applied sciences

MDPI

Article

ResADM: A Transfer-Learning-Based Attack Detection Method for Cyber–Physical Systems

Huan Wang [1,2,3], Haifeng Zhang [1,2,3], Lei Zhu [2,3,4,*], Yan Wang [1,2,3] and Junyi Deng [1,2,3]

[1] School of Computer Science and Technology, Guangxi University of Science and Technology, Liuzhou 545006, China; wanghuan@gxust.edu.cn (H.W.); 221068704@stdmail.gxust.edu.cn (H.Z.); 221068701@stdmail.gxust.edu.cn (Y.W.); dengjunyi@gxust.edu.cn (J.D.)
[2] Liuzhou Key Laboratory of Big Data Intelligent Processing and Security, Liuzhou 545006, China
[3] Guangxi Education System Network Security Monitoring Center, Liuzhou 545006, China
[4] School of Information Technology, Guangxi Police College, Nanning 530028, China
* Correspondence: zhulei@gxjcxy.edu.cn

Abstract: Deep learning has proven to be effective for enhancing the accuracy and efficiency of attack detection through training with large sample sizes. However, when applied to cyber–physical systems (CPSs), it still encounters challenges such as scarcity of attack samples, the difficulty of selecting features for high-dimensional data, and weak model-generalization ability. In response, this paper proposes ResADM, a transfer-learning-based attack detection method for CPSs. Firstly, an intentional sampling method was employed to construct different sets of samples for each class, effectively balancing the distribution of CPS-attack samples. Secondly, a feature-selection method based on importance was designed to extract the meaningful features from attack behaviors. Finally, a transfer-learning network structure based on ResNet was constructed, and the training parameters of the source model were optimized to form the network-attack detection method. The experimental results demonstrated that ResADM effectively balanced the data classes and extracted 32-dimensional attack-behavior features. After pre-training on the UNSW-NB15 dataset, ResADM achieved a detection accuracy of up to 99.95% for attack behavior on the CICIDS2017 dataset, showcasing its strong practicality and feasibility.

Keywords: attack identification; cyber–physical systems (CPS); transfer learning; residual network (ResNet)

Citation: Wang, H.; Zhang, H.; Zhu, L.; Wang, Y.; Deng, J. ResADM: A Transfer-Learning-Based Attack Detection Method for Cyber–Physical Systems. *Appl. Sci.* **2023**, *13*, 13019. https://doi.org/10.3390/app132413019

Academic Editor: Mohamed Benbouzid

Received: 3 November 2023
Revised: 3 December 2023
Accepted: 4 December 2023
Published: 6 December 2023

1. Introduction

Cyber-Physical Systems (CPSs) are a complex, tight integration of computer networks and physical systems that interact with each other. They are characterized by their large scale and high complexity. With the rapid development of computer network technology and the expanding reach of the Internet, an increasing number of enterprise and government services rely on the Internet to provide convenience in people's lives and work. However, the openness and interconnectedness of CPS applications have also resulted in more prominent network security issues [1]. Network attack behaviors, particularly advanced-persistent-threat (APT) attacks, exhibit characteristics including high destructiveness, extreme stealthiness, and long incubation periods. These attacks can carry out targeted attacks and covert theft of critical assets, especially CPSs, making CPSs significant factors affecting the security of government and enterprise networks [2]. Therefore, the network security concern is pressing.

For the timely detection of malicious activities and attack behaviors in CPSs, as well as to prevent attackers from stealing sensitive data and reducing the risk of CPS service interruption, accurate attack detection methods are particularly important. Traditional host-based methods have required the constant monitoring of the operation and the state of the host system in order to detect system events, but they have been unable to analyze

behavior information related to the CPS. On the other hand, network-based methods have faced difficulties in obtaining the internal state information of the monitored system when dealing with big-data scenarios, leading to poor detection performance [3].

With the continuous development of machine-learning and deep-learning technologies, intrusion detection methods based on anomalies have flourished, with the assistance of artificial intelligence models. Yang et al. [4] proposed multiple efficient machine-learning algorithms that realized network anomaly behavior detection models in lightweight IoT environments and achieved excellent results in simulation experiments on multiple datasets. Sinha et al. [5] proposed a convolutional neural network (CNN) fused with a bidirectional, long short-term memory (BiLSTM) network model that effectively mined the hidden features of network attack behaviors and improved the accuracy of network-attack detection.

However, the application of machine-learning methods in CPS attack detection have encountered several challenges: (1) Scarcity of attack samples: Machine-learning methods rely heavily on the availability of training data. However, CPS attack behavior data have often been scarce, making it challenging to collect a comprehensive dataset [6]. (2) Difficulty in selecting high-dimensional data features: CPS attack behavior samples encompass numerous data features related to network interactions. Handling high-dimensional data has not only increased storage and computational costs but has also posed challenges to uncovering hidden features of attack behaviors [7]. (3) Weak model generalization: Trained machine-learning models have often struggled to adapt to CPS attack behaviors in new application scenarios, leading to high false-positive rates and reduced effectiveness [8].

The proposed CPS attack detection method, ResADM (ResNet-Based Attack Detection Method), addressed these challenges and provided the following contributions:

- A data-cleaning method was proposed that utilized intentional sampling to clean the CPS traffic data. Subsequently, a sampling technique was employed to address the issue of data category imbalance by extracting a comparable number of training samples from each category.
- A feature-selection method was developed based on feature importance to identify significant CPS attack features and eliminate irrelevant ones. This approach effectively addressed the challenge of feature selection in high-dimensional data, making the process more manageable.
- A transfer-learning network structure was constructed using the residual network (ResNet) model. The optimization of the training parameters for the source model enabled the formation of a network-attack-behavior detection model that improved the model's generalization ability.
- To validate the effectiveness of ResADM, pre-training was conducted on the UNSW-NB15 dataset, followed by simulation experiments on the CICIDS2017 dataset. The results demonstrated that ResADM had achieved a balanced distribution of the data categories, extracted 32-dimensional attack-behavior features, and achieved a high accuracy of 99.95% in detecting CPS attack behaviors.

This paper is organized as follows: Section 2 provides an introduction to the background. Section 3 presents the model design proposed in this paper. Section 4 describes the conducted experiments and provides a comprehensive analysis of the experimental results. In Section 5, a summary of the entire article is presented, along with highlighting the future research directions.

