack discriminator based on extreme value theory and multilayer activation vectors. The extreme value theory can detect unknown classes because it focuses intensely on the tail characteristics of statistical distributions. In general classification tasks, known classes often occupy the main portion of the distribution, whereas unknown classes typically appear in the tail regions [15]. Therefore, we can obtain an unknown attack discriminator by feeding the multilayer activation vectors of normal traffic/known attacks into the extreme value theory. Moreover, to enable the activation vectors at each layer to better represent the features of their respective classes, we reconstruct the original input using the activation vectors and then compare the reconstructed vector with the original input vector to obtain the reconstruction error. By continuously reducing the reconstruction error loss, the reconstructed vectors and the original input vectors become progressively more similar, and the activation vectors produced by the detection system become more representational. In summary, our contribution can be summarized as follows:

- We propose a federated learning algorithm based on representative client selection for vehicular networks, which uses gradient information to cluster clients at each iteration and selects representative clients from each cluster to participate in model aggregation, thereby significantly reducing the number of training iterations and accelerating model updates.
- To improve the robustness of the detection system, we design an unknown attack discriminator based on multilayer activation vectors and extreme value theory that can effectively identify unknown attacks in vehicular networks.
- Comprehensive experiments on three open datasets show that the proposed intrusion detection system for vehicular networks can effectively identify known/unknown attacks and can quickly update the detection system. Moreover, the comparison experiments show that the proposed detection system outperforms the state-of-the-art methods in terms of precision rate, recall rate, and F1-Score.

The rest of the paper is organized as follows. Section II describes related works. Section III presents the preliminaries. In Section IV, we introduce the proposed intrusion detection system for vehicular networks. In Section V, we evaluate the overall performance of the proposed system. In Section VI, we discuss the deployment of detection systems and privacy issues. Finally, we summarize our work in Section VII.

II. Related Work

In recent years, researchers have designed a large number of intrusion detection systems to secure networks. In this section, we classify existing intrusion detection systems into three categories: 1) centralized training-based systems, 2) distributed training-based systems, and 3) systems that can recognize unknown attacks.

A. Intrusion Detection System Based on Centralized Training

With the successful application of deep learning in image recognition and natural language processing, many researchers have utilized deep learning to build intrusion detection systems. To defend against network attacks from outside the vehicle, Anbalagan *et al.* [16] designed an intelligent intrusion detection system based on convolutional neural networks that improve detection accuracy by converting traffic data in text form into image form. Wang *et al.* [17] proposed an intrusion detection system integrating temporal and spatial dimensions, which constructs a vehicle state matrix to characterize the potential correlation of data between each sensor and other sensors, thus enabling defense against multiple attacks. Almutlaq *et al.* [18] proposed a two-phase intrusion detection system based on a deep neural network rule extraction method, which first distinguishes between normal traffic and abnormal traffic. If the traffic is found to be malicious, the type of attack traffic is detected in the second phase. To quickly detect and localize attacks, Deng *et al.* [19] proposed an intrusion detection system based on voltage signals, which establishes voltage fingerprints for each control LAN identifier and uses these fingerprints to quickly localize attacks. Xun *et al.* [20] designed an intrusion detection system called FeatureBagging-CNN, which identifies the unique voltage signal of each ECU by recognizing its unique voltage signal characteristics to detect attacks from external or internal sources. These centralized training-based detection systems can effectively detect network attack traffic. However, vehicular networks are distributed networks, and detection systems obtained using centralized training methods are difficult to install quickly on vehicles and RSUs. In addition, centralized training requires uploading data to a cloud server, which can lead to particularly high model update overhead.

B. Intrusion Detection System Based on Distributed Collaborative Training

To enable joint training of intrusion detection systems, Abdel-Basset *et al.* [21] proposed a federated learning-based intrusion detection learning framework that collaboratively trains the detection system by offloading the learning task to distributed edge nodes to improve the detection range. Considering the limited computing power of vehicles, Houda *et al.* [22] proposed a federated learning-based collaborative intrusion detection mechanism that offloads the training model onto roadside units to reduce the vehicle computational load. In addition, the mechanism uses blockchain to store and share the training models to secure the aggregated models. Cui *et al.* [23] proposed a collaborative intrusion detection system

based on software-defined networks and federated learning, which utilizes vehicular network traffic monitored by multiple SDNs to jointly train the intrusion detection system, thus improving the scalability of the detection system. Qu *et al.* [24] proposed a collaborative intrusion detection system based on the improved residual network, which introduces an attention mechanism into the residual network to efficiently capture key features of the traffic data, thus improving the accuracy of the detection system. The above schemes can effectively accomplish the task of training and installing the detection model in the vehicular networks. However, due to the dynamic and open nature of vehicular networks, intrusion detection systems need to be quickly updated and have the ability to detect unknown attacks, while the above detection systems have difficulty meeting these requirements.

C. Intrusion Detection Systems for Recognizing Unknown Attacks

In recent years, a large number of researchers have explored the detection of unknown attacks to improve the robustness of intrusion detection systems. Verkerken *et al.* [25] proposed a novel multi-stage hierarchical intrusion detection method to detect unknown zero-day attacks, which first distinguishes the presence of anomalies in the network traffic and then further categorizes the anomalous traffic. Zhang *et al.* [26] proposed an intrusion detection system that can identify anomalous traffic, which utilizes k-means and isolated forests to construct a boundary that distinguishes normal traffic from abnormal traffic. Using this boundary, the detection system can detect anomalous network traffic that contains unknown attacks, but the system cannot further classify the anomalous traffic. Fan *et al.* [27] proposed an auto-updating intrusion detection system for vehicular networks, which utilizes a two-layer filter to detect potential unknown attacks and updates the attack classifier with the data of the detected unknown attacks. To improve the efficiency of detecting unknown attacks, Verma *et al.* [28] proposed an intrusion detection system for 5G IoT scenarios, which utilizes an autoencoder and one-class support vector machines to achieve the identification of known/unknown attack traffic. Although the above schemes achieve detection of unknown attacks, these schemes either only detect anomalous traffic or have a large detection overhead.

In summary, researchers have designed many excellent intrusion detection systems to secure vehicular networks. However, these schemes have some limitations. The centralized training-based detection system requires RSUs to continuously upload new samples to the cloud server when updating, resulting in a large update overhead. The detection system based on distributed training can greatly reduce the update overhead, but it is difficult to achieve fast updates to the detection model and identify unknown attacks. Meanwhile, some researchers have proposed intrusion detection systems that can detect unknown attacks. Still, these systems either can only detect anomalous traffic or the detection model is too complex, resulting in a large detection overhead. Considering these limitations, we propose a robust intrusion detection system for vehicular networks, which can quickly update the detection model and efficiently recognize unknown attacks.

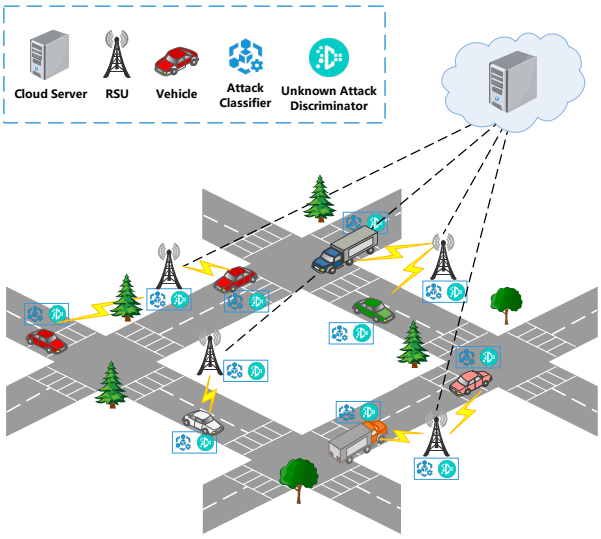


Fig. 1. Network model

III. Preliminary

A. Network Model

The proposed network model mainly consists of a cloud server, RSUs, vehicles, and intrusion detection systems. It is worth mentioning that the intrusion detection system contains an attack classifier and an unknown attack discriminator. The detailed network model is shown in Fig. 1.

1) Cloud Server: The cloud server is the server in the federated learning mechanism, which is mainly responsible for performing the parameter aggregation task of the global model. The updated model parameters will also be sent to each RSU by the cloud server.

2) RSU: RSU is the client in the federated learning mechanism that trains a detection model by utilizing the local dataset. There are two reasons why RSU is chosen as the client: 1) The network data uploaded or downloaded by the vehicle is forwarded through the RSU, so the RSU can obtain real-time network traffic to form a local dataset. 2) The RSU has a more powerful computation and storage capacity compared to the vehicle, which can better realize the training of the attack classifier.

3) Vehicle: The trained Intrusion detection systems are mainly installed in vehicles to defend against potential network attacks.

