IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 December 2022, accepted 12 February 2023, date of publication 22 February 2023, date of current version 27 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3247512

RESEARCH ARTICLE

An Asynchronous Federated Learning Arbitration Model for Low-Rate DDoS Attack Detection

ZENGGUANG LIU[1], CUIYUN GUO[2], DEYONG LIU[3], AND XIAOCHUN YIN[2], (Member, IEEE)
[1]School of Information Engineering, Shandong Vocational College of Science and Technology, Weifang, Shandong 261053, China
[2]School of Computer Science and Technology, Weifang University of Science and Technology, Shouguang, Shandong 262700, China
[3]Shandong Software Engineering Technology Center, WeiFang University of Science and Technology, Shouguang, Shandong 262700, China

Corresponding author: Xiaochun Yin (xiaochunyin@wfust.edu.cn)

This work was supported in part by the Key Technologies Research and Development Program of Weifang under Grant 2021GX056, in part by the Foundation for the Talents by the Shandong Vocational College of Science and Technology, in part by the Foundation for the Talents by the Weifang University of Science and Technology under Grant KJRC2021002, in part by the Natural Science Foundation of Shandong Province under Grant ZR2021MF086, and in part by the Key Research and Development Program of Shandong Province under Grant 2019GNC106034.

ABSTRACT Low-rate Distributed Denial of Service (LDDoS) attacks have been one of the most notorious network security threats, which use periodic slight multi-variate time series pulse flows to degrade network quality. Limited by the poor data in a single client, a powerful and satisfactory LDDoS attack detection model is hard to be trained. Federated Learning (FL) is a promising paradigm offering joint learning through multiple clients. We propose an asynchronous federated learning arbitration framework based on bidirectional LSTM (bi-LSTM) and attention mechanism (AsyncFL-bLAM). In the AsyncFL-bLAM, the leader node election algorithm is proposed for constructing the framework of asynchronous federated learning. The proposed bLAM model composed of feature extracter and arbitrator takes on the responsibility of LDDoS detection locally. Furthermore, the novel AsyncFL framework helps to upload and aggregate the bLAM models' parameters asynchronously between leader node and client nodes. Experimental results show that the AsyncFL-bLAM outperforms the state-of-the-art models in accuracy, and reduces the overall communication rounds.

INDEX TERMS Arbitration mechanism, asynchronous federated learning, deep learning, low-rate distributed denial-of-service.

I. INTRODUCTION

With the increasing number of Internet of Things (IoT) devices, network attacks are increasing in both intensity and frequency. Recently, LDDoS attacks are reported as the most common ones in IoT. According to CNCERT [1], as many as 8,423 hacked IoT botnets, with no less than 100 IoT devices, were used to orchestrate and launch LDDoS attacks in 2021. Various variants of LDDoS attacks are found recently. Hivenets [2] could transform a single under-controlled IoT device into an intelligent robot to make autonomous decisions with minimal supervision. A multi-targets LDDoS attack model [3] used the bots' unused gaps between bursts to fire another attacks. The novel LSTM-CGAN [4] and TTS-GAN [5] methods could generate high-quality LDDoS adversarial samples for attacking directly. As we know, LDDoS attacks are a kind of complex large-scale attack behavior with strong time-domain characteristics in IoT network. Unlike volumetric traffic or high-rated flooding attacks, LDDoS attacks show similar patterns in terms of speed and volume to the traffic produced by legitimate clients. Therefore, LDDoS attacks are difficult to identify due to the characteristics of a low average rate. To the best of our knowledge, Blacknurse [6] is the only big-scale low-rate attack found in a real network. This is why there is not enough LDDoS dataset for learning. Another aspect is that when LDDoS attacks are ongoing, the users and attackers are impacted and can not send/receive traffic to/from the network normally. This leads to some packets in the collected datasets being missed, making the training datasets worse.

In order to obtain a stable model with high performance, the intuitive idea is to use datasets that are scattered

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Pau.

in different data centers and integrate them for learning. However, in reality, data centers regard datasets as important business secrets and are not allowed to leak them to others. Fortunately, FL [7], proposed recently, may have the prospects to solve the problem. FL is a distributed machine learning process where clients collaboratively train a model under the orchestration of a central server while keeping the training data decentralized. After resolving the training dataset problem, the next step is to fix missed packets in the training dataset. To solve this issue, we introduce the bi-LSTM method and attention mechanism. The advantage of bi-LSTM can refer and fill the right values in the placeholder of noise since it can learn from the previous values in the sequence and the upcoming values. And the attention mechanism can help to focus on the import features. With them, we can reduce the impact of missed packets and train for a high-performance AsyncFL-bLAM model.

