electronics

MDPI

Article

FL-MAAE: An Intrusion Detection Method for the Internet of Vehicles Based on Federated Learning and Memory-Augmented Autoencoder

Ling Xing, Kun Wang, Honghai Wu *, Huahong Ma and Xiaohui Zhang

School of Information Engineering, Henan University of Science and Technology, Luoyang 471000, China; xinglingmy@haust.edu.cn (L.X.); 200320050343@stu.haust.edu.cn (K.W.); mhh@haust.edu.cn (H.M.); 9906117@haust.edu.cn (X.Z.)

* Correspondence: honghai2018@haust.edu.cn

Abstract: The Internet of Vehicles (IoV) is a network system that enables wireless communication and information exchange between vehicles and other traffic participants. Intrusion detection plays a very important role in the IoV. However, with the development of the IoV, unknown attack behaviors may appear. The lack of analysis and collection of these attack behavior has led to an imbalance in the sample data categories of the IoV intrusion detection, which causes the problem of low detection accuracy. At the same time, the intrusion detection model usually needs to upload data to the cloud for training, which will introduce the privacy risk due to of the leakage of vehicle users' information. In this paper, we propose an intrusion detection method for the IoV based on federated learning and memory-augmented autoencoder (FL-MAAE). We add a memory module to the autoencoder model to enhance its ability to store the behavior feature patterns of the IoV, make it robust to imbalanced samples, and use the reconstruction error as the evaluation index, so as to detect unknown attacks in the IoV. We propose a federated learning based training method for the IoV intrusion detection model. Local training of intrusion detection models in roadside units can effectively protect the privacy of data resources. We also designed an aggregation method based on the performance contribution of participants to improve the reliability of model aggregation. We conducted experiments on the NSL-KDD intrusion detection dataset to evaluate the performance of the proposed method. Experimental results show that our method has the best intrusion detection performance. In the case of contaminated samples, the accuracy and F1 score of the proposed method are 9.6% and 7.39% higher than those of the comparison methods on average.

Keywords: Internet of Vehicles; intrusion detection; network security; federated learning; autoencoder

Citation: Xing, L.; Wang, K.; Wu, H.; Ma, H.; Zhang, X. FL-MAAE: An Intrusion Detection Method for the Internet of Vehicles Based on Federated Learning and Memory-Augmented Autoencoder. *Electronics* **2023**, *12*, 2284. https://doi.org/10.3390/electronics12102284

Academic Editor: Suleiman Yerima

Received: 4 April 2023
Revised: 30 April 2023
Accepted: 16 May 2023
Published: 18 May 2023

1. Introduction

The Internet of Vehicles (IoV) is a vehicle-to-everything (V2X) interconnection with intelligent networked vehicles as the main information perception subject, and is designed to realize the interconnection between vehicles and people, roads, infrastructure, etc. As a new paradigm of the Internet of Things [1], the Internet of Vehicles can serve many applications, including assisted/autonomous driving, safety information sharing, and traffic control, among others.

In the IoV, vehicles generate a large amount of information (such as user information) [2] and exchange information about both themselves and other vehicles around them. In this environment, malicious vehicle nodes can easily affect the availability, integrity and confidentiality of the network. Some malicious nodes might send false information to deceive other vehicles, resulting in the disclosure of vehicle information and vehicle owner privacy information; moreover, some malicious nodes could forge multiple identities and use them to create false traffic scenarios, thereby affecting traffic order, disrupting the normal operation of the network, and threatening the safety of users' lives and property.

These malicious nodes pose a direct threat to the integrity of the data exchanged in the network. Moreover, while traditional network security protection mechanisms (such as firewalls and encryption) can usually prevent attacks from outside the network, they cannot effectively detect intrusions within the network (e.g., malicious behavior of legitimate users). Therefore, corresponding intrusion detection methods are required if intrusion behaviors are to be detected in the IoV.

Intrusion detection methods detect abnormal network behavior by monitoring the communication traffic of nodes in the IoV. Most research works regard intrusion detection as a classification problem of normal samples and attack samples. Most early intrusion detection schemes employed traditional machine learning methods—such as support vector machines [3], decision trees [4], etc.—to manually extract features for classification. However, these methods find it difficult to deal with the massive and multi-dimensional intrusion detection data associated with the IoV. Deep learning methods represented by neural networks have brought new ideas to the field of intrusion detection. Deep learning can effectively learn the inherent laws and representation levels of sample data. The nonlinear network structure, composed of multiple hidden layers, can adapt to the requirements of higher-dimensional learning and prediction, and thereby improve the intrusion detection performance. Various deep learning algorithms, such as the convolutional neural network (CNN), recurrent neural network (RNN), and generative adversarial network (GAN), etc., have been widely utilized in intrusion detection. For example, Zheng et al. [5] proposed a deep-learning-based intrusion detection method for the IoV, which uses a CNN to extract spatial range features from network traffic and LSTM to extract time-related features in order to detect malware traffic in the network. Alladi et al. [6] used sequence reconstruction and a threshold algorithm implemented via a deep neural network to determine whether a given message sequence in the IoV is real or abnormal.

However, there are some defects in the deep-learning-based intrusion detection method for the connected car network. Firstly, it is difficult to collect attack behavior data from the IoV. With the continuous development and expansion of the IoV, many new types of attack will appear [7,8]. This results in a lack of attack-related data in the dataset used to train the intrusion detection model, which reduces the intrusion detection performance due to the missed detection of attacks that have not yet appeared. Secondly, these schemes usually require devices to have powerful computing capabilities, so they are usually trained in the cloud, and vehicle-related data needs to be uploaded to the cloud. During this process, the user's privacy data, such as location information, behavior habits, and consumption records, are also uploaded, which can easily lead to user privacy leakage problems [9,10].

To address the above issues, this paper proposes a method for intrusion detection in the IoV based on federated learning and a memory-augmented autoencoder (FL-MAAE). The FL-MAAE uses an autoencoder model with a memory structure (called memory-augmented autoencoder) to enhance the storage capacity of the communication behavior characteristics of the IoV, and realize the storage and positioning of the vehicle's latent space features. The memory-augmented autoencoder only uses normal sample data during training. When the attack type and normal type of sample data are unbalanced, it will not interfere with model training, so it is robust to unbalanced samples. Due to the overfitting of the traditional autoencoder, the attack samples can be better reconstructed, resulting in an increase in the false negative rate. The method adds a memory module to enhance the modeling ability of normal communication behavior characteristics. The memory module will cause a large reconstruction error of the attack sample, and it is not easy to generate false positives, thus improving the detection accuracy. In order to prevent user privacy leakage during model training, we use the behavior data of local vehicles in the IoV to train the memory-augmented autoencoder model on the roadside unit based on federated learning. The model parameters are uploaded to the cloud server and based on the performance contribution of the participants. The global model is aggregated to ensure the security and privacy of the IoV data.

The main contributions of this paper can be summarized as follows:

- We propose a model called the memory-augmented autoencoder for IoV intrusion detection, which enhances the ability of the autoencoder network to characterize behavioral patterns by using a memory module, and design new reconstruction error and feature error loss functions for constraining the memory module, enabling efficient detection of intrusions and improving the accuracy of intrusion detection.
- We designed a federated learning-based intrusion detection model training method for the IoV. The local training of the intrusion detection model in the roadside unit can effectively protect the privacy of the data resources of the IoV. In addition, an aggregation method based on the performance contribution of the participants is designed to improve the reliability of model aggregation.
- The proposed FL-MAAE method was simulated on the NSL-KDD dataset, and its performance was evaluated in terms of accuracy, precision, recall, and F1 score. Comparisons with state-of-the-art methods (GAN, AE, FedAGRU) were also conducted.

2. Related Work

2.1. Internet of Vehicle Intrusion Detection

In this section, we discuss signature-based and anomaly-based IoV intrusion detection and deep-learning-based intrusion detection methods, as shown in Table 1.

In recent years, intrusion detection methods in the IoV have attracted increasing research attention. Intrusion detection methods can be broadly divided into signature-based and anomaly-based intrusion detection methods. Signature-based intrusion detection methods typically assume intruder activity, which can be represented by a pattern; the goal of the system is to determine whether the activity of the subject conforms to these patterns. Gao et al. [11] proposed a distributed DDoS intrusion detection approach, based on big data technology, that uses HDFS to store attack behavior data and the random forest algorithm to detect traffic in self-organizing vehicle networks in order to identify DDoS attacks. Zaidi et al. [12] used statistical methods to analyze traffic in the IoV and find malicious network nodes. This method uses hypothesis testing to model the intrusion detection problem and simulate it under different traffic conditions, achieving good performance in malicious node detection. Hoang et al. [13] proposed an intrusion detection system for in-vehicle networks, which uses an autoencoder to reduce the data dimension on the CAN bus, then employs a recurrent neural network combined with a SoftMax classifier to distinguish between attack data and normal data. Notably, although the above signature-based intrusion detection methods can effectively identify existing attacks in the signature database, they cannot detect unknown attacks and find it difficult to adapt to new attack behaviors. Many solutions to specific types of attacks, such as jamming attacks [14] and selective forwarding attacks [15], have been proposed. These specific attack types are of wide concern to researchers because they also pose a great security threat to the Internet of Vehicles.