4) Intrusion Detection System: The proposed intrusion detection system consists of an attack classifier and an unknown attack discriminator. It is worth mentioning that the generation of an unknown attack discriminator depends on the attack classifier. Therefore, in the federated learning mechanism, we first train an excellent attack classifier and then generate the unknown attack discriminator based on the classifier and extreme value theory.

B. Components of Attack Classifier

To attain accurate detection of malicious network traffic, the proposed attack classifier is constructed using 1D-CNN and a

multi-head attention mechanism. 1D-CNN is used to extract key features of the network flow, and the multi-head attention mechanism then computes the correlation of these key features with each other to form a global feature representation of the network flow. The detailed construction process of the attack classifier is as follows:

1) 1D-CNN: A classical convolutional neural network consists of convolution operation and pooling operation [29], [30]. The convolution operation is used to extract the key features of the network traffic, and the specific convolution process is shown below:

$$h_i = \sigma(\sum_{j=0}^{m^c-1} w_j^c x_{i+j} + b) \tag{1}$$

where m^c is the size of the convolution kernel, w_j^c is the value of the j-th weight of the convolution kernel, b is the offset, σ is the activation function, x_{i+j} is the $(i+j)$-th element of the input sequence, and h_i is the i-th element of the output of the convolution operation. It is worth mentioning that h_i is the output of a convolution operation. For an input sequence x of length l, the result $C^{res} = \{h_0, h_1, h_2, \cdots h_s\}$ is obtained after multiple convolution operations, where $s = l - m^c + 1$.

Typically, convolutional neural networks utilize pooling layers to reduce the risk of overfitting. However, using pooling operations may result in the loss of important information due to the small number of network traffic features. Therefore, to ensure that the key features are not lost, we refrain from pooling operation on the resulting output sequence C^{res}.

2) Multi-head Attention: Compared to the single-head attention mechanism, multi-head attention allows multiple heads to learn different concerns in parallel, which helps to extract important features and improve the accuracy of the model [31]. Therefore, we use the multi-head attention mechanism to capture the correlation between elements of the output sequence C^{res}. Assuming H is the number of attention heads, for the i-th attention head, we have three linear transformations for query (Q), key (K) and value (V):

$$\begin{cases} Q_i = C^{res} \cdot W_{Qi} \\ K_i = C^{res} \cdot W_{Ki} \\ V_i = C^{res} \cdot W_{Vi} \end{cases} \tag{2}$$

where W_{Qi} is the query weight matrix, W_{Ki} is the key weight matrix and W_{Vi} is the value weight matrix.

According to the query matrix Q_i and the key matrix K_i, we can obtain the attention matrix A_i:

$$A_i = Softmax\left(\frac{Q_i \cdot K_i^T}{\sqrt{d_k}}\right) \tag{3}$$

where $Softmax$ is an activation function and d_k is the dimension of the K_i matrix. It is worth mentioning that the matrix A_i contains the relationships between any two elements in the convolution result C^{res}. Then, we utilize the attention matrix A_i to extract key information from matrix V_i, generating a sequence O_i that reflects the relationships between elements. The value of O_i is calculated as follows:

$$O_i = Attention(Q_i, K_i, V_i) = A_i \cdot V_i \tag{4}$$

Note that O_i will pay too much attention to one part of the sequence C^{res}, leading it to neglect other important information. Therefore, we need to perform a merge operation for the H heads to prevent the loss of important sequence features. The specific merge operation is shown below:

$$O = (O_1 \oplus O_2 \oplus \cdots \oplus O_H) \cdot W^O \tag{5}$$

where O is the output of the multi-head attention mechanism and W^O is a trainable weight matrix. Then, we obtain O' by flattening O and obtain an activation vector y by a linear transformation for O'. The specific process is shown in Equation (6).

$$y = [y_1, y_2, \cdots, y_R, y_{R+1}] = (W^T \cdot O') + b^f \tag{6}$$

where $R+1$ is the number of classification results (i.e. normal traffic and R known attacks), W^T is the weight matrix of the linear transformation, and b^f is the offset. Generally, the probability that a network flow belongs to each class can be obtained by processing y using the softmax function:

$$p_j = \frac{exp(y_j)}{\sum_{i=1}^{R+1} exp(y_i)} \tag{7}$$

where p_j is the probability that the network flow belongs to the j-th class and y_j is the j-th element of y. Hence, the attack classifier enables the classification of normal traffic and known attacks. Note that the attack classifier is the base model in federated learning, and each client (i.e. RSU) uses local data to train the classifier.

IV. The Proposed Intrusion Detection System for Vehicular Networks

In this section, we first introduce traditional federated learning and discuss its limitations, next describe the proposed representative client-based federated learning approach, then present the construction process of the unknown attack discriminator, and finally introduce the training process of the overall detection system.

A. Traditional Federal Learning Process

In the initial phase of federal learning, the cloud server needs to send down the proposed attack classifier to each RSU, which trains the attack classifier using the local dataset. After completing the trained attack classifier model, the server selects some clients for global model aggregation. Specifically, we assume that there are N clients participating in the model aggregation, with n_i samples per client and M total samples. Then, the parameters of the global model can be calculated as follows:

$$\theta_g^t = \sum_{i=1}^{N} \frac{n_i}{M} \theta_i^t \tag{8}$$

where θ_g^t is the global model parameters at time t and θ_i^t is the model parameters of the i-th client. Commonly, clients need to go through multiple model aggregations to make the global model optimal. The training process of federated learning can be divided into the following three steps:

This article has been accepted for publication in IEEE Transactions on Mobile Computing. 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/TMC.2025.3582237

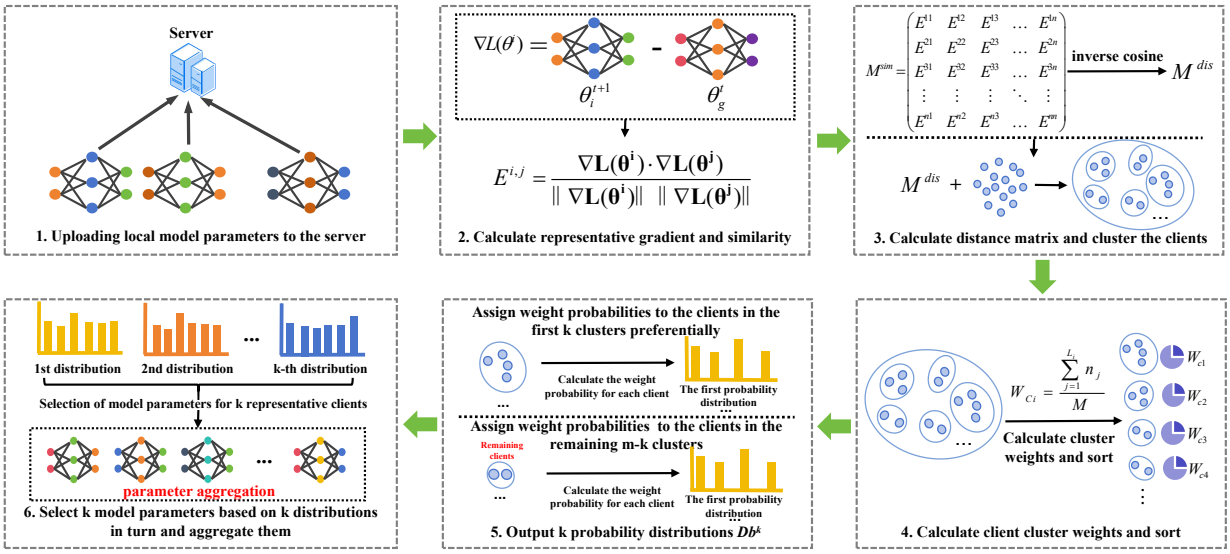


Fig. 2. Flowchart of the proposed federated learning algorithm

1) Local model training: The aggregated global model parameters θ_g^t are sent to each client (i.e. RSU), which uses θ_g^t to update the parameters of the local model. Then, the client will train the model again using the local dataset to get new model parameters θ_i^{t+1}.

2) Parameters aggregation: To increase the model generalization ability and avoid the bias flow problem, the cloud server needs to select some clients' model parameters for aggregation. There are two methods for client selection in traditional federated learning 1) randomly selecting clients and 2) selecting clients to participate in parameters aggregation by client's sample size. The cloud server generates global model parameters θ_g^{t+1} by aggregating the model parameters of these clients.

3) Local model parameters update: After completing the aggregation, the cloud server sends the global model parameters θ_g^{t+1} to each client, which uses θ_g^{t+1} to update the local model.