2. Background

2.1. Attack Detection

Network attack detection is a crucial task in maintaining CPS security, as it aims to identify and prevent unauthorized network activities. Traditional statistical, learning-based methods have had certain limitations in this regard, including vulnerability to detection by attackers and sensitivity to normality assumptions. As machine-learning technology continues to advance, these conventional methods are gradually being replaced. In recent years, extensive research has focused on utilizing machine-learning models,

such as Bayesian, support vector machine, and decision tree, for network-attack detection. However, these methods rely heavily on expert knowledge and struggle to uncover the underlying and concealed features of network attack behaviors. Furthermore, with the increasing number of devices and the expanding network data in network scenarios, this problem becomes even more pronounced, necessitating the use of more self-learning and adaptive methods to address this challenge.

Deep-learning technology, with its ability to construct nonlinear network structures with multiple hidden layers, offers the advantage of deeply learning sample data and adapting to high-dimensional learning and prediction requirements. It also saves a significant amount of time during feature extraction. As a result, deep learning has achieved remarkable success in various fields, including image recognition, natural language processing, and speech recognition. Within the context of CPS attack detection, some studies [9] have utilized a multi-scale CNN structure to explore the spatial features of network attacks while employing LSTM to extract temporal features, resulting in improved accuracy of attack detection. Furthermore, previous research has proposed the use of BiLSTM networks to analyze bidirectional, temporal features of data streams, enabling the discovery of hidden temporal features of network attacks and significantly reducing false alarm rates [10]. However, it is important to acknowledge the challenges associated with deep-learning methods, such as gradient-vanishing and -exploding, which has led to unstable weight updates in the model and resulted in under-fitting or over-fitting situations.

2.2. Residual Network

The residual network, also known as ResNet, is a classic convolutional neural network that utilizes residual blocks to implement skip connections in stacked CNN networks. This approach mitigated the problem of feature loss and addressed the issues of vanishing and exploding gradients by directly passing the previous layer's features on to the subsequent layers, facilitating easier training and optimization [11]. Residual networks have found wide applications in tasks such as image recognition and text classification. The structure of ResNet is illustrated in Figure 1, where x represents the input; the weight layer refers to the convolutional layers responsible for feature extraction; $F(x)$ denotes the result obtained after convolution; x identity represents the identity mapping, where the input data are directly passed through the residual connection without any transformation to ensure the consistency of input dimensions, enabling smooth residual connections; $F(x) + x$ indicates the combination of the output from two convolutional layers with the previous convolutional layer; and ReLU (rectified linear unit) is the activation function.

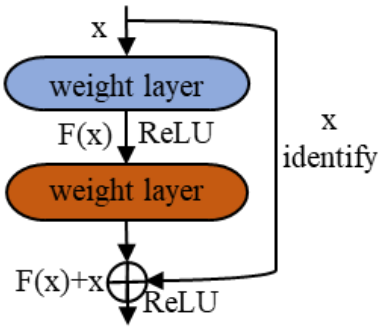


Figure 1. Structure of the residual block.

Some studies have utilized ResNet as a foundation and enhanced the richness and the accuracy of the feature extraction by incorporating cross-layer transfer and embedding modules [12]. This approach has successfully addressed the challenges associated with detecting train bottom parts. Furthermore, there has been research that combined auto-encoder networks with residual networks to create a novel network intrusion-detection method [13]. In this method, pre-processed data were used to extract features through an auto-encoder network by reconstructing the reconstruction error. The extracted fea-

tures were then employed to train a specifically designed residual network, which was subsequently used for classification detection on a test set using the trained model. The experimental results demonstrated that this approach achieved good performance in terms of accuracy, a true-positive rate, and a false-positive rate.

However, network intrusion-detection work has often faced a scarcity in the training data for business systems and intelligent security devices. This limitation has presented challenges to deep-learning-based intrusion-detection models when applied to new cyber–physical-system applications, as their generalization capability has been weak. The distribution of network traffic samples in a new CPS application scenario might differ from the training dataset, leading to reduced effectiveness. Therefore, enhancing the generalization capability of network-attack-detection models is a critical issue that necessitates further research and improvements in model design. By addressing this challenge, these models can be more effective and reliable in new CPS scenarios.

2.3. Transfer Learning

Transfer learning is a machine-learning method that leverages existing knowledge from source tasks to enhance the learning performance in targeted tasks. As depicted in Figure 2, this approach utilizes labeled data, pre-trained models, and domain adaptation techniques to overcome challenges related to limited labeled data and an abundance of unlabeled data in the targeted domain. By doing so, transfer learning has reduced the computational complexity in big-data environments and improved the generalization ability of conventional models under specific conditions. Consequently, transfer learning has emerged as a promising framework for addressing the weak generalization capabilities of machine-learning models [14].

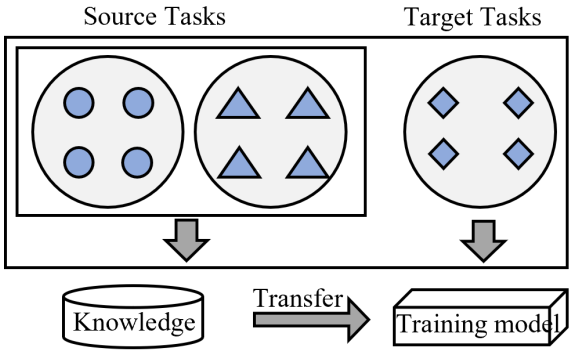


Figure 2. Schematic diagram of transfer learning.

Yilmaz et al. [15] proposed the use of transfer learning to tackle this issue in IoT network-intrusion-detection tasks. They employed pre-trained models to extract knowledge from a source domain and adapt it to the characteristics of the target domain. In a similar vein, Mehedi et al. [16] developed a deep residual convolutional neural network model for network-intrusion detection in industrial systems. Through the utilization of transfer learning, they successfully achieved the detection of unknown attacks. Their work represented a valuable exploration and the practical application of transfer learning within the context of network-intrusion detection in industrial control systems. Hence, transfer learning, as an effective machine-learning method, holds significant importance and demonstrates great potential in addressing the challenges associated with limited datasets and weak generalization ability in CPS intrusion detection.

3. ResADM Architecture

The overall architecture of the proposed ResADM is illustrated in Figure 3. The method comprised three main modules: the backbone network pre-training module, the transfer-learning module, and the backbone network re-training module.