Anomaly-based intrusion detection methods regard intrusion behavior as an abnormal value. After establishing a normal behavior model, the subject's current activity is compared with this normal behavior model. When the deviation from the normal contour model is large, the behavior is identified as intrusion behavior. In the IoV environment, most network traffic is normal behavioral traffic and lacks relevant attack tag data; therefore, anomaly-based intrusion detection methods that only need normal behavioral data to detect intrusions are a current research hotspot. Sedjelmaci et al. [16] designed a lightweight ad hoc network intrusion detection framework for vehicles, called AECFV. This approach adopts the packet drop rate (PDR), the packet send rate (PSR), the message duplication rate (MDR), and the signal strength intensity (SSI) to model normal behavior, then uses the SVM algorithm to train malicious vehicle node detection. This scheme has low computational overhead and a small false positive rate. Li et al. [17] also implemented an intrusion detection system for abnormal vehicle detection based on the SVM algorithm. This approach uses behavior and context information to train the SVM classifier by deploying on each

vehicle node to analyze the adjacent node behavior and exchange information, and also employs Dempster—Shafer evidence theory fused node data. Focusing on the intrusion problem in the CAN bus of the in-vehicle network, Levi et al. [18] adopted a hybrid machine learning method. First, the normal behavior of the vehicle is learned by training a hidden Markov model, after which a regression model is used to adjust the A threshold for anomaly prediction. This adaptive threshold achieves better detection performance than the previous static threshold.

Table 1. Comparison of different IoV intrusion detection methods.

Categories	Research Work	Contribution	Limitation
Signature-based	Gao et al. [11]	Using HDFS to store DDoS attack behavior data	Only DDoS attacks can be detected
	Zaidi et al. [12]	Analyzing IoV attack traffic using hypothesis testing	When the amount of data is large, the processing time is longer and unknown attacks cannot be discovered
	Hoang et al. [13]	Using Softmax classifier to distinguish normal and attack data on CAN bus	Applies only to periodic messages, not aperiodic messages
Anomaly-based	Sedjelmaci et al. [16]	Four features, PDR, PSR, MDR, and SSI, are used to simulate normal vehicle behavior and detect attacks through cluster heads	The cluster head node is selected only according to the mobility and trust of the node, which may be controlled by an attacker
	Li et al. [17]	The behavior and context information of normal vehicle nodes are analyzed to detect attacks using SVM	Lack of public dataset validation
	Levi et al. [18]	Train a hidden Markov model to learn the normal behavior of the vehicle and adaptively adjust the detection threshold using a regression model	When the data dimension is high and the data volume is large, the advantage is not obvious
Deep-learning-based	Hu et al. [19]	Increase the diversity of features based on the split convolution module	Poor detection accuracy and execution efficiency on small samples
	Park et al. [20]	Building twin convolutional neural networks using small-sample learning methods to extract time-dependent dynamic features in traffic	Relatively low detection performance
	Zhou et al. [21]	Proposed an incremental long short-term memory-based method to detect attacks	Weak detection of stealth attacks
	Ashraf et al. [22]	Detecting attacks based on long short-term memory networks and autoencoders	Serial spatio-temporal feature extraction is susceptible to the impact of the previous sub-model
	Liu et al. [23]	Collaborative intrusion detection based on blockchain and federated learning	High communication overhead
	Chen et al. [24]	Increase the weight of important nodes based on the attention mechanism to reduce the overhead of federated intrusion detection	Depends on the consistency and stability of the global model
	Attota et al. [25]	Improving learning efficiency for different classes of attacks using multi-view ensemble learning	computational and communication overheads impact nodes

2.2. Intrusion Detection Based on Deep Learning

Traditional machine learning methods suffer from dimensional disasters in intrusion detection and have limitations in the face of massive multidimensional IoV data. Deep learning can be more efficient in analyzing and extracting the behavioral data of the IoV and

reflecting the essential features of the behaviors; therefore, this has motivated researchers to implement intrusion detection using deep learning methods such as autoencoders, convolutional neural networks, long and short-term memory networks, etc. Many emerging technologies such as deep learning [26], mobile edge computing [27], etc., have been applied to solve the network security problems of the Internet of Things [28,29]. These methods also provide ideas for intrusion detection of the Internet of Vehicles. Hu et al. [19] implemented a wireless network intrusion detection system using adaptive synthetic sampling and a convolutional neural network implemented by a split convolutional module to increase the diversity of spatial features and reduce the impact of inter-channel information redundancy on the model. Park et al. [20] convert network traffic into grayscale maps and build twinned convolutional neural networks based on small sample-learning methods to determine the type of attack based on the similarity scores of attack samples. To capture the time-dependent dynamic features in network traffic, Zhou et al. [21] proposed an incremental long and short-term memory network intrusion detection method that introduces state changes into LSTM and processes dynamic information in network data by obtaining the state of the LSTM implicit layer. Ashraf et al. [22] combined LSTM and a self-encoder to extract the network traffic of the vehicular network from temporal features to improve the accuracy of intrusion detection in vehicular networks.

However, in deep learning methods, vehicle data usually needs to be collected and trained on the cloud server, which may easily result in the leakage of private data. Federated learning can perform joint model training provided that all of the multiple participants meet certain data privacy conditions. This is an effective method of solving the problem of data silos that has been proposed in recent years. Accordingly, some researchers have applied it to the field of intrusion detection. Liu et al. [23] proposed a coordinated intrusion detection method based on blockchain and federated learning for edge computing in vehicles. Under this approach, the vehicle performs distributed training of the intrusion detection model, then uploads the model parameters to the blockchain, composed of RSU. Model aggregation is carried out on the platform, after which the training and updating of the global intrusion detection model is completed collaboratively without revealing private data. Chen et al. [24] proposed an intrusion detection method for wireless edge networks based on federated learning, applying an attention mechanism to increase the weights of important devices in order to avoid unnecessary updates being uploaded to the cloud server. This scheme successfully reduces the large communication overhead. Attota et al. [25] introduced a federated-learning-based intrusion detection method, MV-FLID, which trains multiple views of IoT network data in a decentralized manner so as to detect, classify, and defend against attacks. However, the intrusion detection model of the above method is deployed on the vehicle, and the model training brings a lot of computational overhead to the vehicle. At the same time, when performing model aggregation, the conventional model aggregation method aggregates according to the local data volume of the participant as the weight. When there is a participant whose dataset is too large, it will affect the model aggregation. Therefore, this paper proposes that the roadside unit should be responsible for the training of the intrusion detection model; at the same time, based on the performance of the local model, the contribution of the roadside unit to the model training is calculated to solve the above problems.

Compared to the existing intrusion detection methods in the Internet of Vehicles, the advantages of the method proposed in this article are that (1) the method is anomaly-based and can detect unknown attacks; (2) it performs better in detection performance; and (3) it can avoid leakage of user privacy.

3. System Model and Threat Model

3.1. System Model

The system model in this paper is built on the federated learning framework, which can be broadly divided into two stages. In the first stage, the roadside unit collects the vehicle communication behavior data in the area, constructs the dataset for the training

of the local intrusion detection model, and uses the data downloaded from the cloud. The initial model is trained locally. In the second stage, each roadside unit uploads the trained model to the cloud, which aggregates the models uploaded by multiple roadside units to obtain a global model. The cloud server then sends the aggregated global model to the roadside unit to detect malicious vehicles in the area. As shown in Figure 1, it is mainly composed of the following three entities:

- Vehicle: The vehicle is the main information perception body in the IoV, and the equipped on board unit (OBU) can communicate with the vehicle or roadside unit through dedicated short range communication (DSRC) or LTE-V technology. The generated communication behavior data will be transmitted to nearby roadside units for the training of the intrusion detection model.
- Roadside Unit: The roadside unit is a fixed device deployed along the road. Each unit is equipped with an edge computing server nearby and has strong computing and communication capabilities. The roadside unit, as a gateway node for information sharing in the IoV, is used to collect the communication behavior data of vehicles in its area to train the local intrusion detection model and interacts with the central server based on a federated learning approach to update the model. The trained intrusion detection model will be deployed on the roadside unit to detect the vehicle data in the coverage area, which enables the intrusion vehicle nodes to be identified.
- Cloud Server: The cloud server is mainly used for federated learning model aggregation. It operates by aggregating the uploaded model parameters according to the contributions made by roadside units participating in the training and establishing a global intrusion detection model, which facilitates optimization of the model, the cloud server, and each of the multiple rounds of interaction between roadside units. After the model converges, the cloud server sends the model parameters to each roadside unit participating in the federated training, thereby updating the local intrusion detection model.