Steps 1-3 are one round of the federated learning process. After multiple training rounds, the global model will gradually converge, and the process of federated learning ends. Note that the client selection in step 2 affects the speed of model convergence. In dynamic vehicular networks, fast convergence of the detection model is beneficial for better adapting to changes in the network environment. Nevertheless, in traditional federated learning, the client selection method based on the number of samples faces the problem of repeated client selection, while the random-based selection method suffers from the problem of unstable training efficiency. Therefore, the clients participating in model aggregation need to be selected based on the model training effect, i.e., selecting representative clients in each iteration.

B. Select Representative Clients

To realize fast convergence of the global model, we need to select representative clients to participate in model aggregation. The flowchart of our proposed federated learning algorithm based on representative client selection is shown in Fig. 2. Several studies have shown that comparing client's representative gradients can measure the similarity between models [32], [33]. The representative gradient is the difference between the updated client model parameters and the global model parameters. Therefore, we can obtain a set of representative gradients $G = \{\nabla L(\theta^1), \nabla L(\theta^2), \cdots, \nabla L(\theta^n)\}$. Then, we do cosine similarity calculation on these representative gradients as shown below:

$$E^{i,j} = \frac{\nabla \mathbf{L}(\theta^{\mathbf{i}}) \cdot \nabla \mathbf{L}(\theta^{\mathbf{j}})}{\|\nabla \mathbf{L}(\theta^{\mathbf{i}})\| \|\nabla \mathbf{L}(\theta^{\mathbf{j}})\|} \tag{9}$$

where $E^{i,j}$ is the similarity between client i and client j. It is worth noting that the similarity computation is done in pairs, so we get a matrix M^{sim} that records the similarity between clients, as follows:

$$M^{sim} = \begin{pmatrix} E^{11} & E^{12} & E^{13} & \dots & E^{1n} \\ E^{21} & E^{22} & E^{23} & \dots & E^{2n} \\ E^{31} & E^{32} & E^{33} & \dots & E^{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ E^{n1} & E^{n2} & E^{n3} & \dots & E^{nn} \end{pmatrix}$$

Note that the matrix M^{sim} is a symmetric matrix, i.e., $E^{i,j}$ equals $E^{j,i}$ and $E^{i,i}$ equals 0. To show the distance characteristics between different clients, we perform the inverse cosine process to generate M^{dis} for M^{sim}. In M^{dis}, the distance values between similar clients are small, and the distance values between dissimilar clients are large. Then, we utilize a hierarchical clustering algorithm to perform a clustering operation on M^{dis} so that select representative clients participate in model aggregation. Specifically, the hierarchical clustering process can be divided into two steps:

1) Neighbor cluster merging: According to the distance matrix M^{dis}, we can find the two closest clusters C_i and C_j. Then, these two clusters are merged to generate a new cluster C_n.

Algorithm 1: Select Representative Clients

Input: m clusters $Clu^m = \{C_1, C_2, \cdots, C_m\}$, A large integer E

Output: k distributions $Db^k = \{db_1, db_2, \cdots, db_k\}$, where $db_i = \{p_1, p_2, \cdots, p_N\}$ (p_i is the probability that client i is selected, and the initial value is 0.)

Calculate the weight of each cluster in Clu^m:
/* Calculate the weight of cluster C_i, L_i is the number of clients in C_i, and n_j is the number of samples in the j-th client, M is the total sample size. */
$W_C = \{W_{C1}, W_{C2}, \cdots, W_{Cm}\}$
for $i \leftarrow 1$ **to** m **do**
  $W_{Ci} = \frac{\sum_{j=1}^{L_i} n_j}{M}$;
Select the k highest weighted clusters Clu^k from Clu^m based on W_C;
Assign weights to clients in the Clu^k preferentially:
$W^{sum} = \{W_1^{sum}, W_2^{sum}, \cdots, W_k^{sum}\}$;
for $i \leftarrow 1$ **to** k **do**
  $W_i^{sum} = 0$;
  for $j \leftarrow 1$ **to** L_i^k **do**
    /* n_j^i is the number of samples from the j-th client in the i-th cluster */
    $W_i^j = \frac{n_j^i}{M} \times k \times E$;
    $W_i^{sum} = W_i^{sum} + W_i^j$;
    Assign the weight W_i^j to the corresponding client in the db_i.
Assign weights to the clients in the remaining m-k clusters Clu^r:
/* Assuming that $r = m - k$ */;
flag=1;
for $\iota \leftarrow 1$ **to** r **do**
  for $t \leftarrow 1$ **to** L_ι^r **do**
    /* n_t^ι is the number of samples from the t-th client in the ι-th cluster */
    $W_\iota^t = Min\{\frac{n_t^\iota}{M} \times k \times E, E - W_{flag}^{sum}\}$;
    $W_{flag}^{sum} = W_{flag}^{sum} + W_\iota^t$;
    Assign the weight W_ι^t to the corresponding client in the db_{flag};
    if W_{flag}^{sum} == E **then**
      flag=flag +1;
Convert the weights in the Db^k into probability form;
for $\kappa \leftarrow 1$ **to** k **do**
  for $u \leftarrow 1$ **to** N **do**
    $p_u = \frac{p_u}{Sum(db_\kappa)}$;
return Db^k;

2) Updated distance matrix M^{dis}: After neighbor clusters are merged, we need to update the distance matrix M^{dis} to record the distance from the new cluster C_n to each cluster. The distance from the new cluster C_n to the other clusters is calculated as follows:

$$dist(C_n, C_o) = Min(dist(C_i, C_o), dist(C_j, C_o)) \quad (10)$$

where $dist$ is the Euclidean distance function, C_o is the other clusters, and $dist(C_i, Co)$ denotes the distance from C_i to C_o. Through constant repetition of steps 1 and 2, we can obtain m clusters with the number of clients in each cluster $C^{num} \geq 1$. Assuming we have to select k clients to participate in model aggregation, $m \geq k$ (i.e., the number of clusters to be clustered should be greater than the number of clients participating in model aggregation).

To improve the probability that representative clients are selected, we process the obtained m clusters. The specific processing is shown in Algorithm 1. The core idea of Algorithm 1 is to preferentially assign weights to the clients of the representative k clusters and form k probability distributions. Then, the clients of the remaining m-k clusters are assigned weights and sequentially filled into these k probability distributions to form the final probability distribution Db^k. By utilizing k probability distributions Db^k, we can sample k times and obtain k clients participating in model aggregation. Note that since we preferentially assign weights to the clients of the most representative k clusters, the clients of these k clusters have a greater probability of being selected. It is worth mentioning that the clients in the remaining m-k clusters also have the probability of being selected for model aggregation. Thus, the proposed method considers the diversity of clients.

Through multiple local training and model aggregation, we can finally obtain an optimal attack classifier model F^r that can effectively detect known attacks and normal traffic present in the vehicular network. In conclusion, our proposed federated learning algorithm based on representative client selection can provide a positive effect on the training of the detection system. Specifically, the proposed vehicular network intrusion detection system can be continuously updated by leveraging the designed federated learning algorithm, thereby adapting to the rapidly changing network environment. In addition, our designed unknown attack discriminator relies on the activation vector of the attack classifier. By utilizing the designed federated learning algorithm, the activation layer feature information of the attack classifier can be enriched. As a result, the classifier obtained from federation learning-based training can provide richer and more diverse feature information for the unknown attack discriminator, which improves the detection performance of unknown attacks. Finally, the detection bias problem caused by centralized training is avoided because data features of representative clients from different regions are considered in the federated learning process.

C. Unknown Attack Discriminator

The proposed intrusion detection system for vehicular networks consists of an attack classifier and an unknown attack discriminator, where the training of the unknown attack discriminator relies on the attack classifier. It is worth mentioning that the input of the unknown attack discriminator is the output of each layer of the classifier neural network, i.e., the activation

This article has been accepted for publication in IEEE Transactions on Mobile Computing. 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/TMC.2025.3582237