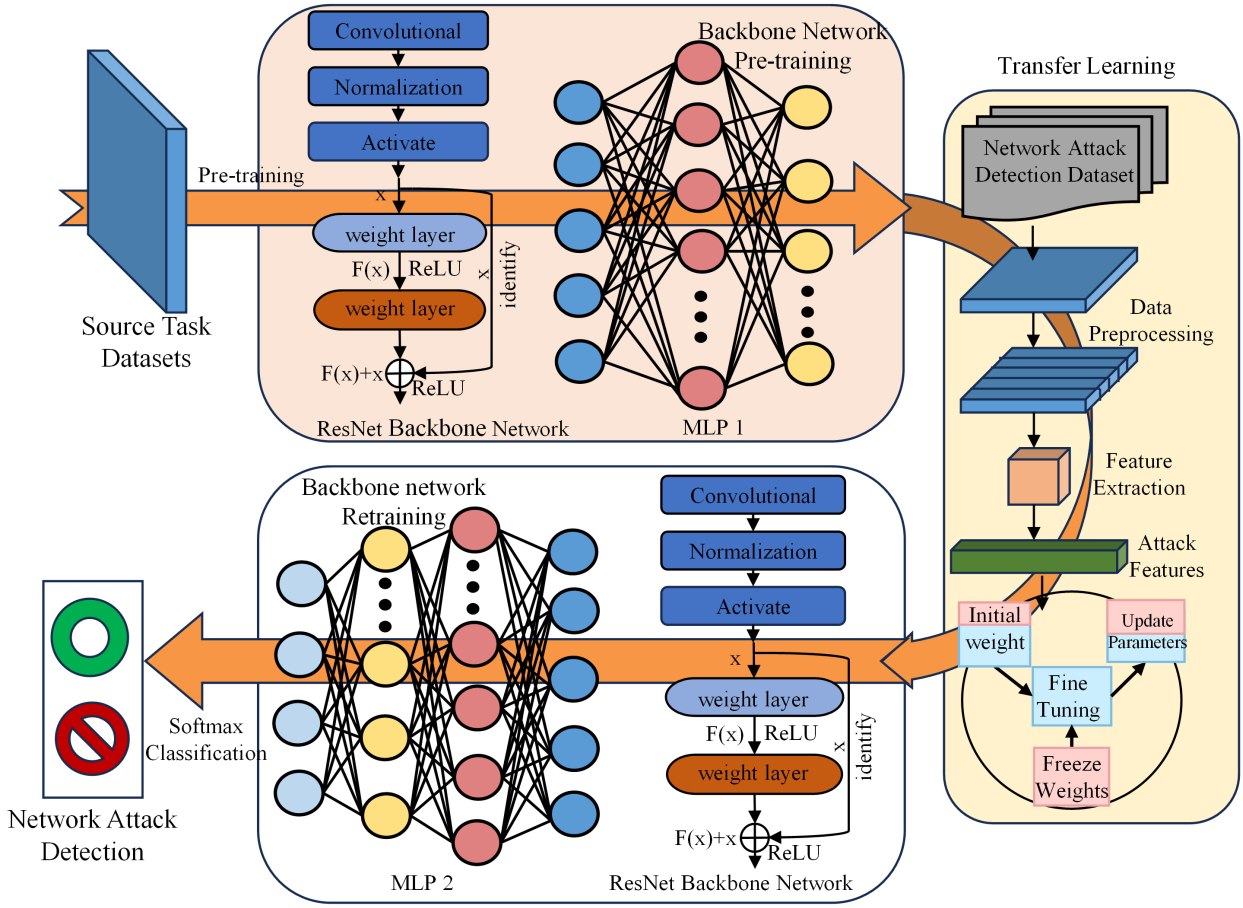


Figure 3. Architecture of ResADM.

3.1. Pre-Training Module

In the backbone network pre-training module, ResNet-50 was employed as the backbone network. The pre-training process involved training the backbone network using the source task dataset through supervised learning, resulting in a trained backbone network with all parameters. The residual block structure formed the foundation of ResNet. The convolutional process typically involved feature dimension reduction. The inclusion of a Cov1×1 operation in the shortcut connection aimed to ensure consistency in the dimensions of the input and output features. In the residual network layer, represented as $y = H(x)$, the residual block could be expressed as $H(x) = F(x) + x$. In the case of identity mapping, $y = x$ represents the observed value, $H(x)$ represents the predicted value, and thus $F(x)$ corresponds to the residual. Based on these considerations, the process of calculating the residual could be derived, where W_i is the weight matrix and W_s is the linear projection matrix, as shown in Equation(1).

$$H(X) = F(x, W_i) + W_S x, \tag{1}$$

3.2. Transfer-Learning Module

The pre-trained model and its parameters were passed into the transfer-learning module for the pre-processing of the target task dataset. The pre-processing process consisted of two main parts: the data-cleaning layer and the feature-selection layer.

3.2.1. The Data-Cleaning Layer

The collected dataset of online CPS behavior often contained a significant amount of noise, missing values, and outliers. These issues not only hindered the analysis of network traffic but also could have a detrimental effect on the accuracy and reliability of detection

models. Consequently, a purposive-sampling-based data-cleaning method was devised, and the specific steps included:

1. Illegal character replacement: Replace unrecognizable illegal characters in the dataset. Replace the "NaN" (Not a Number) and "Inf" (Infinity) characters in the dataset with the mean value of their respective columns.
2. Invalid column removal: Analyze the elements in the dataset columns and remove columns that consist entirely of "0", empty values, or duplicate items, in order to reduce data dimensionality.
3. Purposive sampling: Randomly select an equal number of samples from all the data, but with different categories, for model training and testing. This helped to avoid the problem of class imbalance caused by a large amount of benign traffic and a small amount of attack traffic, which could affect the model training.
4. Data standardization: Apply the min–max normalization method to the sampled data, which maps numerical features of the data samples to the range [0, 1]. Use one-hot encoding to encode categorical features in the samples.
5. Dataset splitting: The processed data samples are divided into a training set and a testing set, according to the requirements of the model training process.