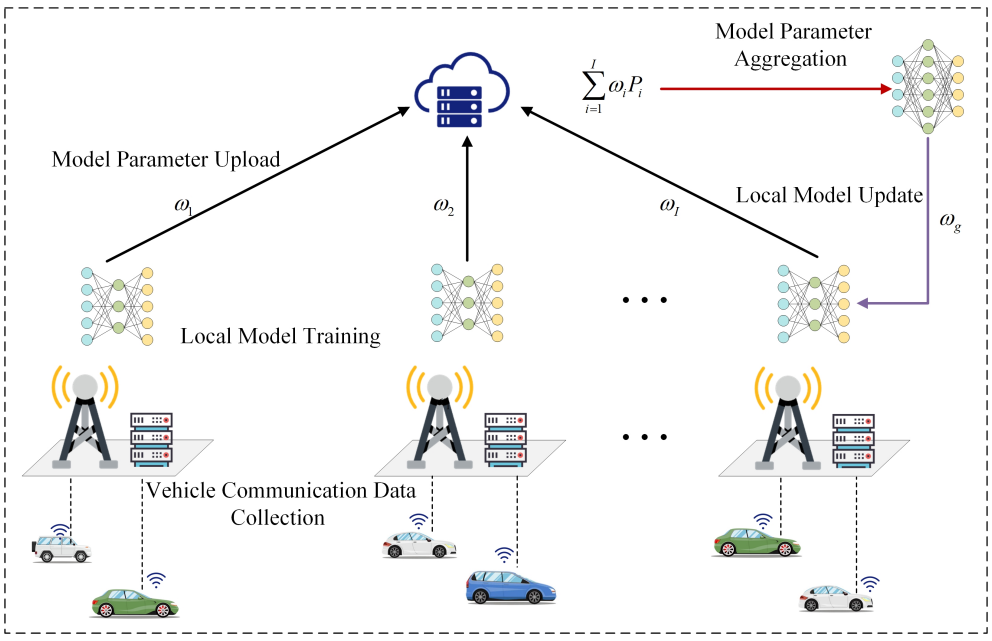

Figure 1. System model.

In the IoV environment, vehicles and roadside units establish wireless connections through wireless communication technologies (such as DSRC or LTE-V) to improve data access services for high-speed moving vehicles, as well as to realize information exchange between vehicles and roadside units. In the model presented in this paper, the roadside unit collects and stores the vehicle communication behavior data with the help of the connected

edge computing server, which is used to train the intrusion detection model. At the same time, the roadside unit collects the vehicle communication behavior data in the area to detect whether there are malicious vehicle nodes sending information.

In order to ensure the security of the local data of the roadside unit, the roadside unit uses this local data to train the intrusion detection model received from the cloud server, only then does it upload the trained model parameters to the cloud. Notably, it does this without uploading the local training data, thus ensuring the security and privacy of this data. The cloud server updates the global model according to the contribution made by each roadside unit participating in the training, then sends the updated global model to the roadside unit.

3.2. Threat Model

In this section, we mainly consider the threat of internal cyber-attacks in the IoV. These include the following typical attack behaviors:

- DoS attack: Malicious vehicle nodes use reasonable service requests to occupy an excessive amount of network resources, thereby preventing service providers from providing normal services. For example, a malicious vehicle may launch a DoS attack on the RSU, exhausting its computing and communication resources and causing it to stop responding or even crash.
- Sybil attack: Malicious vehicle nodes forge information corresponding to multiple identities and broadcast multiple pieces of forged node information to the network, allowing them to gain network control, interfere with queries, and deny responses and other permissions. For example, malicious vehicle nodes can create the illusion that there are many vehicles on the road, so that other normal vehicles will mistakenly assume that traffic on this road is congested. Through a false identity attack of this kind, attackers can inject misinformation into the network, causing confusion.
- Sinkhole attack: A malicious vehicle node creates a "hole" centered on itself, attracting data packets from surrounding vehicles to pass through. In this way, the passing data packets can be modified or deleted. This attack can spread false information or cause information to be lost, interfering with normal network behavior.
- Broadcast tampering attack: Malicious vehicle nodes tamper with relevant security information or service information in the IoV and then publish this information. For example, malicious nodes could tamper with their own basic safety message (BSM) and then publish it to affect traffic order. It is also possible to broadcast false service information to attract other vehicles to visit a particular region.

The typical IoV attack methods presented above usually exhibit network behaviors that differ from those of normal nodes, meaning that their communication behavior and traffic statistics characteristics can be used as important attributes to distinguish normal behavior from attacks. However, new types of cyber-attacks continue to emerge in an endless stream, such as zero-day attacks, the attack modes of which are currently unknown and difficult to detect. If the detection is carried out directly, there will be a high rate of both false positives and false negatives. In contrast, the federated-learning-based intrusion detection method outlined in this paper can update the proposed model according to the vehicle communication behavior information regularly collected by RSU, enabling the detection accuracy of the model to be continuously improved.

Since the communication behavior data of the IoV can reflect the behavior information of vehicle nodes, it is possible to collect network traffic, audit information and other related data for analysis, and establish an intrusion detection model to discover network intrusion behavior. Next, we formally describe the intrusion behavior and intrusion detection in the IoV.

We represent the intrusion behavior in the IoV as $I = < A, E, T, N, R >$, where A is the intruder who initiates the attack, E is the device node used by the intruder to initiate the attack, T is the target node being attacked, which can be a vehicle or a roadside unit, N is the network region where the attack takes place, and R is the result produced after one attack. Its primary intrusion can be expressed as the intruder A initiates an attack on the target node T in the network N using its attack node E to achieve the attack result R.

For an intrusion $i \subseteq I$, the detection of this intrusion can be expressed as $ID = [f, rules]$, where rules is the set of conditions for intrusion detection, f is the result of intrusion detection, divided into normal and attack results, and f is expressed as follows:

$$f[rules, D] = \begin{cases} attack, \ i \in rules \\ normal, \ i \notin rules \end{cases} \quad (1)$$

where D is the set of behaviors in a certain time if there exists an intrusion i, $\exists E \rightarrow T$, which can be represented by using $D[i, time]$. When $\forall i, i \in I, r \in rules$ and $f[r, D] = normal$, i.e., a missed alarm is generated, and when, $\forall i, r \notin rules$ and $f[r, D] = attack$, i.e., a false alarm is generated. For intrusion detection, when $\forall i, i \in I$ or $\forall i', i \notin I$ both have $f[r, D] = attack$, that is, the attack is detected.

4. The Proposed FL-MAAE Method

4.1. Memory-Augmented Autoencoder Model

Currently, autoencoders are widely used in intrusion detection. By training autoencoders with normal data, autoencoders can have a large reconstruction error for data that produces anomalies, thus achieving intrusion detection. However, autoencoders can sometimes also achieve good reconstruction of abnormal data, leading to intrusion omission and reducing intrusion detection performance. To address this issue, we propose a model called memory-augmented autoencoder to implement intrusion detection in the IoV. By adding a memory module to the original autoencoder to store and represent vehicle communication behavior data, we train the memory module based on the feature reconstruction loss function using normal vehicle communication behavior data, so that the memory module saves the latent features of normal vehicle communication behavior. When attack behavior data appears, only the latent space features of normal behavior can be queried with the potential space features in the memory module. By calculating the reconstruction error between the query feature-reconstructed data and the attack data, which is significantly higher than the normal reconstruction threshold, the attack can be detected. The proposed storage structure expands the reconstruction error of attack data input to the autoencoder, thus effectively avoiding the omission of attack data. As shown in Figure 2, the memory-augmented autoencoder model consists of three parts: namely, the encoder, decoder, and memory modules. The preprocessed vehicle communication behavior data is first input into the model to obtain the latent space features, after which the memory module is used to store and locate latent space features, and finally to use the reconstruction loss function so that the model can reconstruct the original feature vector. In the model training stage, we only use the sample data marked as normal for training; this means that the memory module only stores the potential features of the normal sample data, so that when performing intrusion detection, the attack behavior sample data is input into the model and matched based on the data from the a memory module. If the feature is reconstructed and output, the reconstruction error will be significantly higher than that of the normal sample; thus, it is determined that the sample in question represents attack behavior. Table 2 lists the utilized notations and variables.