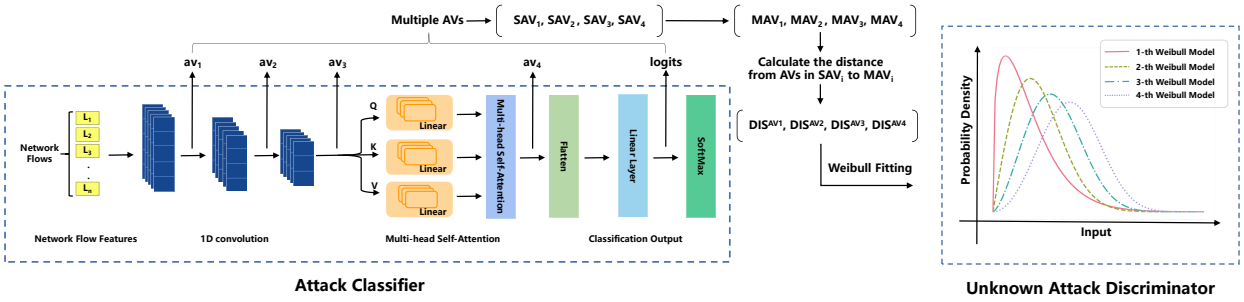


Fig. 3. The construction process of unknown attack discriminator

vectors. The reason for using activation vectors as input is that activation vectors contain rich feature information that can directly reflect the characterization of normal or attack traffic. Another reason for using multiple activation vectors is the small number of vehicular network traffic features. It is difficult to construct boundaries between each known class using only the activation vectors of the last layer of the neural network. Therefore, the attack classifier inputs each layer's activation vectors into the unknown attack discriminator for processing, and the unknown attack discriminator returns R+2 probability values, where R is the number of known attacks and 2 represents normal traffic and unknown attacks. It is important to note that the core idea of an unknown attack discriminator is to isolate unknown attacks from multiple boundaries by utilizing information about the characteristics of known attacks and normal traffic to construct decision boundaries. The construction process of the discriminator is shown in Fig. 3. Assume that we have a set of network flow data $D^t = \{x_1, x_2, \cdots, x_i, \cdots, x_n\}$ and its corresponding labeled set $L^t = \{y_1, y_2, \cdots, y_i, \cdots, y_n\}$, where y_i is the label of network flow sample x_i. Let $Cla = \{c_1, c_2, \cdots, c_R, c_{R+1}\}$ be the label space of D^t, i.e., D^t contains $R+1$ categories (normal traffic and R known network attacks). With federated learning, we obtain a final attack classifier F^r. The dataset D^t and label set L^t are fed into the classifier F^r for processing to obtain $R+1$ sets of classification results. Then, we select the correctly predicted samples and compute their activation vectors for each layer. In this paper, we use a three-layer convolutional neural network and a multi-head attention mechanism so that for a sample x, its activation vector is $AV = \{av_1, av_2, av_3, av_4, logits\}$. Similarly, for all correctly predicted samples, we can obtain the set of activation vectors for each class. Let $SAV_i = \{AV_1, AV_2, \cdots\}$ be the set of activation vectors of the i-th class, and all correctly predicted activation vectors can be denoted as $\sum_i^{R+1} SAV_i$. Note that the activation vector AV implies the main characteristics of a network flow, and the SAV_i can reflect the main features of the i-th class of network flows. Therefore, SAV can be used to construct multiple boundaries to detect unknown network attacks. However, since SAV is difficult to describe in the relationship between AVs, we need to transform SAV into the corresponding distance set DIS^{AV}. DIS^{AV} can define clear decision boundaries and simplify the computational complexity. The process of conversion from SAV to DIS^{AV}

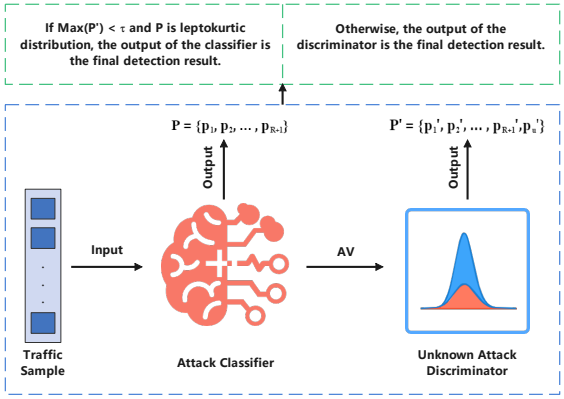


Fig. 4. Classifier and discriminator collaborative detection flowchart

can be divided into the following two steps:

1) Calculate the mean activation vector: Since distance is a relative concept, we need to determine a fixed point as a reference. The best fixation point is the mean activation vector (MAV) of the SAV. For SAV_i, its MAV_i is calculated as follows:

$$MAV_i = \frac{1}{N^{av}} \sum_{j=1}^{N^{av}} AV_j \tag{11}$$

where N^{av} is the number of AVs in SAV_i.

2) Calculate the distance between AVs and MAV: For SAV_i, we take MAV_i as a fixed point and calculate the distance from each AV to MAV_i as follows:

$$d(AV_j, MAV_i) = \sqrt{(AV_j - MAV_i)^2} \tag{12}$$

where $d(AV_j, MAV_i)$ is the distance from the j-th AV to MAV_i. By using Equation (12), we can obtain the final distance set DIS_i^{AV}. Then, we input the distance set DIS^{AV} into the Weibull model of extreme value theory to fit multiple decision boundaries. It is important to note that the Weibull model is a distribution function and fitting the Weibull model using DIS^{AV} is a process of determining the values of the parameters. The equation to calculate the Weibull model is as follows:

$$f(d; a, \lambda, \gamma) = \frac{a}{\gamma}\left(\frac{d-\gamma}{\lambda}\right)^{a-1} e^{-\left(\frac{d-\gamma}{\lambda}\right)^a} \tag{13}$$

where a is the shape parameter, λ is the scale parameter, γ is the location parameter, and d is the distance. Using Equation (13), we can obtain $R+1$ Weibull models (i.e., each known class has a corresponding Weibull model).

During the detection phase, the workflow of collaborative detection between the classifier and the discriminator is shown in Fig. 4, and the specific process is described as follows:

1) Input a sample $\mathcal{X}$ into the attack classifier F^r to obtain the activation vector $AV_{\mathcal{X}}$ and classification results $P = \{p_1, p_2, \cdots, p_{R+1}\}$.
2) Calculate the distance from the $AV_{\mathcal{X}}$ to each MAV to obtain a distance set $D^{\mathcal{X}} = \{d_1, d_2, \cdots, d_{R+1}\}$.
3) The distances in $D^{\mathcal{X}}$ are input into the corresponding Weibull model to obtain a set of probability values $P^{\mathcal{X}} = \{p_1^x, p_2^x, \cdots, p_i^x, \cdots, p_{R+1}^x\}$, where p_i^x is the probability that $\mathcal{X}$ does not belong to class i, so $\hat{p}_i = 1 - p_i^x$ is the probability that $\mathcal{X}$ belongs to class i.
4) The unknown attack is not recognized due to the fact that the original predicted probability of normal traffic and known attack traffic sums to 1. Therefore, we need to revise the prediction scores for normal traffic and known attack traffic so that unknown attacks have a probability of being recognized. For normal traffic and known attacks, the new prediction score can be calculated as follows:

$$Score_i' = logits_i \times \hat{p}_i \tag{14}$$

where $Score_i'$ is the new prediction score for class i, and $logits_i$ is the initial prediction score for class i. For unknown attacks, the prediction score can be calculated as follows:

$$Score^u = \sum_{i=1}^{R+1} (logits_i \times p_i^x) \tag{15}$$

where $Score^u$ is the prediction score for the unknown attack. Thus, we can obtain a new set of prediction scores $\{Score_1', Score_2', \cdots, Score_{R+1}', Score^u\}$. Then, we performed $Softmax$ on this set of predicted scores and obtained a set of probability values $P' = \{p_1', p_2', \cdots, p_{R+1}', p_u'\}$, where p_u' is the probability of unknown attacks. We determine that $\mathcal{X}$ is an unknown attack when $Max(P') = p_u'$ and $Max(P')$ is greater than a threshold τ. The reason for adding the condition of $Max(P') > \tau$ is that when the value of $Max(P')$ is low, the difference in prediction scores between normal flows/known attacks and unknown attacks is relatively small, which leads to an increase in the false positive rate of the unknown attack discriminator. Therefore, to reduce the false alarm rate, we combine the classifier's output to determine the final class of $Max(P') < \tau$ samples. More specifically, when the classifier gives a leptokurtic distribution (i.e., there exists a probability value that is much larger than the others), we choose the output of the classifier as the detection result. Otherwise, we choose the output of the discriminator as the detection result.

D. Training process for attack classifier and unknown attack discriminator

Since the construction of an unknown attack discriminator relies on an attack classifier, we need to train an attack classifier first using federated learning. Then, we use the generated optimal global model to train the unknown attack discriminator. The training process of the attack classifier and unknown attack discriminator is shown below:

1) Initialize the global model (i.e., attack classifier) and send the model to each client. In the first round of training, the client uses local data for training and uploads the trained model parameters to the cloud server for model aggregation.
2) Since we do not have an updated global model in the first round of training, we cannot calculate the representative gradient. Starting from the second round of training, we use the representative client selection algorithm to select specific clients to participate in model aggregation.
3) During local training, we use AVs to reconstruct the original input samples to improve the feature representation of AVs. Suppose the input sample is X and the reconstructed sample is $\hat{X}$, then the reconstruction loss can be calculated as follows:

$$Loss^R = (X - \hat{X})^2 \tag{16}$$

In addition, to realize the accurate identification of normal traffic and known attacks, we need to calculate the classification loss with the following loss function:

$$Loss^C = -\frac{1}{n} \sum_{i=1}^{n} \sum_{c=1}^{R+1} y_{ic} log(p_{ic}) \tag{17}$$

where n is the number of samples, $R+1$ is the number of categories, y_{ic} is the true label of the i-th sample in category c, and p_{ic} is the predicted probability that sample i belongs to category c. The overall loss during local training is calculated as follows:

$$Loss = Loss^R + Loss^C \tag{18}$$

4) After multiple rounds of local training and model aggregation, the loss of the model will no longer decrease, and we will obtain an optimal global model F^r. The local client can get the AVs that predict the correct samples by utilizing F^r. These AVs are uploaded to the cloud server, and the cloud service can generate the unknown attack discriminator by utilizing the steps described in subsection C.
5) Note that since the unknown attack discriminator is composed of multiple probability density distribution functions, the discriminator occupies less storage space and has low computational overhead.