3.2.2. The Feature Selection Layer

Effective feature selection could significantly reduce data dimensionality and enable the model to discover underlying patterns and structures in the data more efficiently, mitigating the impact of data complexity and noise. In this study, a feature-selection method based on feature importance was designed, leveraging the efficient graph-computing algorithm in the light gradient-boosting machine (LightGBM). The LightGBM employs a gradient-based optimization sampling algorithm to efficiently reduce the number of data instances and features for each feature. This process helped to narrow down the search space for split points [17]. The algorithm sorted all the data in descending order based on the absolute value of the gradient. Subsequently, a subset P was extracted by selecting the data with the maximum gradient, while a random subset Q was randomly chosen from the remaining data. The gain was then calculated from the subset $(P \cup Q)$, and the variance gain $G_f(i)$ for feature f at split point i was defined according to Equations (2)–(4), where S represents the number of instances in the subset $(P \cup Q)$ and $S_l(i)$ and $S_h(i)$ represent the number of instances with feature f values less than or greater than i, respectively. Additionally, g_k represents the negative gradient of instance X_k, where $P_l = \{X_k \in P : X_{kj} \leq v\}$, $Q_l = \{X_k \in Q : X_{kj} \leq v\}$, $P_h = \{X_k \in P : X_{kj} \leq v\}$, and $Q_h = \{X_k \in Q : X_{kj} \leq v\}$.

$$G_f(i) = \frac{1}{S}(G_{f1}(i) + G_{f2}(i)), \tag{2}$$

$$G_{f1}(i) = \frac{1}{S_l(i)}(\sum_{X_k \in P_l} g_k + \frac{1-a}{b} \sum_{X_k \in Q_l} g_k)^2, \tag{3}$$

$$G_{f2}(i) = \frac{1}{S_h(i)}(\sum_{X_k \in P_h} g_k + \frac{1-a}{b} \sum_{X_k \in Q_h} g_k)^2, \tag{4}$$

The decision model in LightGBM was constructed based on the aforementioned steps for feature set $F_i = \{f_1, f_2, ..., f_m\}$, where $i = \{1, 2, 3, ..., m\}$. The feature importance score FIS_i was calculated by considering the number of times each feature had been used to split the training data across all decision models, as shown in Equation (5). In this formula, w_i represents the weight assigned to each feature, and f_i represents the feature set.

$$FIS_i = \{s|s = w_i f_i\}, \tag{5}$$

Based on the above approach, the feature importance of the attributes for different categories of CPS behaviors was calculated during the classification process. All attributes

were then sorted in descending order based on their importance. From the sorted attribute set, the top k attributes were selected, one by one, to create the feature value set. This feature value set was used as input for retraining the baseline LightGBM model. The value of k was gradually increased until the maximum classification accuracy was achieved after retraining. The top k attributes at this point were selected as the feature set for model training.

3.3. Retraining Module

To address the differences in the feature distribution and learning between the pre-trained ResNet-50 backbone network used on the UNSW-NB15 dataset and the CICIDS2017 dataset, this study proposed the integration of a multilayer perceptron (MLP) layer after the backbone network. Unlike conventional deep neural networks, the MLP architecture in ResADM comprised 2 dense layers with 128 and 64 neurons, respectively, in addition to the input and output layers. Furthermore, a regularization layer with a dropout rate of 0.5 was incorporated into the MLP. By retaining certain pre-trained ResNet weights, the model effectively learned the features specific to the target dataset, thus enhancing the generalization capability of the attack-detection model.

To classify CPS attack behaviors, a softmax function was employed. The CPS traffic data were passed through the feature learning layer of ResNet-50 to extract the deep features from the input CPS traffic data. The learned attack-behavior features were then fed into the softmax layer for classification, resulting in recognition results for the CPS traffic samples. Specifically, if there were m categories of labels L in the CPS traffic sample F, where $L \in \{1, 2, ..., m\}$, the conditional probability of softmax predicting that the sample F_i belonged to category m was calculated using Equation (6), where w_i represents the weight vector of the m-th category of traffic.

$$P(L = m|F_i) = \frac{exp(w_m^T)F_i}{\sum_{i=1}^{m} exp(w_i^T F_i)}, \quad (6)$$

4. Experiment and Analysis

4.1. Experimental Environment

The experimental work described in this paper was performed on a deep-learning workstation running the Ubuntu operating system. The code was implemented using Python, and an RTX 3090 GPU accelerator was utilized for acceleration. TensorFlow was employed as the deep-learning framework. The hardware and software configuration of the workstation used in the experiments is provided in Table 1.

Table 1. Experimental environment configuration.

Category	Configuration	Parameter
Hardware	CPU	AMD 7T83
	RAM	DDR4 ECC 128GB
	GPU	NVIDIA GeForce RTX 3090
	SSD	2T
Software	Operating System	Ubuntu 22.04
	Python	3.9
	LightGBM	3.2.1
	TensorFlow	2.11

4.2. Evaluation Indicators

The performance metrics used to evaluate the machine-learning model have been extensively explained in previous studies [18,19]. Here is a brief introduction to these metrics:

Accuracy measured the overall performance of the model by calculating the ratio of correctly classified samples to the total number of samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}, \quad (7)$$

Precision evaluated the ability of the model to correctly identify negative samples by calculating the ratio of true-negatives to the sum of true-negatives and false-positives.

$$Precision = \frac{TP}{TP + FP}, \quad (8)$$

Recall, also known as sensitivity or the true-positive rate, measured the model's ability to correctly identify positive samples. It was calculated as the ratio of true-positives to the sum of true-positives and false-negatives.

$$Recall = \frac{TP}{TP + FN}, \quad (9)$$

The F1-score was a measure of the model's balance between precision and recall. It was the harmonic mean of precision and recall and provided an overall assessment of the model's performance.

$$F1 - score = \frac{2TP}{2TP + FN + FP}, \quad (10)$$

The formulas for calculating these metrics are shown in Equations (7)–(10), where TP represents true-positives, FP represents false-positives, TN represents true-negatives, and FN represents false-negatives.

4.3. Dataset Selection

The completeness of the training data directly affects the effectiveness of deep-learning models and their performance in real-world applications. Public datasets in the field of intrusion detection, such as KDD CUP 99 and NSLKDD, have had some limitations. These included outdated data; lack of integrity and diversity in attacks; and data sanitization issues. As a result, most attacks in these datasets have lacked meaningful payload information and thus have not effectively reflected attack trends. After conducting multiple experimental attempts, we found in this study that the UNSW-NB15 and CICIDS2017 datasets contained a significant amount of valid attack payloads. These datasets could accurately reflect the trends of network attacks over a period of time, meeting the requirements of this study. The UNSW-NB15 dataset consisted of 257,673 flow records, with 175,341 records used for the training set and 82,332 records used for the testing set. Each flow record contained 49 features, including 5 flow features, 13 basic features, 8 content features, 9 time features, 12 additional features, and 2 label tags. The CICIDS2017 dataset consisted of 8 CSV files, comprising a total of 2,273,097 network flow samples. Each flow sample included 79 features and 1 label.