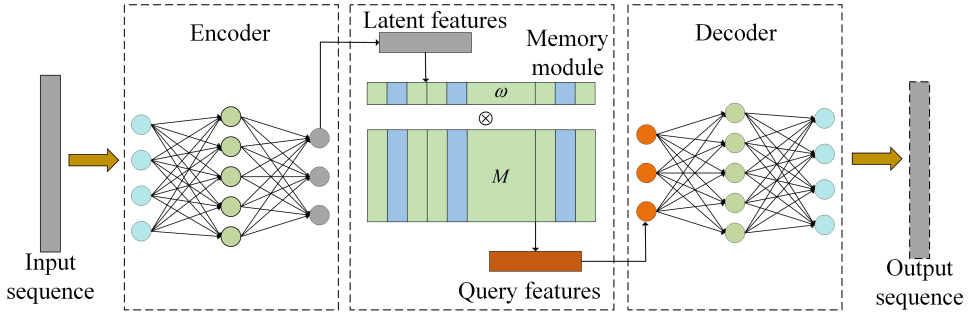

Figure 2. Memory-augmented autoencoder model architecture.

Table 2. List of notations.

Notation	Description
x and $\hat{x}$	Input and output of MAAE
z and $\hat{z}$	Input and output of memory module
R^D	Latent space feature dimensions
θ_E and θ_D	Parameters of encoder and decoder
K	Number of memory items
m_i	i-th memory item
$exp(\cdot)$	Exponential operation
$\rho(\cdot)$	Calculate Pearson similarity
ω_i	Memory item weight
$R(x, \hat{x})$	Sample reconstruction loss
$FR(z, \hat{z})$	Feature reconstruction loss
R_T	Overall loss function threshold
R_{min}	Minimum sample reconstruction loss
τ	Preset intrusion judgment parameters
D_i	Local dataset
P_i	Model performance contribution
ω_g	Global model parameters
ω_i^e	i-th local model parameters

4.1.1. Encoder

The encoder is used to encode the input communication behavior data into communication behavior features in the latent space, and the specific data contained in the communication behavior data are based on the actual setting. The encoder of this model is implemented by a four-layer fully connected neural network, which is used to map the input data features to the latent space. The encoder can be expressed by the following formula:

$$z = E(x; \theta_E) \tag{2}$$

The communication behavior data $x \in R^N$, and N are the dimensions of the input data, $z \in R^D$ is the communication behavior feature in the latent space after the data are input to the encoder, D is the dimension of the communication behavior feature, and θ_E is the model parameter of the encoder.

4.1.2. Memory Module

The memory module is used to store and locate the latent space features of the data. It would be more difficult to store the latent space features directly. Here we use multiple memory items instead. During training, z matches the combination of memory items with the highest similarity, and training optimizes the memory items so that the normal behavior data can be reconstructed better. In contrast, when performing intrusion detection, the anomalous data will match the combination of stored items with normal behavior, resulting in a larger reconstruction error to detect the intrusion. We use the matrix $M \in R^{K \times D}$ to represent this, where D is the dimensionality of the feature and K is the

number of memory items. The specific latent space features stored in the memory items are denoted using m_i. The latent space features z obtained by encoding the encoder are input to the memory module M. The features $\hat{z}$ are obtained by applying specific combinations of memory terms as shown in the following equation:

$$\hat{z} = \omega M = \sum_{i=1}^{K} \omega_i m_i \tag{3}$$

where ω_i is the weight of each memory item to indicate the similarity between the latent space feature z and the memory item m_i, which is calculated using the softmax method to derive ω_i with the following formula:

$$\omega_i = \frac{exp(\rho(z, m_i))}{\sum_{j=1}^{K} exp(\rho(z, m_j))} \tag{4}$$

where $\rho(\ )$ is the similarity calculation formula. The conventional similarity calculation usually uses cosine similarity, but its cosine calculation cannot be performed when the dimension is missing, so we propose to use the Pearson correlation coefficient to calculate the correlation degree between latent space features and memory items. The value of the coefficient is always between -1 and 1; close to 0 is said to have no correlation, close to 1 or -1 is said to have strong correlation, and the $\rho(\ )$ calculation formula is as follows:

$$\rho(X, Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{5}$$

The features $\hat{z}$ obtained by matching from the memory module are input to the decoder, and the output $\hat{x}$ similar to the original input data x is reconstructed from the latent features $\hat{z}$. According to the above description, this model reconstructs the communication behavior feature z using multiple memory items, and optimizes the memory items through training so that the normal communication behavior data can be better reconstructed. When intrusion detection is performed, abnormal communication behavior data will be matched to the combination of memory items that store normal behavior, resulting in a large reconstruction error, thereby realizing intrusion detection.

4.1.3. Decoder

The decoder is used for reverse coding according to the reconstructed communication behavior characteristic $\hat{z}$ to obtain the reconstructed data $\hat{x}$ with the same size as the original communication behavior data x. The specific structure of the decoder is set symmetrically according to the encoder. The data decoding process of the decoder can be expressed by the following formula:

$$\hat{x} = D(\hat{z}; \theta_D) \tag{6}$$

where, θ_D is the decoder parameter.

4.1.4. Loss Function

The model is trained by two loss functions: sample reconstruction and feature reconstruction. The sample reconstruction loss is used to calculate the difference between the input communication behavior data and the reconstruction data, and is calculated using the square error function; the calculation formula of sample reconstruction loss $R(x, \hat{x})$ is as follows:

$$R(x, \hat{x}) = \frac{1}{N} \sum_{n=1}^{N} (x_n - \hat{x}_n)^2 \tag{7}$$

where N is the number of input samples, x_n and $\hat{x}_n$ are the n-th input sample and the corresponding reconstructed output sample samples, respectively. We use the sample reconstruction loss as a criterion to determine whether a sample is an attack sample or not.

In order to train and optimize the memory module in the model, this paper proposes a feature reconstruction loss function, which is used to reduce the error between the memory input and output, and improve the memory's ability to characterize normal behavior characteristics. The calculation formula of feature reconstruction loss $FR(z, \hat{z})$ is as follows:

$$FR(z, \hat{z}) = \frac{1}{I} \sum_{i=1}^{I} (z_i - \hat{z}_i)^2 \tag{8}$$

where I is the feature dimension, z_i and $\hat{z}_i$ represent the i-th element value of communication behavior feature z and reconstruction communication feature $\hat{z}$, respectively. When there is a difference between z and $\hat{z}$, it represents a bias in the representation of the behavior feature by the memory module. Therefore we use the feature reconstruction loss for reducing the existence of discrepancy between z and $\hat{z}$, thus improving the ability of the memory module to characterize the features. The total loss function of the model can be expressed as the following formula:

$$L = R(x, \hat{x}) + \lambda FR(z, \hat{z}) \tag{9}$$

where λ is the parameter used to balance the two loss functions. Convergence of the total loss function is achieved by continuous training of the model. After the training is completed, the following method is used to calculate the reconstruction loss threshold:

$$R_T = R_{min} + \tau \tag{10}$$

Among them, R_{min} represents the minimum value of the sample reconstruction loss during the training process, and τ represents the preset hyperparameter, which is set according to actual needs.There is a difference between unknown attack behavior and normal behavior traffic, so it is difficult for anomalous samples to be reconstructed by the decoder again after encoder downscaling. This is due to the fact that in a batch of training data, most of the network parameters are trained by normal samples, and the reconstruction error of anomalous samples will be much larger than that of normal samples, so we use the reconstruction error to detect the attack behavior. When performing intrusion detection, the abnormal judgment of the sample reconstruction loss $R(x, \hat{x})$ is performed according to the reconstruction loss threshold R_T. To obtain the vehicle intrusion detection results, the specific method is as follows: when $R(x, \hat{x}) \leq R_T$, the corresponding communication behavior data is normal data; when $R(x, \hat{x}) > R_T$, the corresponding communication behavior data is abnormal data, and the corresponding communication behavior data will be sent. The vehicle is identified as an intrusion vehicle.

4.2. Federated Learning Training

At present, many vehicle networking intrusion detection methods usually need to collect vehicle communication behavior data, and train intrusion detection models on the cloud server side. However, the vehicle communication behavior data contains the user's private information, and uploading these data to the cloud introduces the risk of privacy leakage. Therefore, we train an intrusion detection model based on federated learning. First, the roadside unit is trained locally using its own data and a local model is obtained. Then, multiple roadside units only upload the parameters of the model to the cloud for model aggregation. Finally, the cloud server sends the aggregated model to the roadside unit. The above steps are repeated until the intrusion detection model converges. Since we train the intrusion detection model based on federated learning, there is no need to transmit or share data with other parties or cloud servers; rather, the model is trained locally and only model parameters or gradients are uploaded, thereby avoiding the risk of user privacy disclosure.