V. Experimental results

In this section, we will describe the open-source dataset used for the experiments, introduce the experimental setup, and analyze the performance of the proposed vehicular intrusion

TABLE I
DIVIDE KNOWN CLASSES AND UNKNOWN ATTACKS FROM THREE DATASETS

Dataset	Known Classes			Unknown Attacks	
	Traffic Type	Training Sample Size	Test Sample Size	Traffic Type	Sample Size
CICIOV	Normal, Speed, RPM, Steering_wheel	926496	397069	DoS, GAS	42327
TON-IOT	Normal, DoS, DDoS, Injection, Scanning	12954	3238	XSS, Password	1225
EDGE-IIOT	Normal, DDoS_ICMP, DDoS_UDP, DDoS_TCP, Backdoor, Port_Scanning,DDoS_HTTP, SQL_injection, Ransomware, XSS	1289960	552840	Uploading, Password, Vulnerability_scanner	137337

detection system. Specifically, we will discuss the following five research questions:

- **Q1**: How is the overall performance of the proposed vehicular intrusion detection system?
- **Q2**: What is the increase in convergence speed of the proposed federated learning training method compared to other methods?
- **Q3**: Compared to other methods, does the proposed federated learning training method consume less time to update the system?
- **Q4**: Is the constructed detection system better for identifying network attacks compared to other methods?
- **Q5**: What is the detection performance of intrusion detection systems for known/unknown attacks at different thresholds?

A. Datasets

In the experiments, we use the open-source datasets CICIOV [34], ToN-IoT [35], and EDGE-IIOT [36] to train an vehicular network intrusion detection system. The CICIOV dataset is an attack dataset for vehicular network communication that collects attacks such as DoS, False Speed, and False Steering. ToN-IoT and EDGE-IIOT have been commonly used training datasets for intrusion detection systems in recent years. The ToN-IoT dataset is a multi-attack type IoT dataset, which contains attacks such as DoS, DDoS, and data injections. EDGE-IIOT is a large-scale dataset that contains millions of network traffic with attacks such as SQL injection, port scanning, and DDoS. Note that to simulate unknown attacks that exist in real scenarios, we further divide each dataset into a known attack part and an unknown attack part, and the training dataset of the attack classifier does not contain unknown attacks. During the testing phase, we combined the test set samples with unknown attack samples to evaluate the performance of the proposed vehicular intrusion detection system. The specific division of the three datasets is shown in Table I. It is worth mentioning that this method of simulating unknown attack scenarios is referenced from Yang *et al.* [37].

B. Experimental settings

1) Experimental Environment: All training tasks run on an AND EPYC 7763 server with 240G RAM and NVIDIA A100 GPUs. In our experiments, we simulate that there are thirty RSUs participating in the federated learning training, and the training data is divided into 30 parts. Note that each RSU does not have an equal number of samples. We select 6 representative clients in each training iteration to participate in model aggregation. In addition, we use the Pytorch 2.0.0 algorithm library and Libmr algorithm library to build the proposed vehicular intrusion detection system.

2) Evaluation Metrics: To evaluate the performance of the proposed vehicular intrusion detection system in recognizing known or unknown network attacks, we calculated the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) predicted by the detection system. According to these definitions, we can calculate the precision, recall, and F1 score using Eq. (19), Eq. (20), and Eq. (21), respectively.

$$Precision = \frac{TP}{TP + FP} \tag{19}$$

$$Recall = \frac{TP}{TP + FN} \tag{20}$$

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{21}$$

C. Overall performance of the proposed detection system (Q1)

To answer Q1, we use the confusion matrix to evaluate the overall performance of the proposed scheme. The confusion matrix of the proposed scheme in three datasets is shown in Fig. 5. We can clearly observe that the proposed scheme mostly exceeds 90% detection accuracy for known attacks and normal traffic, and exceeds 80% detection accuracy for unknown attacks. The reason for being able to achieve a high detection accuracy is that we perform secondary detection on samples with low confidence values, thus reducing the probability of the samples being misidentified. In addition, the accuracy of the proposed scheme in detecting normal traffic is close to 100%, and unknown attacks are mostly not misidentified as normal traffic. The reason for this phenomenon is that the number of samples in normal flows is much higher than that of attack flows. Therefore, the proposed intrusion detection system is able to fully understand the data features of normal flows, thereby achieving close to 100% detection accuracy for normal flows.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. 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/TMC.2025.3582237

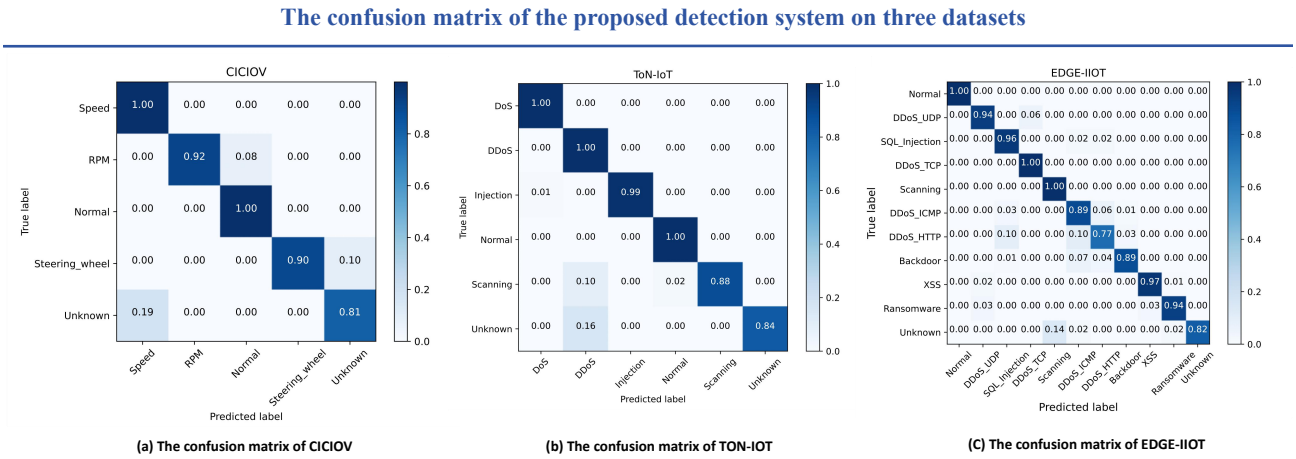
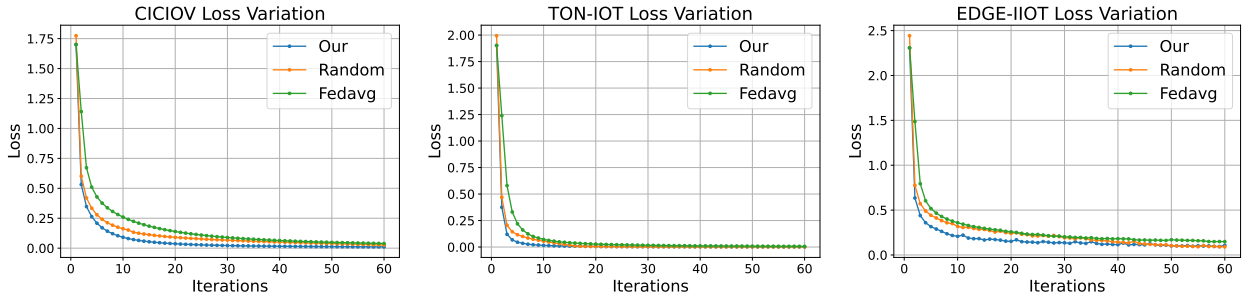


Fig. 5. Overall performance of the proposed detection system on three datasets



(a) The loss variation of CICIOV (b) The loss variation of TON-IOT (c) The loss variation of EDGE-IIOT

Fig. 6. Comparison of loss variation during training

D. Loss variation of our proposed method on three datasets compared to other methods (Q2)

It is well known that the loss value variation can reflect the speed of model convergence. Usually, as the training progresses, the loss value will gradually decrease to a fixed value, which means that the model training is completed. To demonstrate that the proposed collaborative intrusion detection system can be updated quickly, we use loss variation to evaluate the convergence speed of the system. Meanwhile, we compare the proposed scheme with the Fedavg algorithm and randomly selected client algorithm, and the specific results are shown in Fig. 6. We can observe clearly that our proposed method leads to a rapid decrease in the value of the loss. The reason for this phenomenon is that the Fedavg algorithm selects participants for model aggregation based on the sample size of clients, which leads to almost the same clients participating in model aggregation each time and makes it difficult to achieve effective model aggregation. Although the algorithm based on randomly selecting clients can avoid selecting the same set of clients multiple times, the clients selected to participate in model aggregation may have very similar model parameters, resulting in ineffective model aggregation. However, the proposed method reduces the number of model iterations by selecting representative clients for model aggregation during each iteration of the federated learning training process.