4.4. Experimental Setup

The experimental dataset was constructed using the CICIDS2017 dataset, which was a standard dataset for network traffic anomaly detection. The training process of ResADM is shown in Algorithm 1..

The CICIDS2017 dataset needed to be cleaned according to the prescribed model design. Firstly, identify and replace any illegal characters found in the "Label" column in the dataset. Next, delete column 55 (Fwd Header Length) that duplicated column 34 (Fwd Header Length). Furthermore, discard rows where the values in columns 31 (Bwd PSH Flag), 33 (Bwd URG Flags), 56 (Fwd Avg Bytes/Bulk), 57 (Fwd Avg Packets/Bulk), 58 (Fwd Avg Bulk Rate), 59 (Bwd Avg Bytes/Bulk), 60 (Bwd Avg Packets/Bulk), and 61 (Bwd Avg Bulk Rate) were all equal to zero. In column 14 (Flow Bytes/s), there were 1358 instances of "NaN" and 1509 instances of "Inf". Similarly, column 15 (Flow Packets/s) contained 2867 instances of "Inf". To rectify these invalid values, calculate the column mean and substitute these values accordingly.

Algorithm 1: ResADM for Attack Detection

Input : ResNet-50, UNSW-NB15 dataset, pre-processed CICIDS2017 dataset
Output: ResADM attack detection results
Pre-train ResNet-50 on the UNSW-NB15 dataset;
Initialize ResADM with the pre-trained ResNet-50;
Compute feature importance using LightGBM on the CICIDS2017 dataset;
for $k = 2$ **to** N **do**
  Get top k importances;
  Retrain ResADM with the top k importances;
  if *ReTrain.Score* > *Score* **then**
    Features[] ← Get top k importances;
  end
end
Data ← getDataWithFeatures(CICIDS2017);
result ← train(ResADM.add(MLP));
result ← softmax(result);
return result;

Once the dataset was cleaned, a purposive sampling of network attack samples could be performed. To begin, randomly select 10,000 benign flows labeled as "BENIGN" from the file "Monday-WorkingHours". Extract 5000 instances of brute-force attacks labeled as "FTP-Patator" and another 5000 instances labeled as "SSH-Patator" from the file "Tuesday-WorkingHours". Merge these samples to form a set of 10,000 attack samples labeled as "Patator". Similarly, sample 10,000 instances of denial-of-service attacks labeled as "DoS" from the file "Wednesday-workingHours". Lastly, extract 10,000 instances of distributed denial-of-service attacks labeled as "DDoS" from the file "Friday-WorkingHours-Afternoon-DDos". These steps result in a dataset containing 40,000 network flow samples, each with 78 features and 1 label. Randomly split the dataset into an 80% training set and a 20% test set. The distribution of each class within the divided dataset can be found in Table 2.

Table 2. Number of samples in each category of the dataset.

	BENIGN	DDoS	DoS	Patator
Training set	7953	8000	8014	8033
Testing set	2047	2000	1986	1967

The training set had to be passed to the attack-feature-selection layer for feature-importance analysis using the LightGBM model. The feature importance was calculated in the multi-classification process of traffic samples, and the features were ordered accordingly. Select the top k features with the highest importance for retraining the baseline multi-classification model. Gradually increase the value of k until it reaches 32, which maximizes the accuracy of the retrained baseline model. Remove the remaining features and keep the selected 32 features, as shown in Figure 4.

To visualize the correlation between these 32 features and their sample labels, apply the K-means clustering algorithm to the network flow samples, as illustrated in Figure 5. The visualization demonstrated a clear distinction between the benign flow samples and the various categories of network attack flow samples, indicating that these 32-dimensional features effectively characterized normal traffic and attack traffic. Although DoS and DDoS attacks shared some behavioral features, there were still significant differences due to the presence of a large number of dispersed source addresses in the DDoS attacks. Therefore, the selected 32-dimensional features could be used as a feature set for training the backbone network.

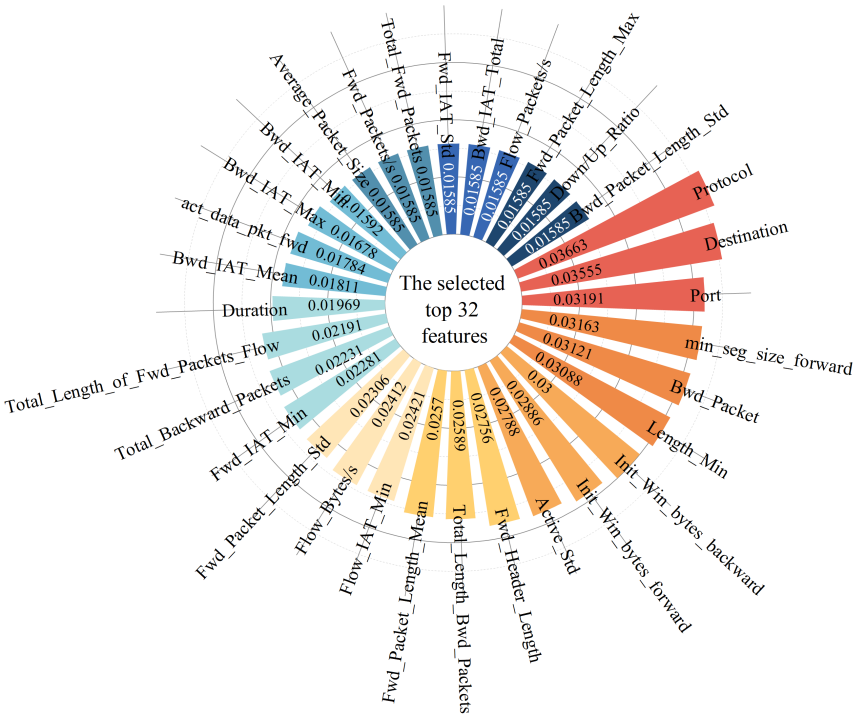


Figure 4. Selected features.