4.2.1. System Initialization

The cloud server selects the initialization model parameters ω, loss function L, learning rate η, batch size B, and other related parameters for the intrusion detection model. We also preset the number of iteration rounds of the cloud server and roadside unit E.

4.2.2. Local Model Training

The roadside unit receives the initial model parameters ω, loss function L, and learning rate η from the server, and uses the local dataset D_i for training. The local dataset is formed by the roadside unit collecting the vehicle communication behavior data of the coverage area. This includes the following: basic safety message (BSM)—safety status data exchanged between vehicles, including position, speed, heading, and other related information; network data information—key network information generated by vehicle communication, including network protocol type, source address, destination address, timestamp, and other information; traffic statistics information—statistical feature information of the data sent by roadside units to vehicles, including data packet transmission rate per unit time, data packet discard rate, and other information. Roadside units collect this data to form local datasets. Each dataset is required to contain only normal vehicle communication behavior data to facilitate its modeling of normal vehicle communication behavior. During model training, the data needs to be standardized for comprehensive evaluation. In this paper, the z-score algorithm is used for standardized processing. The formula is as follows:

$$z = \frac{x - \mu}{\sigma} \tag{11}$$

where x is the input data, μ is the mean of the input data, and σ is the standard deviation of the input data. Moreover, the character features in the dataset are encoded and converted into numerical features using LabelEncoder in the sklearn library, after which the processed data features are converted into matrices to facilitate the processing of these data by the intrusion detection model. Since the roadside unit is trained with the help of the edge server, it can be assumed that the roadside unit has sufficient computing power without considering the computational complexity of the algorithm.

4.2.3. Model Parameter Aggregation

Traditional federated learning usually uses the FedAvg algorithm to perform weighted average aggregation of the model according to the proportion of the data volume of the participants participating in the federated training to the total training volume [30–32]. However, in the IoV environment, the local datasets collected by each roadside unit are often uneven, meaning that the final aggregated model will bias the roadside unit with a large amount of data. For this reason, this paper proposes an aggregation method that considers the contribution made by the local model performance. The aggregation method based on the performance contribution sets a verification dataset on the cloud server side. Whenever the roadside unit uploads the model parameters to the cloud server, these parameters are used to restore the local model, and the model is verified with the verification dataset to obtain the detection performance index. We use the detection performance of the model on the validation set as the model weight, and aggregate the local models for all roadside units. This method focuses on the detection performance of the local model itself, and assigns higher weights to the models with high detection performance for aggregation. What the model has learned from the data has a higher degree of reliability for solving the intrusion detection task. The model parameters uploaded by the roadside unit are first tested on a verification dataset on the cloud server, from which four performance evaluation parameters of the model are obtained: accuracy, precision, recall, and F1 score. Aggregation based on model performance contribution is more reliable than aggregation based on data volume alone. The calculation formula for performance contribution is as follows:

$$P_i = \frac{Acc_i + Pre_i + Rec_i + F1_i}{Total(Acc + Pre + Rec + F1)} \tag{12}$$

Here, Acc_i, Pre_i, Rec_i, and $F1_i$, respectively, represent the accuracy, precision, recall, and F1 score of the model uploaded by roadside unit number i on the validation dataset located on the server, while $Total(Acc + Pre + Rec + F1)$ represents the sum of the performances of the four performance indicators (accuracy, precision, recall, and F1 score) of the models uploaded by all roadside units.

The cloud server receives the local model parameters uploaded by each roadside unit and performs weighted aggregation according to the performance contribution of each roadside unit, thereby obtaining the global model parameters ω_g. The aggregation method is as follows:

$$\omega_g = \sum_{i=1}^{I} \omega_i P_i \tag{13}$$

Here, I denotes the number of roadside units participating in federated training, ω_i represents the model parameters uploaded by the i-th roadside unit, and P_i is the performance contribution of the i-th roadside unit.

4.2.4. Local Model Updating

The roadside unit receives the aggregated global model parameters ω_g sent by the cloud server, then updates the local intrusion detection model according to ω_g. The updated expression of the local model is as follows:

$$\omega_i^e \leftarrow \omega_g \tag{14}$$

where ω_i^e is expressed as the model parameter of the number of iterations in the e-th round of the i-th roadside unit. The above steps are iterated multiple times until the optimal global intrusion detection model is obtained.

The federated learning framework allows RSUs to retain datasets locally and train deep learning models in collaboration with other RSUs, thereby ensuring that no third party has access to raw vehicle data. In this way, federated-learning-based models can achieve intrusion detection without compromising privacy. Details of the FL-MAAE algorithm are presented in Algorithm 1.

Algorithm 1 FL-MAAE Algorithm.

Input: E: number of iteration rounds; R_i: roadside unit; D_i: local dataset; T: test dataset;
Output: classification result of test set T;

server initialization defines initial model parameters ω, loss function L, learning rate η;
calculates the model contribution $P_i = \frac{Acc_i + Pre_i + Rec_i + F1_i}{Total(Acc + Pre + Rec + F1)}$;
$e = 1$;
while $e \leq E$ **do**
    if $i \in I$ **then**
        input x into the encoder to get the latent space feature z;
        input the latent space feature z into the memory module for query and get $\hat{z}$;
        take the $\hat{z}$ input to the decoder and obtain the reconstructed output $\hat{x}$;
        calculate the loss function L;
        backpropagate L to update the model;
    end if
    R_i uploads local model parameters ω_i to cloud server;
    $\omega_g = \sum_{i=1}^{I} \omega_i P_i$;
    for $i \in I$ **do**

Algorithm 1 *Cont.*

$\omega_i^e \leftarrow \omega_g$;
end for
$e \leftarrow e + 1$;
end while
for sample data in test set T **do**
test samples x_t are input into the model to get $\hat{x}_t$;
calculate the reconstruction error $R(x_t, \hat{x}_t)$;
judge whether the data is intrusion sample data according to the threshold;
end for

5. Performance Evaluation

In this section, extensive experiments are carried out to evaluate the performance of the proposed scheme. First, the experimental settings are presented, including environment settings, dataset descriptions, and performance metrics. Next, the method proposed in this paper is compared with some current state-of-the-art methods.

5.1. Experimental Setup

The simulation experiments are designed and implemented in the Python language. The federated learning framework is implemented based on the FedML [33] open-source library, and the memory-augmented intrusion detection model is implemented using PyTorch [34]. We use random search to optimize model parameters. Within a given range of hyperparameters, we randomly select a set of hyperparameter combinations, then evaluate their performance, and finally select the optimal group. The specific parameter settings of the model are shown in Table 3.

Table 3. Main parameters of FL-MAAE method.

Parameter	Value
Encoder layer	4
Decoder layer	4
Memory item	15
Learning rate	0.001
Latent feature dimension	16
Optimizer	Adam
Client node number	10

5.2. Dataset Description

This experiment uses the NSL-KDD dataset [35] to verify the effectiveness and performance of the proposed method. NSL-KDD is one of the most commonly used datasets in the field of network intrusion detection. In NSL-KDD, each piece of network traffic data consists of 42-dimensional features: of these, 38 dimensions are numeric features, 3 are character features, and the last dimension is labeling features. The first 41-dimensional features can be divided into basic features, content features, and traffic features. Basic features contain the basic information of each piece of traffic data, while content features are primarily the payload information of data packets, which can be used to represent network behavior information. For their part, traffic characteristics present certain traffic statistics, such as time-based or host-based traffic statistics. We normalized these features using the z-score algorithm and used the processed data sequence for model training and validation. The final one-dimensional feature marks the traffic data as either normal or attack data. There are four types of attacks: DoS, Probe, R2L, and U2R. We unify the four attack types as either attack samples (marked as 1) or normal sample data (marked as 0).

We extract 70% of the data from the dataset to form the training dataset, while the remaining 30% is the test dataset. In the training phase, the training dataset is equally

divided across each RSU participating in federated learning, and only data with normal labels is used for FL-MAAE model training.