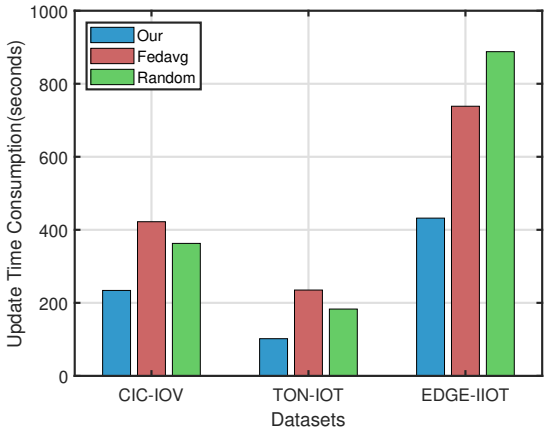


Fig. 7. Comparison of update system time consuming for different methods on three datasets

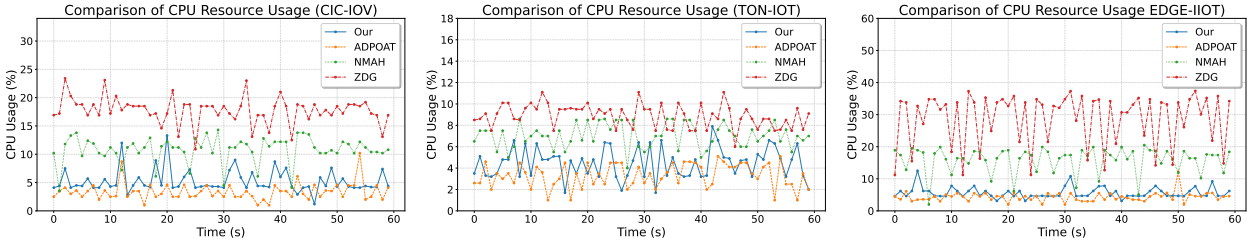
E. Comparison of System Update Time Consumption(Q3)

To demonstrate that the proposed federated learning training method can update the detection system quickly, we measured the time consumption required by different methods to update the model once, and the experimental results are shown in Fig. 7. It is worth mentioning that federated learning mechanisms typically include 1) client local training, 2) model parameter/gradient uploading and offloading, and 3) global

This article has been accepted for publication in IEEE Transactions on Mobile Computing. 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/TMC.2025.3582237

TABLE II
COMPARISON WITH OTHER METHODS

Method	CICIOV			TON-IOT			EDGE-IIOT		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
NMAH	92.47	94.01	93.23	93.05	98.82	95.85	91.65	92.86	92.26
ADPOAT	90.48	89.83	90.16	96.33	78.32	86.4	89.21	87.39	88.29
ZDG	84.2	97.43	90.33	95.44	90.79	93.06	85.23	97.23	90.83
OUR	97.41	97.34	97.38	98.39	99.46	98.92	96.03	97.27	96.65



(a) Comparison of CPU resource usage (CI-CIOV)
(b) Comparison of CPU resource usage (TON-IOT)
(c) Comparison of CPU resource usage (EDGE-IIOT)

Fig. 8. Compare the CPU resource usage of different detection systems

model aggregation. Therefore, the time overhead of model updating also depends on these steps. In particular, since we introduced the representative client selection algorithm, which needs to do clustering operations on the gradients of multiple local models, it leads to our scheme having a higher time consumption in one training iteration. However, because the proposed scheme selects representative clients in each iteration, the model converges faster than the other schemes (the Loss variation in Fig. 6 shows that the proposed scheme converges much faster than the other schemes). Therefore, the proposed scheme can utilize less time to complete the updating of the model under the comprehensive measurements.

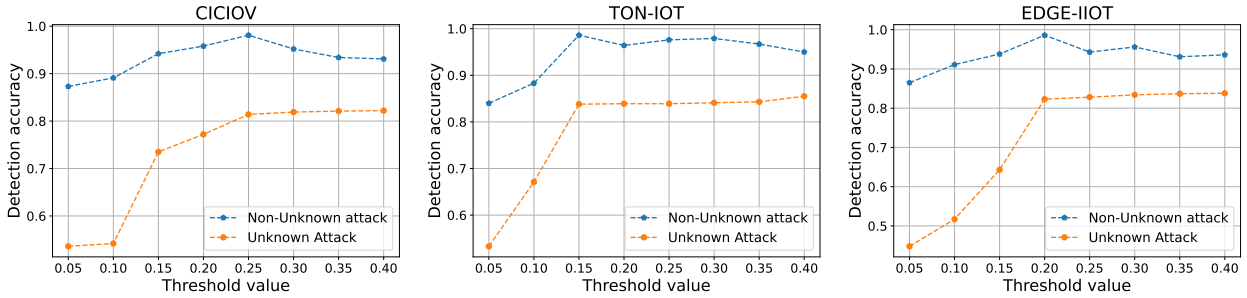
F. Comparison with other methods (Q4)

To answer Q4, we compare the proposed vehicular intrusion detection system with existing schemes. The main features of the compared schemes are described below:

- **NMAH** [25]: To enable more comprehensive detection of possible malicious attacks in the network, the scheme proposes a multi-stage-based intrusion detection system, which utilizes a one-class support vector machine (OCSVM) algorithm in the first stage to identify whether the network traffic is malicious or not. If anomalous traffic is detected, the traffic is detected a second time using random forests and neural networks to determine the class of the attack. In addition, if the anomalous traffic does not get a high prediction value in the secondary detection, a third detection is triggered to prevent the traffic from being misidentified. It is worth mentioning that there exists a threshold value for each detection stage, and the authors utilize these three thresholds to achieve effective detection for network attacks. However, designing too many thresholds can affect the accuracy of the detection system, and it is challenging to design these three thresholds to optimize the system capability.
- **ADPOAT** [26]: To achieve the detection of network attacks with a small number of labeled samples, the scheme proposes an intrusion detection system based on Kmeans clustering, which first performs a clustering operation on all labeled anomalous traffic to obtain multiple clusters of samples (each cluster corresponds to a network attack). Then, the Isolation Forest algorithm is utilized to compute the isolation score of the unlabeled samples and to compute the similarity scores between the unlabeled samples and each of the clusters in order to detect the type of attack on the unlabeled samples. However, although the scheme can achieve the detection of malicious attacks with a small amount of labeled data, it is difficult to guarantee the accuracy of detecting attacks using unsupervised learning methods.
- **ZDG** [28]: To detect possible zero-day attacks in the network, the scheme builds an intrusion detection system using autoencoders and OCSVM, which first uses federated learning to train two autoencoders for detecting normal and anomalous traffic, respectively. Then, each edge base station trains two OCSVM classifiers based on local normal data and attack data. The final classification result is jointly determined by these four classifiers. Since OCSVM can recognize all anomalous traffic except normal traffic, the detection system also enables the detection of zero-day attacks. However, the simple integration of multiple classifiers does not always improve the accuracy of the detection system, and the joint detection of multiple classifiers can seriously reduce the detection efficiency.

The performance comparison results of the proposed scheme with the existing methods are shown in Table II. We can clearly

This article has been accepted for publication in IEEE Transactions on Mobile Computing. 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/TMC.2025.3582237



(a) Impact of threshold τ on accuracy (CICIOV) (b) Impact of threshold τ on accuracy (TON-IOT) (c) Impact of threshold τ on accuracy (EDGE-IIOT)

Fig. 9. Impact of threshold τ changes on detection accuracy in three datasets

observe that the proposed scheme obtains better detection results on the three datasets, and the detection precision and recall of the proposed scheme are higher than the other schemes. The reason for this phenomenon is that these systems use some unsupervised or semi-supervised learning algorithms to achieve the detection of unknown attacks, which leads to a decrease in the accuracy of the detection system. However, the proposed classifier based on 1D-CNN and a multi-attention mechanism is obtained based on fully supervised learning, which can identify known attacks accurately. In addition, we utilize reconstruction errors to enhance the quality of the activation vectors so that the multiple decision boundaries constructed based on the activation vectors can effectively isolate unknown attacks. Therefore, the designed unknown attack discriminator can efficiently recognize unknown attacks.