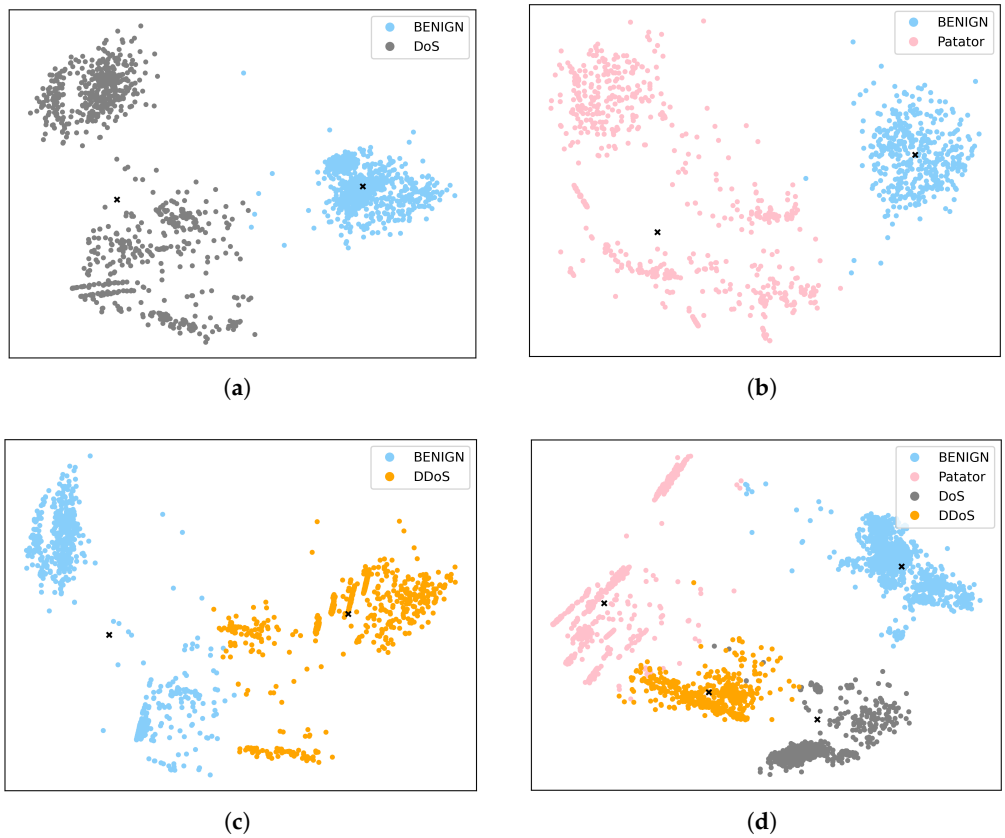


Figure 5. Comparison of CPS attack behavior samples. (**a**) DoS. (**b**) Patator. (**c**) DDoS. (**d**) All samples.

4.5. Result Analysis

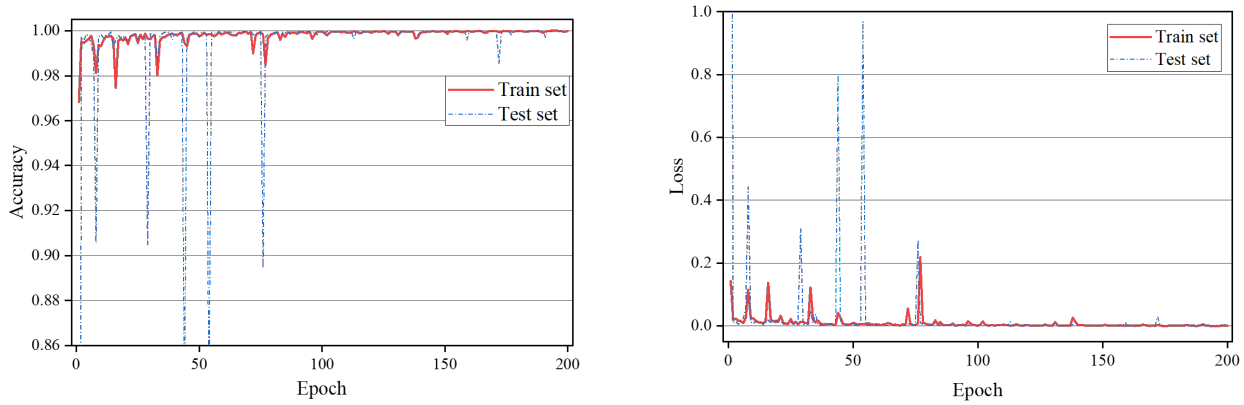
The training set was reduced to include only the 32 effective features. The data were then transformed and prepared to meet the required dimensions for the ResADM model. The transformed training set was used to train the model. Initially, the dataset was trained

on the backbone network built with the ResNet-50 architecture, which had been pre-trained on the UNSW-NB15 dataset. This pre-training enabled the utilization of learned generic features from the pre-trained model for transfer learning in the task of network-attack-behavior detection. During the fine-tuning process, a portion of the convolutional layer weights was frozen, and only the newly added, fully connected layers were trained. This approach reduced the need for parameter fine-tuning, conserved computational resources, and allowed for the efficient adaptation of the features to the categories of the network-attack-behavior dataset.

To ensure the stable optimization of the model parameters during the fine-tuning process, the learning rate was set to 0.001. The fully connected layers and a softmax-classifier layer were added after the backbone network. The fully connected layers retrained the features to align with the categories of the network-attack-behavior dataset and converted the output of the backbone network into a probability distribution, facilitating network-attack-behavior detection. During the training process of ResADM, the model underwent 200 iterations at a batch size of 64. The Adam optimizer was used for optimization. The overall multi-classification loss function employed was SparseCategoricalCrossentropy. The loss was calculated as described in Equation (11), where y_i represents the true class label (not one-hot encoded), and p_i denotes the predicted probability distribution of the model, indicating the probability of belonging to each class.

$$Loss = -\sum_{i=1}^{N} y_i * log(p_i), \tag{11}$$

The trends of accuracy and loss values during the 200 iterations of ResADM training are shown in Figure 6. In Figure 6a, it was observed that the model quickly converged and achieved a high level of accuracy. After approximately 100 iterations, the accuracy stabilized at around 0.99, indicating that the model had successfully learned to accurately classify network attack behaviors. Figure 6b displays the loss values of the model on both the training and testing sets. The low values of the loss function on both sets demonstrated that the model had effectively minimized errors during the training process. This suggested that the model had successfully learned the underlying patterns and features associated with network attack behaviors. The convergence of the accuracy and the low levels of loss indicated the effectiveness of the ResADM model in detecting network attack behaviors. These results demonstrated that the model had generalized well on the given dataset, achieving high accuracy and effectively minimizing errors.



(**a**) (**b**)

Figure 6. Training Curve of the Model over 200 Rounds. (**a**) Accuracy. (**b**) Loss.