5.3. Performance Metrics

The output of intrusion detection can be regarded as binary classification. There are four outcomes: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). The proposed method is evaluated using four evaluation metrics: accuracy, precision, recall and F1-score, which are calculated as follows:

Accuracy: The proportion of correctly classified samples to all samples, which can be expressed as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{FN}} \tag{15}$$

Precision: The proportion of samples classified as attack samples that are in fact attack samples, which can be expressed as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{16}$$

Recall: The proportion of all attack samples that are correctly classified as attack samples, which can be expressed as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{17}$$

F1 score: The weighted average of precision and recall, which can be expressed by as follows:

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

5.4. Comparative Analysis

The proposed FL-MAAE method is compared with some state-of-the-art baseline methods, which include the following: the generative adversarial network (GAN)-based distributed IoT intrusion detection method proposed by Ferdowsi et al. [36], which proposes a distributed GAN architecture that learns abnormal behavior data to detect intruding nodes; the autoencoder-based intrusion detection method proposed by Faber et al. [37] for cloud and mobile environments, which uses an autoencoder neural network to implement anomaly-based intrusion detection methods, which it implements based on time window features and network flow; the intrusion detection method for wireless edge networks based on federated learning and an attention-gated recurrent unit (FedAGRU) proposed by Chen et al. [24], which adds an attention mechanism to the gated recurrent unit to improve the weight of important devices, so that only the important weights are updated to the cloud server. We reproduced the above schemes, then compared and analyzed their intrusion detection performance with that of FL-MAAE under the federated learning framework. As Table 4 shows, most of FL-MAAE's intrusion detection performance indicators are superior to those of the other three methods. In terms of accuracy, FL-MAAE is 1.39% better than the AE scheme proposed Faber et al., reaching 97.22%. This occurs because, compared with the AE-based intrusion detection method, the memory module we added to the AE model enhances the model's ability to represent network behavior. In terms of precision, FL-MAAE performs slightly worse than FedAGRU, while in terms of recall, FL-MAAE is 2.41% better than FedAGRU, reaching 95.73%. At the same time, FL-MAAE also outperforms all compared methods in terms of F1 score, which effectively takes into account both the precision and recall, indicating that the model quality is higher.

Table 4. Comparison of intrusion detection performance (%) of different methods.

Methods	Accuracy	Precision	Recall	F1 Score
GAN	91.34	89.72	88.43	89.08
AE	95.83	92.08	91.35	91.71
FedAGRU	94.92	97.41	93.32	95.31
FL-MAAE	97.22	96.54	95.73	96.13

To demonstrate the robustness of FL-MAAE , we contaminate it by adding samples with attack labels to the training dataset. Figure 3 illustrates the results of models trained on our method and comparison schemes using datasets with different contamination rates. From the figure, it can be seen that as the contamination rate increases, various intrusion detection performance indicators gradually decline. Moreover, compared with other schemes, our proposed scheme shows the slowest decline in intrusion detection performance and the lowest influence of contaminated samples as the contamination rate increases. This is because FL-MAAE can effectively suppress the interference of contaminated samples on the global model by calculating the contribution of each participant in federated learning. At the same time, the memory module in the memory-augmented autoencoder model enhances the overall anti-interference ability of the model. The experimental results accordingly show that our proposed FL-MAAE exhibits better robustness than the comparison methods when the training dataset is contaminated.

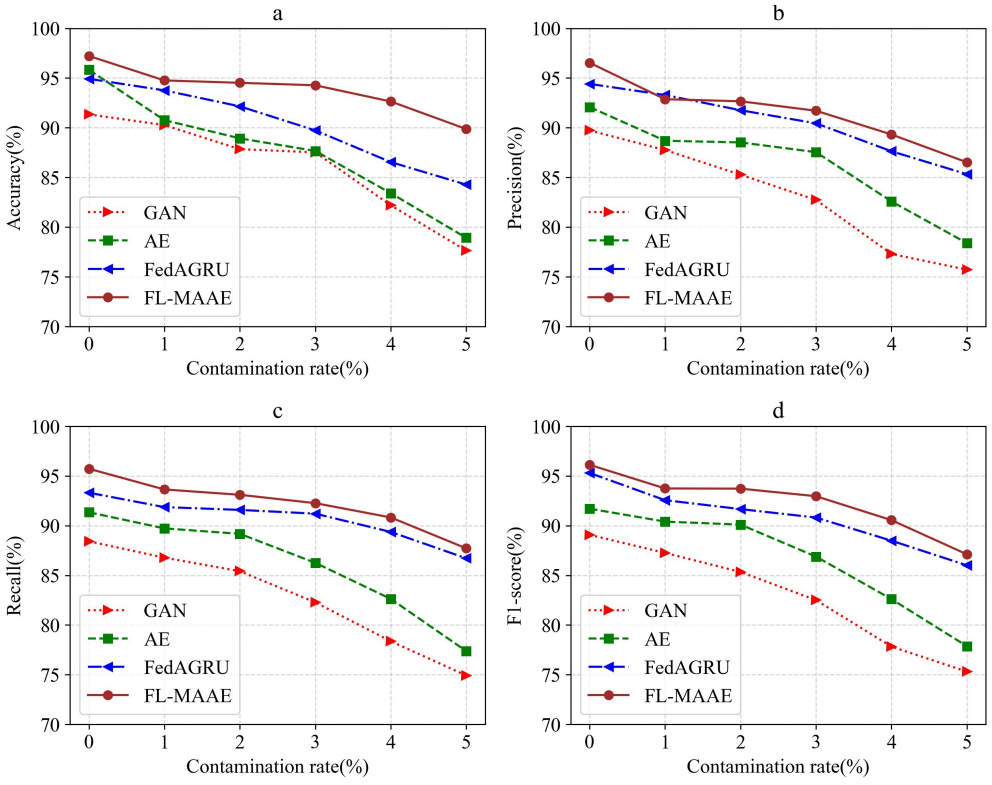

Figure 3. Performance comparison of each method at different contamination rates: (**a**) accuracy comparison, (**b**) precision comparison, (**c**) recall comparison, and (**d**) F1 score comparison.

5.5. Further Analysis

We evaluate the learning rate of the proposed model under federated learning. Figure 4 plots the intrusion detection performance of FL-MAAE with different numbers of communication rounds under the federated learning framework. As the communication rounds increase from one to five, the performance indicators of the model increase significantly; however, after the sixth round of communication, the performance of the model no longer

changes significantly. From this, it can be seen that FL-MAAE has a faster learning speed and requires fewer communication rounds to achieve model convergence.

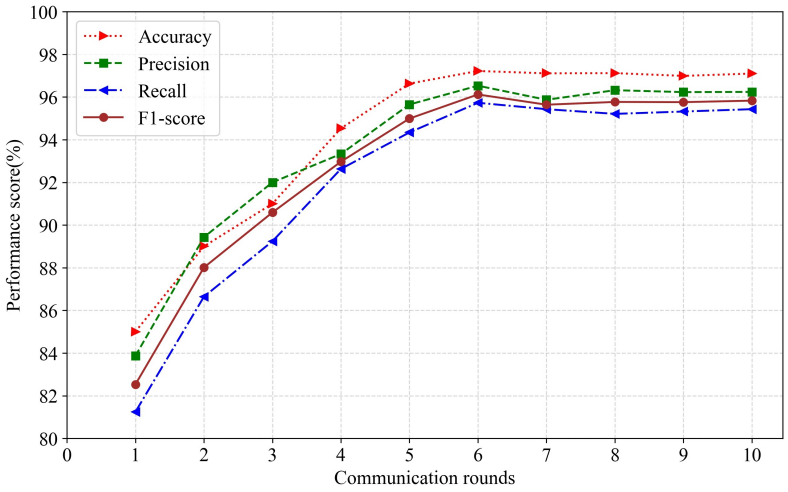

Figure 4. Influence of the number of communication rounds on the performance of the proposed model under federated learning.

We analyzed the influence of the number of participants on the performance of the model under federated learning in the FL-MAAE method. As shown in Figure 5, the increase in the number of participants will have a certain adverse effect on the model performance, when the number of federation participants is less than 30, the performance indicators of the FL-MAAE method do not change much, but as the number further increases, the detection performance will decrease. This is because as the number of participants increases, more uncertainty will be brought to model aggregation. At the same time, the complexity of model aggregation will be higher, making model aggregation more difficult.