To compare the computational resource overhead of the proposed system with other systems, we deployed the detection systems trained using the three datasets on an Intel(R) Core i9-11900K 3.5GHz computer equipped with 64GB of RAM and measured the CPU usage of each detection system for 60 seconds continuously. The results of the computational resource overhead comparison are shown in Fig. 8 (a)-Fig. 8 (c). We can clearly observe that the CPU overhead sorted from low to high is ADPOAT, our proposed system, NMAH, and ZDG.

By analyzing the architecture of the detection system, we found that the main CPU overhead of ADPOAT lies in the use of the lightweight Kmeans clustering and Isolated Forest algorithms, thus obtaining a low computational overhead. However, the unsupervised learning property of Kmeans gives ADPOAT the worst detection performance. Moreover, the first reason for the high CPU overhead of NMAH and ZDG is the use of the OCSVM algorithm for malicious traffic detection. In large-scale datasets, the computational overhead of OCSVM's kernel function computation and optimization process grows rapidly with the number of samples. The second reason is that both NMAH and ZDG use multiple models to detect malicious traffic collaboratively. NMAH uses the OCSVM model, random forest model, and multilayer perceptron model to achieve collaborative detection of malicious traffic. ZDG uses two OCSVM models and two autoencoders to detect malicious traffic. Therefore, the computational overhead of NMAH and ZDG is higher than that of other systems. The computational overhead of our proposed detection model is much lower than that of NMAH and ZDG. This is because the main computational overhead of our system is incurred by the attack classifier, but the computational overhead of the attack classifier does not increase with the number of samples. In a comprehensive comparison, our proposed system is able to achieve the best detection performance while using less CPU resources.

G. Impact of threshold τ variation on detection accuracy (Q5)

In this subsection, we analyze the effect of different thresholds τ on the detection accuracy. It is worth mentioning that the threshold τ is a hyperparameter mentioned in Section IV.C. The results of the analysis are shown in Fig. 9. We can observe that the better thresholds τ for CICIOV, TON-IOT, and EDGE-IIOT are 0.25, 0.15, and 0.20, respectively. The overall accuracy of the detection system is lower when the threshold is less than the better thresholds. This is because thresholds designed to be smaller will result in more false alarms and mis-alarms in the detection system, leading to a decrease in overall accuracy. When the threshold is larger than the better thresholds, the unknown attack detection accuracy does not change much, while the detection accuracy of non-unknown attacks decreases slightly. The reason for this is that an increase in the threshold value indicates that a large number of samples use the detection results of the attack classifier as output. However, the attack classifier is unable to detect unknown attacks, which results in a large number of unknown attacks being incorrectly detected as non-unknown attacks and causes a decrease in accuracy.

VI. Discussion

A. Discuss Assumptions and Limits

1) System assumptions: We propose a number of system assumptions to enable the system to be applied in a real vehicular network. First, the vehicles possess sufficient computing and storage capabilities to run the intrusion detection system, and the system's operation does not interfere with the normal functioning of the vehicles. Second, RSUs have sufficient computing and storage capacity to train local intrusion detection systems. Then, all RSUs participating in the federated training are trusted, and no parameter poisoning

occurs. Finally, cloud servers involved in federated training do not maliciously corrupt received model parameters.

2) Possible limitations: In this study, we focus on addressing the issues of fast update and unknown attack detection for vehicular intrusion detection systems. However, our approach may have some potential limitations. The rapid movement of vehicles can lead to rapid changes in the network conditions, and the load changes on the RSUs are especially noticeable. When RSUs face higher vehicle densities or sudden data uploads, RSUs may be subject to higher computational and communication loads, which can reduce the efficiency of local training and even degrade the efficiency of global model aggregation. In our future work, to address the above issues, we consider introducing a network state-aware training scheduling mechanism to dynamically adjust the number of local training rounds. Meanwhile, we design hierarchical aggregation algorithms to help alleviate the computational and communication burden of RSUs and improve aggregation efficiency.

B. A discussion of privacy and security issues in federal learning

In this subsection, we discuss privacy and security issues related to federated learning and explore methods to prevent sensitive data from being exposed. During the federated learning process, although the raw data does not leave the local device, there are still multiple privacy and security threats, mainly including 1) gradient leakage, 2) model inversion attack, 3) identity inference attack, 4) parameter poisoning attack, and 5) parameter eavesdropping attack, etc.

To enhance the security of federal learning, we discuss how to prevent sensitive data exposure from four perspectives: 1) ensuring the trustworthiness of the entities involved in the training, 2) establishing secure communication channels, 3) the anti-traceability of the private data, and 4) adding noise to local model using differential privacy. First, to ensure that the devices participating in the training are trustworthy, it is necessary to develop authentication algorithms to verify the legitimacy of the devices. Second, to ensure the secure transmission of model parameters, encryption algorithms (e.g., AES) should be employed to encrypt and decrypt the transmitted parameters, thereby preventing eavesdropping and tampering. Then, when preprocessing vehicle traffic, the RSU needs to remove identifying information from the network traffic (such as vehicle ID, IP address, port number, etc.) to avoid identity inference attacks. Finally, differential privacy techniques are used to introduce controlled noise during local model updates, thus preventing attackers from inferring the sensitive data through gradients or parameters.

C. Discussion of broader applications

Our approach is designed with a focus on fast updating and robust detection of vehicular intrusion detection systems. In other networks (e.g., IoT or cloud computing) there may also be a need for detection system fast updating and robustness detection, so our approach can be extended to other networks under certain conditions. In IoT, end devices usually have weak computing power and high communication costs. Therefore, when applying our approach to IoT, we need to design a more lightweight attack classifier for end devices and use edge computing nodes as parameter aggregation centers in federated learning. In cloud computing, nodes have more adequate computing resources, but there are differences in network bandwidth, latency, and stability because nodes may be distributed in different geographic locations. Therefore, when applying our approach to cloud computing, additional network state-awareness algorithms need to be designed to solve the problems caused by network performance heterogeneity.

VII. Conclusion

In this study, we propose an intrusion detection system for vehicular networks that consists of an attack classifier and an unknown attack discriminator. To enable the proposed intrusion detection system to fully consider the network traffic characteristics of different regions, we design a federated learning algorithm based on representative client selection. The algorithm selects representative clients for model aggregation at each training iteration, thereby increasing the generalization ability of the detection system and accelerating detection system updates. Moreover, to address the threat of unknown attacks that may exist in open vehicular networks, we design an unknown attack discriminator based on extreme value theory and multilayer activation vectors. The multilayer activation vectors are generated by the classifier and input into the discriminator, which then analyzes whether the network flow is an unknown attack. Comprehensive experiments on three open datasets demonstrated that the proposed intrusion detection system can effectively detect both known and unknown attacks in vehicular networks and outperform state-of-the-art methods in terms of precision, recall, and F1 score.

References

[1] B. Fong, H. Kim, A. C. M. Fong, G. Y. Hong, and K. F. Tsang, "Reliability optimization in the design and implementation of 6g vehicle-to-infrastructure systems for emergency management in a smart city environment," *IEEE Communications Magazine*, vol. 61, no. 8, pp. 148–153, 2023.

[2] H.-W. Kao and E. H.-K. Wu, "Qoe sustainability on 5g and beyond 5g networks," *IEEE Wireless Communications*, vol. 30, no. 1, pp. 118–125, 2023.

[3] M. Ahmed, M. A. Mirza, S. Raza, H. Ahmad, F. Xu, W. U. Khan, Q. Lin, and Z. Han, "Vehicular communication network enabled cav data offloading: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 7869–7897, 2023.

[4] J. Kim, H. Chung, I. Kim, and G. Noh, "Integration of 5g mmwave-enabled v2i and v2v: Experimental evaluation," *IEEE Communications Magazine*, vol. 62, no. 1, pp. 104–110, 2024.

[5] A. Boualouache and T. Engel, "A survey on machine learning-based misbehavior detection systems for 5g and beyond vehicular networks," *IEEE Communications Surveys Tutorials*, vol. 25, no. 2, pp. 1128–1172, 2023.

[6] B. Lampe and W. Meng, "Intrusion detection in the automotive domain: A comprehensive review," *IEEE Communications Surveys Tutorials*, vol. 25, no. 4, pp. 2356–2426, 2023.

[7] V. P. Chellapandi, L. Yuan, C. G. Brinton, S. H. Żak, and Z. Wang, "Federated learning for connected and automated vehicles: A survey of existing approaches and challenges," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 119–137, 2024.