Following the completion of the ResADM training, the parameters of the backbone network were updated, and ResADM was evaluated using the validation set, as presented in Table 3. The overall accuracy of ResADM was reported as 99.9%, indicating its high performance in accurately detecting network attack behaviors. Furthermore, the recog-

nition performance for each network category exceeded 99%, highlighting the model's effectiveness for classifying different types of network traffic.

Table 3. ResADM results.

Label	Precision	Recall	F1-Score
BENIGN	0.9985	0.9995	0.9990
DDoS	1.0000	1.0000	1.0000
DoS	0.9995	0.9995	0.9995
Patator	1.0000	0.9990	0.9995

ResADM utilized a purposeful sampling-based data-cleaning method, which contributed to its accurate identification and labeling of attack behaviors within large volumes of CPS traffic. This approach yielded high accuracy across all network traffic categories, approaching or equaling 100%. As a result, it effectively addressed the challenge of data-labeling for CPS attack behaviors. Additionally, ResADM employed a feature-selection method based on feature importance, allowing for data dimension reduction and the elimination of unnecessary features. This enabled the model to identify the relevant features associated with covert or deceptive CPS attack behaviors. As a result, ResADM achieved a high recall rate, demonstrating its ability to effectively identify and capture instances of attack behaviors. The combination of high accuracy, excellent recognition performance across network categories, and the capability to handle challenging data labeling and discover effective features made ResADM a robust model for detecting CPS attack behaviors.

Figure 7 presents the confusion matrix heatmap of the validation results obtained by ResADM. From the figure, it could be observed that due to the similarity in the behavioral features between BENIGN traffic and the DDoS attack behavior, the recall rate and F1-score for BENIGN were relatively low. However, leveraging the concept of transfer learning, ResADM utilized the advantages of pre-training the ResNet-50 backbone network and retained relevant features, which effectively addressed this issue. Consequently, ResADM achieved favorable results across all metrics.

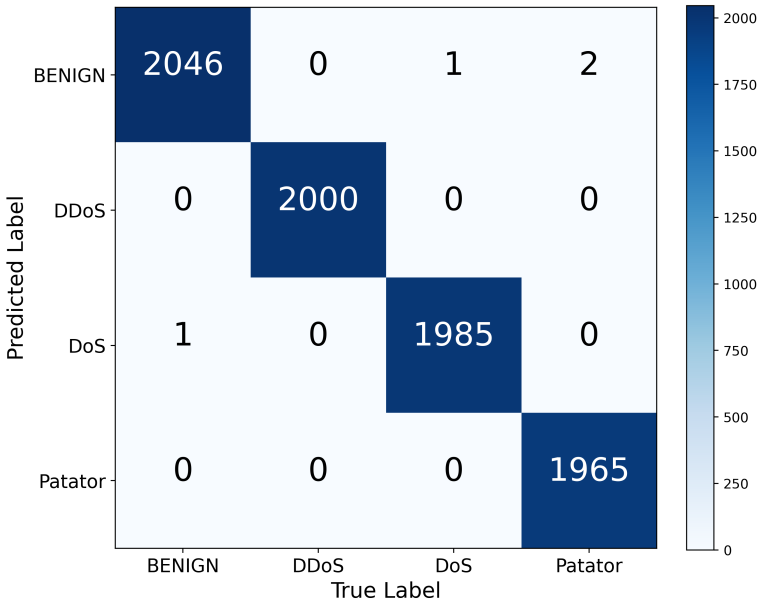


Figure 7. Heatmap of the Confusion Matrix for Validation Results.

To thoroughly validate the advancements of the proposed ResADM method, comparative experiments were conducted in the same simulated environment with the latest relevant works, including CNN [14], DNN [20], and LSTM [21]. The experiments were

carried out using identical dataset distributions and hyper-parameter settings to ensure a controlled experiment. The results of these experiments are presented in Table 4.

Table 4. Results of comparative experiments.

Methods	Accuracy	Precision	Recall	F1-Score
ResADM	0.9998	0.9999	0.9999	0.9999
CNN	0.9967	0.9967	0.9967	0.9967
DNN	0.9977	0.9977	0.9975	0.9976
LSTM	0.9958	0.9951	0.9951	0.9951

The analysis of the results in Table 4 demonstrated that the purposeful sampling-based data-cleaning method proposed in this study had effectively partitioned the dataset of CPS attack behaviors and addressed the challenge of labeling such data. As a result, all the comparative models achieved favorable classification accuracy on this dataset. However, it was important to note that CNN had exhibited higher sensitivity towards image data, while LSTM networks were more suited for sequential data. Therefore, when it came to mining CPS attack-behavior features, the performances of these two models were relatively weaker, as compared to the feature-selection method based on feature importance, as proposed in this study. Furthermore, although the effectiveness of DNN improved with increasing model depth, it still fell short of the performance achieved by the transfer-learning model based on ResNet-50. The proposed ResADM model demonstrated superior performance, as compared to existing methods in terms of classification accuracy and feature extraction for attack-behavior detection. It leveraged transfer learning and feature selection based on feature importance to overcome the limitations of other models and offered a more effective solution for identifying CPS attack behaviors.

5. Conclusions and Discussion

This study introduced ResADM, a network attack-detection method based on transfer learning that aimed to address several challenges in network-attack detection, including data class imbalance, feature selection in high-dimensional data, and weak model-generalization capability. To tackle the issue of data class imbalance, intentional sampling was utilized to construct sample sets with balanced attack categories, effectively addressing the challenge of imbalanced attack categories in large-scale traffic datasets. Furthermore, an importance-based feature-selection method was developed to extract key features of network attack behaviors, addressing the difficulty in identifying attack features. Lastly, a transfer learning model network structure was designed, and a robust network-attack detection model was established by pre-training on the UNSW-NB15 dataset and optimizing the training parameters of the source model. Simulation experiments were conducted on the CICIDS2017 dataset, and promising results were obtained. The experiments demonstrated that ResADM effectively balanced data categories, extracted 32-dimensional attack-behavior features, and achieved a high network-attack-detection accuracy of up to 99.95%. As a result, this research holds significant practicality and feasibility.

However, this study had certain limitations that should be acknowledged. Firstly, the constantly evolving landscape of cyber-attacks has introduced new methods that may not have been adequately represented in outdated datasets. Thus, further validation is necessary to assess the effectiveness of our model against the latest cyber-attack techniques. Additionally, deep-learning models impose significant demands on server performance, so while we endeavored to streamline our model, it requires further validation to evaluate its performance in the face of large-scale DDoS attacks characterized by extensive flooding.