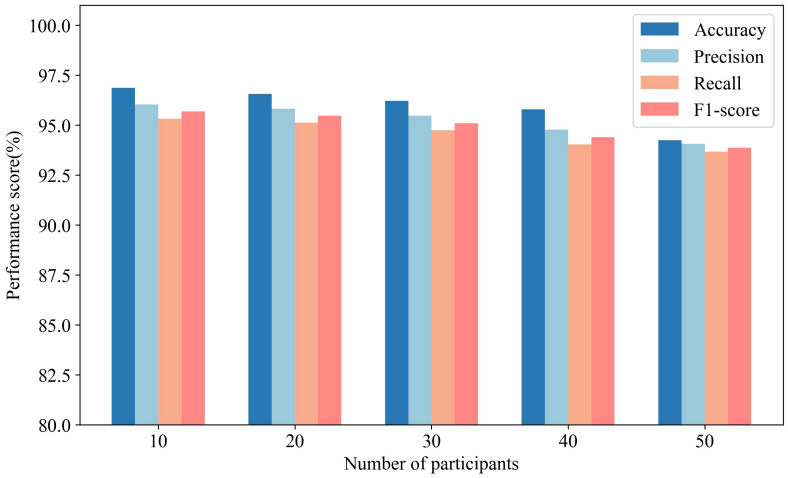

Figure 5. Influence of the number of participants on the performance of the model.

In addition to the above experiments, we also analyzed the influence of the number of memory items in the memory module in FL-MAAE on the model performance. As shown in Figure 6, when the number of memory items is five, intrusion detection performance is reduced, as an insufficient number of memory items precludes the full characterization of the latent features of normal sample data. The detection performance of the model increases with the number of memory items. The best intrusion detection performance is achieved when the number of memory items is 15. When the number of memory items is between 15 and 25, the detection performance tends to be stable. When the number

of items is further increased, the intrusion detection performance degrades, as too many memory terms lead to overfitting of the model.

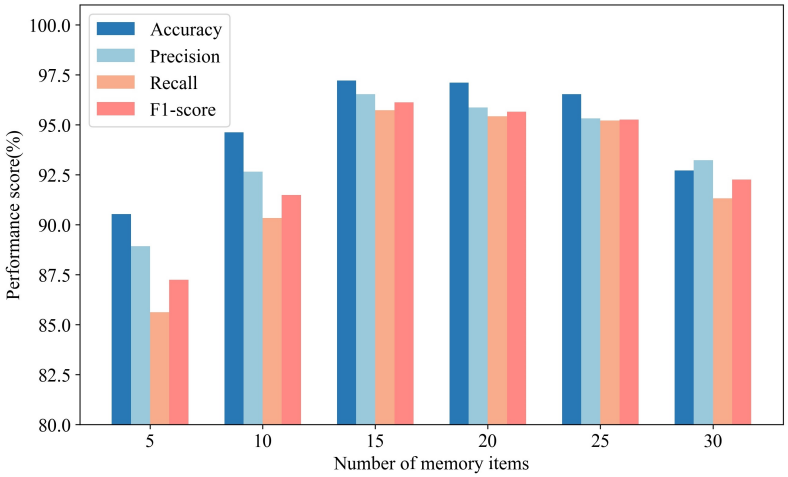

Figure 6. Influence of the number of memory items in the memory module on the performance of intrusion detection.

Since the latent space feature dimension will affect the feature extraction and reconstruction process, thereby affecting the model performance, we analyzed it. As shown in Figure 7, when the feature dimension of the latent space is too small or too large, the intrusion detection performance is low, and the best performance can only be achieved when the dimension is 16. We believe that when the dimension is small, the latent space features cannot effectively represent the behavior data, and when the dimension is large, more redundant features will be generated, thereby reducing the performance of intrusion detection.

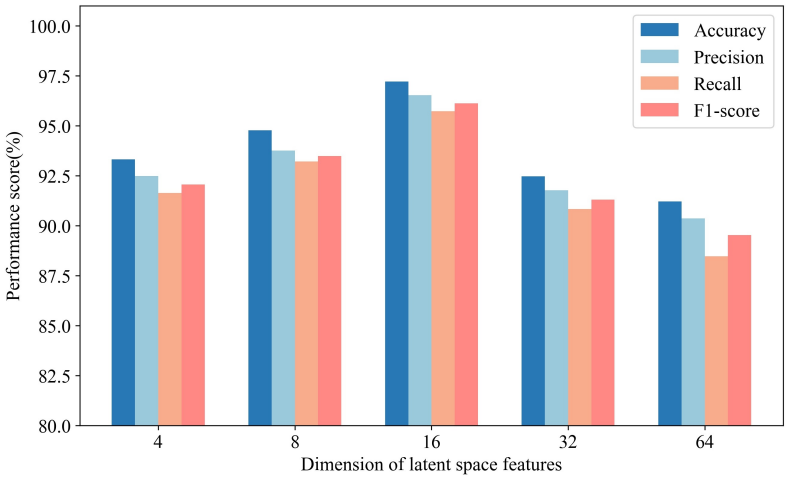

Figure 7. Influence of the latent space feature dimension on the performance of the model.

In summary, the FL-MAAE method in this paper not only shows a certain improvement in intrusion detection performance, but also has good robustness when facing the situation of the training dataset being contaminated. The FL-MAAE method can converge with fewer training rounds, which proves the feasibility and effectiveness of this method in the IoV environment.

6. Conclusions and Future Work

Network security is a key challenge facing the Internet of Vehicles today. Intrusion detection, as an important technology for defending against network attacks and protecting data security, is imperative to apply in the Internet of Vehicles.

In this paper, we propose an intrusion detection method for the Internet of Vehicles based on federated learning and a memory-augmented autoencoder. We added a memory module to the traditional autoencoder model to store the latent features of the normal behavior of the Internet of Vehicles, based on the reconstruction loss as an intrusion judgment indicator, so that various network attacks against the Internet of Vehicles can be effectively detected. In addition, our method performs federated learning training between the roadside unit and the cloud, by locally training the intrusion detection model on the roadside unit, and then uploading the model parameters to the cloud server for aggregation based on performance contribution. This method does not need to upload the Internet of Vehicles data to the cloud, avoiding the leakage of user privacy. Experimental results show that our method has higher accuracy, recall, and F1 score than the existing state-of-the-art methods, and has stronger robustness and faster training speed.

We use a memory-augmented autoencoder to model the communication behavior characteristics of the Internet of Vehicles. However, since the traffic of the Internet of Vehicles is a time series, the memory-augmented autoencoder ignores the time dependence, which may have a certain impact on the detection performance. In future research work, we will study this problem, and further extract temporal features to mine deeper information, so as to improve intrusion detection performance. At the same time, with the increasing data dimension and scale in the Internet of Vehicles, the intrusion detection model is becoming more and more complex. How to reduce the training cost of the federated learning intrusion detection model and make it more suitable for the environment with limited communication resources such as the Internet of Vehicles is also worth further research.

Author Contributions: Conceptualization, L.X. and K.W.; methodology, L.X.; software, K.W. and X.Z.; validation, H.W., H.M. and X.Z.; formal analysis, L.X.; investigation, H.W.; resources, L.X.; writing—original draft preparation, K.W.; writing—review and editing, all authors; visualization, H.W.; supervision, H.M.; funding acquisition, L.X., H.W. and H.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work is fully supported by the National Natural Science Foundation of China (62071170, 62171180, 62072158, 62272146), the Program for Innovative Research Team at the University of Henan Province (21IRTSTHN015), in part by the Key Science and the Research Program of the University of Henan Province (21A510001), Henan Province Science Fund for Distinguished Young Scholars (222300420006), the Science and Technology Research Project of Henan Province under Grant (222102210001), and Leading Talent in Scientific and Technological Innovation in Zhongyuan (234200510018).

Data Availability Statement: The data used to support the findings of this study can be downloaded from https://www.unb.ca/cic/datasets/nsl.html, accessed on 3 May 2022.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoV	Internet of Vehicles
FL-MAAE	Federated learning and memory-augmented autoencoder
V2X	Vehicle to Everything
CNN	Convolutional neural network
RNN	Recurrent neural network
GAN	Generative adversarial network
AE	Autoencoder
OBU	On-board unit
RSU	Roadside unit
DSRC	Dedicated short-range communication
BSM	Basic safety message