[8] J. Shu, L. Zhou, W. Zhang, X. Du, and M. Guizani, "Collaborative intrusion detection for vanets: A deep learning-based distributed sdn approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4519–4530, 2021.

[9] H. Liu, S. Zhang, P. Zhang, X. Zhou, X. Shao, G. Pu, and Y. Zhang, "Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 6073–6084, 2021.

[10] C. He, T. H. Luan, N. Cheng, G. Wei, Z. Su, and Y. Liu, "Federated learning based vehicular threat sharing: A multi-dimensional contract incentive approach," in *2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall)*, 2023, pp. 1–5.

[11] S. I. Popoola, G. Gui, B. Adebisi, M. Hammoudeh, and H. Gacanin, "Federated deep learning for collaborative intrusion detection in heterogeneous networks," in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, 2021, pp. 1–6.

[12] H. Sedjelmaci, N. Kaaniche, A. Boudguiga, and N. Ansari, "Secure attack detection framework for hierarchical 6g-enabled internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 2, pp. 2633–2642, 2024.

[13] A. A. korba, A. Boualouache, and Y. Ghamri-Doudane, "Zero-x: A blockchain-enabled open-set federated learning framework for zero-day attack detection in iov," *IEEE Transactions on Vehicular Technology*, pp. 1–16, 2024.

[14] L. Yang, A. Moubayed, and A. Shami, "Mth-ids: A multitiered hybrid intrusion detection system for internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 616–632, 2022.

[15] A. Bendale and T. E. Boult, "Towards open set deep networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1563–1572.

[16] S. Anbalagan, G. Raja, S. Gurumoorthy, R. D. Suresh, and K. Dev, "Iids: Intelligent intrusion detection system for sustainable development in autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 15 866–15 875, 2023.

[17] L. Wang, X. Zhang, D. Li, and H. Liu, "Multi-sensors space and time dimension based intrusion detection system in automated vehicles," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 1, pp. 200–215, 2024.

[18] S. Almutlaq, A. Derhab, M. M. Hassan, and K. Kaur, "Two-stage intrusion detection system in intelligent transportation systems using rule extraction methods from deep neural networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 15 687–15 701, 2023.

[19] Z. Deng, J. Liu, Y. Xun, and J. Qin, "Identifierids: A practical voltage-based intrusion detection system for real in-vehicle networks," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 661–676, 2024.

[20] Y. Xun, Z. Deng, J. Liu, and Y. Zhao, "Side channel analysis: A novel intrusion detection system based on vehicle voltage signals," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 6, pp. 7240–7250, 2023.

[21] M. Abdel-Basset, N. Moustafa, H. Hawash, I. Razzak, K. M. Sallam, and O. M. Elkomy, "Federated intrusion detection in blockchain-based smart transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 2523–2537, 2022.

[22] Z. Abou El Houda, H. Moudoud, B. Brik, and L. Khoukhi, "Blockchain-enabled federated learning for enhanced collaborative intrusion detection in vehicular edge computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 7, pp. 7661–7672, 2024.

[23] J. Cui, H. Sun, H. Zhong, J. Zhang, L. Wei, I. Bolodurina, and D. He, "Collaborative intrusion detection system for sdvn: A fairness federated deep learning approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 9, pp. 2512–2528, 2023.

[24] Z. Qu and Z. Cai, "Fedsa-resnetv2: An efficient intrusion detection system for vehicle road cooperation based on federated learning," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[25] M. Verkerken, L. D'hooge, D. Sudyana, Y.-D. Lin, T. Wauters, B. Volckaert, and F. De Turck, "A novel multi-stage approach for hierarchical intrusion detection," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 3915–3929, 2023.

[26] W. Zhang, L. Gao, S. Li, and W. Li, "Anomaly detection with partially observed anomaly types," in *2021 2nd International Conference on Computer Communication and Network Security (CCNS)*, 2021, pp. 98–107.

[27] C. Fan, J. Cui, H. Jin, H. Zhong, I. Bolodurina, and D. He, "Auto-updating intrusion detection system for vehicular network: A deep learning approach based on cloud-edge-vehicle collaboration," *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2024.

[28] P. Verma, N. Bharot, J. G. Breslin, D. O'Shea, A. Vidyarthi, and D. Gupta, "Zero-day guardian: A dual model enabled federated learning framework for handling zero-day attacks in 5g enabled iiot," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 3856–3866, 2024.

[29] S. Mohine, B. S. Bansod, R. Bhalla, and A. Basra, "Acoustic modality based hybrid deep 1d cnn-bilstm algorithm for moving vehicle classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16 206–16 216, 2022.

[30] A. Cura, H. Küçük, E. Ergen, and B. Öksüzoğlu, "Driver profiling using long short term memory (lstm) and convolutional neural network (cnn) methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 10, pp. 6572–6582, 2021.

[31] Y. Duan, N. Chen, S. Shen, P. Zhang, Y. Qu, and S. Yu, "Fdsa-stg: Fully dynamic self-attention spatio-temporal graph networks for intelligent traffic flow prediction," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 9, pp. 9250–9260, 2022.

[32] S. U. Stich, "Local SGD converges fast and communicates little," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=S1g2JnRcFX

[33] J. Zhang, A. Li, M. Tang, J. Sun, X. Chen, F. Zhang, C. Chen, Y. Chen, and H. Li, "Fed-cbs: A heterogeneity-aware client sampling mechanism for federated learning via class-imbalance reduction," in *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 2023, pp. 41 354–41 381. [Online]. Available: https://proceedings.mlr.press/v202/zhang23y.html

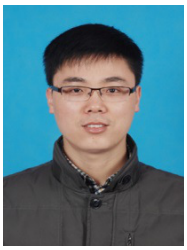
[34] E. C. P. Neto, H. Taslimasa, S. Dadkhah, S. Iqbal, P. Xiong, T. Rahman, and A. A. Ghorbani, "Ciciov2024: Advancing realistic ids approaches against dos and spoofing attack in iov can bus," *Internet of Things*, vol. 26, p. 101209, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660524001501

[35] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. den Hartog, "Ton_iot: The role of heterogeneity and the need for standardization of features and attack types in iot network intrusion data sets," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 485–496, 2021.

[36] 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.

[37] J. Yang, X. Chen, S. Chen, X. Jiang, and X. Tan, "Conditional variational auto-encoder and extreme value theory aided two-stage learning approach for intelligent fine-grained known/unknown intrusion detection," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3538–3553, 2021.

Chunyang Fan is currently a PhD student in the School of Computer Science and Technology, Anhui University, Hefei, China. His research focuses on the vehicular network intrusion detection and mobility management of the software defined vehicular networks.



Jie Cui (Senior Member, IEEE) was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 150 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics, IEEE Transactions on Cloud Computing and IEEE Transactions on Multimedia), academic books and international conferences.

Hulin Jin was born in Jilin Province, China, in 1977. He received his Ph.D. degree in Sejong University of Korea in 2013. He is currently a professor and Ph.D. supervisor of the School of Big Data at Anhui University. His current research interests include big data, computer vision, cloud networks and cloud computing. He has over 100 scientific publications.

Hong Zhong was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 200 scientific publications in reputable journals (e.g. IEEE Journal on Selected Areas in Communications, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Mobile Computing, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Multimedia, IEEE Transactions on Vehicular Technology, IEEE Transactions on Network and Service Management, IEEE Transactions on Cloud Computing, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics and IEEE Transactions on Big Data), academic books and international conferences.

Irina Bolodurina is currently a professor and head of Department of Applied Mathematics, at the Orenburg State University. She received her Ph.D. degree from South Ural State University. Prof. Irina Bolodurina has over 60 scientific publications in academic journals and international conferences which indexing in Scopus and WoS. She has participated in over 20 scientific projects supported by the RFBR and other Russian scientific programs. She's current research interests include theory of optimal control, mathematical modeling, information analysis software, control of social and economic systems, decision support systems, data integration, and processing.

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University, Wuhan, China in 2009. He is currently a professor of the School of Cyber Science and Engineering, Wuhan University, Wuhan, China and the Shanghai Key Laboratory of Privacy Preserving Computation, MatrixElements Technologies, Shanghai 201204, China. His main research interests include cryptography and information security, in particular, cryptographic protocols. He has published over 100 research papers in refereed international journals and conferences, such as IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Security and Forensic, and Usenix Security Symposium. He is the recipient of the 2018 IEEE Sysems Journal Best Paper Award and the 2019 IET Information Security Best Paper Award. His work has been cited more than 10000 times at Google Scholar. He is in the Editorial Board of several international journals, such as Journal of Information Security and Applications, Frontiers of Computer Science, and Human-centric Computing & Information Sciences.