To overcome these limitations and enhance the overall robustness of ResADM, it would be beneficial to broaden the range of experiments and validate the proposed method using additional datasets. This could involve incorporating recently collected data that encompasses a wider range of cyber-attack scenarios. Moreover, comparing the transfer-learning-model network structure with other architecture would enable the selection of

the most suitable model for the given problem. Conducting further research on parameter optimization could also contribute to enhancing the effectiveness of the proposed method.

Author Contributions: Conceptualization, H.W. and L.Z.; methodology, Y.W.; software, H.Z.; validation, Y.W., H.Z. and J.D.; writing—original draft preparation, H.Z.; writing—review and editing, L.Z. and H.W.; visualization, Y.W.; supervision, J.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China (No. 62106053).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are available from the corresponding author on reasonable request. The data are not publicly available due to privatcy.

Acknowledgments: We are extremely grateful to the editors and the anonymous reviewer for their valuable comments and suggestions.

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

Abbreviations

The following abbreviations are used in this manuscript:

APT	Advanced Persistent Threat
BiLSTM	Bidirectional Long Short-Term Memory Network
CNN	Convolutional Neural Network
CPS	Cyber–Physical System
DDoS	Distributed Denial-of-Service Attack
DNN	Deep Neural Network
DoS	Denial-of-Service Attack
LSTM	Long Short-Term Memory Network
LightGBM	Light Gradient-Boosting Machine
MLP	Multilayer Perceptron
ReLU	Rectified Linear Unit
ResADM	ResNet Based Attack Detection Method
ResNet	Residual Network

References

1. Liu, C.G.; Fang, B.X.; Liu, B.X. A hierarchical model of targeted cyber attacks attribution. *J. Cyber Secur.* **2019**, *4*, 1–18.
2. Abu Talib, M.; Nasir, Q.; Bou Nassif, A.; Mokhamed, T.; Ahmed, N.; Mahfood, B. APT beaconing detection: A systematic review. *Comput. Secur.* **2022**, *122*, 102875. [CrossRef]
3. Jian, S.; Lu, Z.; Du, D.; Jiang, B.; Liu, B. Overview of Network Intrusion Detection Technology. *J. Cyber Secur.* **2020**, *5*, 96–122.
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), Big Island, HI, USA, 9–13 December 2019; pp. 1–6.
5. Sinha, J.; Manollas, M. Efficient Deep CNN-BiLSTM Model for Network Intrusion Detection. In Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Pattern Recognition, Xiamen, China, 26–28 June 2020 ; ACM: Pune, India, 2020; pp. 223–231.
6. Wu, T.; Fan, H.; Zhu, H.; You, C.; Zhou, H.; Huang, X. Intrusion detection system combined enhanced random forest with SMOTE algorithm. *EURASIP J. Adv. Signal Process.* **2022**, *2022*, 39. [CrossRef]
7. Saisindhutheja, R.; Shyam, G.K. An efficient metaheuristic algorithm based feature selection and recurrent neural network for DoS attack detection in cloud computing environment. *Appl. Soft Comput.* **2021**, *100*, 106997. [CrossRef]
8. Mahdavi, E.; Fanian, A.; Mirzaei, A.; Taghiyarrenani, Z. ITL-IDS: Incremental Transfer Learning for Intrusion Detection Systems. *Knowl.-Based Syst.* **2022**, *253*, 109542. [CrossRef]
9. Zhang, J.; Ling, Y.; Fu, X.; Yang, X.; Xiong, G.; Zhang, R. Model of the intrusion detection system based on the integration of spatial-temporal features. *Comput. Secur.* **2020**, *89*, 101681. [CrossRef]
10. Liu, Y.; Lyu, Y.; He, Z.; Yang, Y.; Li, J.; Pang, Z.; Zhong, Q.; Liu, X.; Zhang, H. ResNet-BiLSTM: A Multiscale Deep Learning Model for Heartbeat Detection Using Ballistocardiogram Signals. *J. Healthc. Eng.* **2022**, *2022*, 6388445. [CrossRef] [PubMed]

11. Sun, X.; Fu, J.; Wei, B.; Li, Z.; Li, Y.; Wang, N. A Self-Attentional ResNet-LightGBM Model for IoT-Enabled Voice Liveness Detection. *IEEE Internet Things J.* **2023**, *10*, 8257–8270. [CrossRef]
12. Li, L.; Wang, Z.; Zhang, K.; Yang, D.; Xiong, W.; Gong, P. A train bottom parts detection algorithm based on OSE-dResnet neural networks. *Comput. Eng. Sci.* **2022**, *44*, 692–698.
13. Liu, Y.; Su, H.; He, Q.; Shen, P.; Liu, P. An equipment fault detection method based on cloud-edge collaboration variational autoencoder neural network. *Comput. Eng. Sci.* **2023**, *45*, 1188–1196.
14. Yang, L.; Shami, A. A Transfer Learning and Optimized CNN Based Intrusion Detection System for Internet of Vehicles. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 2774–2779.
15. Yilmaz, S.; Aydogan, E.; Sen, S. A Transfer Learning Approach for Securing Resource-Constrained IoT Devices. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4405–4418. [CrossRef]
16. Mehedi, S.T.; Anwar, A.; Rahman, Z.; Ahmed, K.; Islam, R. Dependable Intrusion Detection System for IoT: A Deep Transfer Learning Based Approach. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1006–1017. [CrossRef]
17. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.-Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017. Available online: https://dl.acm.org/doi/10.5555/3294996.3295074 (accessed on 22 August 2023).
18. Wang, Y.; Cheng, J.; Liu, T.; Zhou, Y.; Xiong, Y. A smoke detection method based on fusing multiple network models. *Comput. Eng. Sci.* **2019**, *41*, 1771–1776.
19. Wang, S.C.; Chen, S.P. Improved Abnormal Traffic Intrusion Detection Model Based on Residual Network. *J. Chin. Comput. Syst.* **2023**, 1–9.
20. Liu, J.S.; Zhan, D.Y.; Deng, J.; Wang, L.N. Network Intrusion Detection based on Deep Neural Network and Federated Learning. *Comput. Eng.* **2023**, *49*, 15–21+30.
21. Gao, J. Network Intrusion Detection Method Combining CNN and BiLSTM in Cloud Computing Environment. *Comput. Intell. Neurosci.* **2022**, *2022*, 7272479. [CrossRef] [PubMed]