References

1. Contreras-Castillo, J.; Zeadally, S.; Guerrero-Ibanez, J.A. Internet of Vehicles: Architecture, Protocols, and Security. *IEEE Internet Things J.* **2018**, *5*, 3701–3709. [CrossRef]
2. Jia, X.; Xing, L.; Gao, J.; Wu, H. A Survey of Location Privacy Preservation in Social Internet of Vehicles. *IEEE Access* **2020**, *8*, 201966–201984. [CrossRef]
3. Aslahi-Shahri, B.M.; Rahmani, R.; Chizari, M.; Maralani, A.; Eslami, M.R.; Golkar, M.J.; Ebrahimi, A. A hybrid method consisting of GA and SVM for intrusion detection system. *Neural Comput. Appl.* **2016**, *27*, 1669–1676. [CrossRef]
4. Yang, L.; Moubayed, A.; Hamieh, I.; Shami, A. Tree-Based Intelligent Intrusion Detection System in Internet of Vehicles. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM 2019), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6. [CrossRef]
5. Zeng, Y.; Qiu, M.; Zhu, D.; Xue, Z.; Xiong, J.; Liu, M. DeepVCM: A Deep Learning Based Intrusion Detection Method in VANET. In Proceedings of the 5th IEEE International Conference on Big Data Security on Cloud, IEEE International Conference on High Performance and Smart Computing, and IEEE International Conference on Intelligent Data and Security, BigDataSecurity/HPSC/IDS 2019, Washington, DC, USA, 27–29 May 2019; pp. 288–293. [CrossRef]
6. Alladi, T.; Gera, B.; Agrawal, A.; Chamola, V.; Yu, F.R. DeepADV: A Deep Neural Network Framework for Anomaly Detection in VANETs. *IEEE Trans. Veh. Technol.* **2021**, *70*, 12013–12023. [CrossRef]
7. Yang, L.; Moubayed, A.; Shami, A. MTH-IDS: A Multitiered Hybrid Intrusion Detection System for Internet of Vehicles. *IEEE Internet Things J.* **2022**, *9*, 616–632. [CrossRef]
8. Zekry, A.; Sayed, A.; Moussa, M.; Elhabiby, M. Anomaly Detection using IoT Sensor-Assisted ConvLSTM Models for Connected Vehicles. In Proceedings of the 93rd IEEE Vehicular Technology Conference (VTC Spring 2021), Helsinki, Finland, 25–28 April 2021; pp. 1–6. [CrossRef]
9. Xing, L.; Zhao, P.; Gao, J.; Wu, H.; Ma, H. A Survey of the Social Internet of Vehicles: Secure Data Issues, Solutions, and Federated Learning. *IEEE Intell. Transp. Syst. Mag.* **2023**, *15*, 70–84. [CrossRef]
10. Zhao, P.; Huang, Y.; Gao, J.; Xing, L.; Wu, H.; Ma, H. Federated Learning-Based Collaborative Authentication Protocol for Shared Data in Social IoV. *IEEE Sens. J.* **2022**, *22*, 7385–7398. [CrossRef]
11. Gao, Y.; Wu, H.; Song, B.; Jin, Y.; Luo, X.; Zeng, X. A Distributed Network Intrusion Detection System for Distributed Denial of Service Attacks in Vehicular Ad Hoc Network. *IEEE Access* **2019**, *7*, 154560–154571. [CrossRef]
12. Zaidi, K.; Milojevic, M.B.; Rakocevic, V.; Nallanathan, A.; Rajarajan, M. Host-Based Intrusion Detection for VANETs: A Statistical Approach to Rogue Node Detection. *IEEE Trans. Veh. Technol.* **2016**, *65*, 6703–6714. [CrossRef]
13. Hoang, T.; Kim, D. Detecting in-vehicle intrusion via semi-supervised learning-based convolutional adversarial autoencoders. *Veh. Commun.* **2022**, *38*, 100520. [CrossRef]
14. Abdalzaher, M.S.; Elwekeil, M.; Wang, T.; Zhang, S. A Deep Autoencoder Trust Model for Mitigating Jamming Attack in IoT Assisted by Cognitive Radio. *IEEE Syst. J.* **2022**, *16*, 3635–3645. [CrossRef]
15. Abdalzaher, M.S.; Muta, O. A Game-Theoretic Approach for Enhancing Security and Data Trustworthiness in IoT Applications. *IEEE Internet Things J.* **2020**, *7*, 11250–11261. [CrossRef]
16. Sedjelmaci, H.; Senouci, S. An accurate and efficient collaborative intrusion detection framework to secure vehicular networks. *Comput. Electr. Eng.* **2015**, *43*, 33–47. [CrossRef]
17. Li, W.; Joshi, A.; Finin, T. SVM-CASE: An SVM-Based Context Aware Security Framework for Vehicular Ad-Hoc Networks. In Proceedings of the IEEE 82nd Vehicular Technology Conference (VTC Fall 2015), Boston, MA, USA, 6–9 September 2015; pp. 1–5. [CrossRef]
18. Levi, M.; Allouche, Y.; Kontorovich, A. Advanced Analytics for Connected Car Cybersecurity. In Proceedings of the 87th IEEE Vehicular Technology Conference (VTC Spring 2018), Porto, Portugal, 3–6 June 2018; pp. 1–7. [CrossRef]
19. Hu, Z.; Wang, L.; Qi, L.; Li, Y.; Yang, W. A Novel Wireless Network Intrusion Detection Method Based on Adaptive Synthetic Sampling and an Improved Convolutional Neural Network. *IEEE Access* **2020**, *8*, 195741–195751. [CrossRef]
20. Park, D.; Kim, S.; Kwon, H.; Shin, D.; Shin, D. Host-Based Intrusion Detection Model Using Siamese Network. *IEEE Access* **2021**, *9*, 76614–76623. [CrossRef]
21. Zhou, H.; Kang, L.; Pan, H.; Wei, G.; Feng, Y. An intrusion detection approach based on incremental long short-term memory. *Int. J. Inf. Secur.* **2022**, *22*, 433–446. [CrossRef]
22. Ashraf, J.; Bakhshi, A.D.; Moustafa, N.; Khurshid, H.; Javed, A.; Beheshti, A. Novel Deep Learning-Enabled LSTM Autoencoder Architecture for Discovering Anomalous Events From Intelligent Transportation Systems. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 4507–4518. [CrossRef]
23. Liu, H.; Zhang, S.; Zhang, P.; Zhou, X.; Shao, X.; Pu, G.; Zhang, Y. Blockchain and Federated Learning for Collaborative Intrusion Detection in Vehicular Edge Computing. *IEEE Trans. Veh. Technol.* **2021**, *70*, 6073–6084. [CrossRef]
24. Chen, Z.; Lv, N.; Liu, P.; Fang, Y.; Chen, K.; Pan, W. Intrusion Detection for Wireless Edge Networks Based on Federated Learning. *IEEE Access* **2020**, *8*, 217463–217472. [CrossRef]
25. Attota, D.C.; Mothukuri, V.; Parizi, R.M.; Pouriyeh, S. An Ensemble Multi-View Federated Learning Intrusion Detection for IoT. *IEEE Access* **2021**, *9*, 117734–117745. [CrossRef]
26. Lv, Z.; Qiao, L.; Li, J.; Song, H. Deep-Learning-Enabled Security Issues in the Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 9531–9538. [CrossRef]

27. Lv, Z.; Xiu, W. Interaction of Edge-Cloud Computing Based on SDN and NFV for Next Generation IoT. *IEEE Internet Things J.* **2020**, *7*, 5706–5712. [CrossRef]
28. Lv, Z.; Lou, R.; Li, J.; Singh, A.K.; Song, H. Big Data Analytics for 6G-Enabled Massive Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 5350–5359. [CrossRef]
29. Zhang, J.; Peng, S.; Gao, Y.; Zhang, Z.; Hong, Q. APMSA: Adversarial Perturbation Against Model Stealing Attacks. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1667–1679. [CrossRef]
30. Mills, J.; Hu, J.; Min, G. Communication-Efficient Federated Learning for Wireless Edge Intelligence in IoT. *IEEE Internet Things J.* **2020**, *7*, 5986–5994. [CrossRef]
31. Cao, X.; Zhu, G.; Xu, J.; Cui, S. Transmission Power Control for Over-the-Air Federated Averaging at Network Edge. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 1571–1586. [CrossRef]
32. Zhang, X.; Liu, Y.; Liu, J.; Argyriou, A.; Han, Y. D2D-Assisted Federated Learning in Mobile Edge Computing Networks. In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC 2021), Nanjing, China, 29 March–1 April 2021; pp. 1–7. [CrossRef]
33. He, C.; Li, S.; So, J.; Zeng, X.; Zhang, M.; Wang, H.; Wang, X.; Vepakomma, P.; Singh, A.; Qiu, H.; et al. FedML: A Research Library and Benchmark for Federated Machine Learning. *arXiv* **2020**, arXiv:2007.13518. [CrossRef]
34. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019 (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; pp. 8024–8035.
35. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA 2009), Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
36. Ferdowsi, A.; Saad, W. Generative Adversarial Networks for Distributed Intrusion Detection in the Internet of Things. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM 2019), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6. [CrossRef]
37. Faber, K.; Faber, L.; Sniezynski, B. Autoencoder-based IDS for cloud and mobile devices. In Proceedings of the 21st IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGrid 2021), Melbourne, Australia, 10–13 May 2021; pp. 728–736. [CrossRef]