In summary, we make the following contributions in this paper:

- We design an equal time step sliding window method for data pre-processing with the goal of maximizing data utilization.
- We develop a local model based on bi-LSTM and attention mechanism to eliminate the impact of noise data and preserve time dependency during LDDoS attacks detection.
- We propose a leader node election algorithm and design weights to construct the framework of asynchronous FL, and leverage the framework for getting the high accuracy classifier and reducing time complexity with keeping data decentralized.

The rest of this paper is organized as follows: Section II discusses related work. Section III presents the framework of the proposed asynchronous federated learning. And then, we describe the essential parts of this framework, such as the equal time step sliding window method, the bi-LSTM network with attention mechanism, and the detailed method of asynchronous learning. Next, experiments are carried out and discussed in terms of the classification accuracy, precision, recall, and even the time complexity of FL on the public ISCX dataset for estimating the model in section IV. Section V concludes the paper.

II. RELATED WORK

LDDoS attacks have attracted wide attention since they were proposed as far back as 2003. LDDoS attacks reveal that TCP's retransmission timeout mechanism can be exploited using maliciously chosen low-rate gusty attack flow to make TCP throughput fall to a very low rate. To develop the LDDoS attacks defense models and verify the models' effectiveness, large-scale datasets with high credibility and high fidelity are needed. But these kinds of datasets are not enough.

In order to address the issue, researchers attempted to make full use of existing public datasets through better feature representations algorithms or machine learning methods. Diro and Chilamkurti [8] investigated, compared, and tested traditional and deep learning approaches in public datasets. And they concluded that deep learning models were superior to shallow ones in detection accuracy, false alarm rate, and scalability. Yue et al. [9] investigated a new identification approach based on wavelet transform and combined neural networks to classify normal network traffics from LDDoS attacks. Liu et al. [3] proposed to do pre-classify by locality-sensitive features extraction and then made use of Convolutional Neural Networks (CNN) to learn high dimensionality feature representations in pre-classified smaller buckets. However, the datasets used in the above literature are not large enough, which brought the performance bottleneck of detection models. At the same time, other researchers tried to generate a private LDDoS dataset for training. Hong et al. [10] used the NS3 tool to simulate a network for generating and capturing training datasets. Song and Wang [11] proposed LDDoS emulation technologies based on lightweight virtualization, which can construct an LDDoS emulation scenario with 400 routing nodes in a single physical server. Charlier et al. [12] presented the novel framework SynGAN that generated adversarial network attacks using the Generative Adversarial Networks (GAN) based on real attack traffic. Liu and Yin [4] proposed an LSTM-CGAN model towards generating LDDoS adversarial samples. The model extracted time-domain characteristics of LDDoS by LSTM and generated mimicking behaviors of attacks by Condition Generative Adversarial Networks (CGAN) model. Madane et al. [5] extended CGAN to transformer-based one for better long LDDoS attack sequences. These methods help to the fake large-scale training dataset.

However, compared with the ground truth data, the credibility and fidelity of simulation or GAN fake one needs to be further tested. Since the ground truth data in a single cloud datacenter was not enough and fake data was not qualified, researchers tried to use FL technology to adopt multiple real-world dataset across different cloud datacenters. McMahan et al. [13] presented a practical synchronous FL method to the non-IID data distributions. Rahman et al. [14] proposed a synchronous FL-based scheme for IoT intrusion detection. As experimental results and empirical analysis explored, The scheme maintained data privacy availably with higher accuracy. Wang et al. [15] proposed an intrusion detection method based on FL and CNN to solve the problem of training a depth model with high accuracy under the limited label data generated by a single mechanism. Li et al. [16] proposed a novel federated deep learning scheme to detect cyber threats by making use of CNN and gated recurrent units (GRU). Extensive experiments on a real dataset demonstrate the scheme's high effectiveness in detecting various types of cyber threats. But the accuracies of local CNN models used in above FL methods are still below expectation, and the synchronous FL methods themselves are time consumption. Therefore, on the one hand, with the emergence of the LSTM networks, researchers saw their abilities in recognizing long-time dependent LDDoS attack sequences. Sun et al. [17] proposed a hybrid CNN and GRU

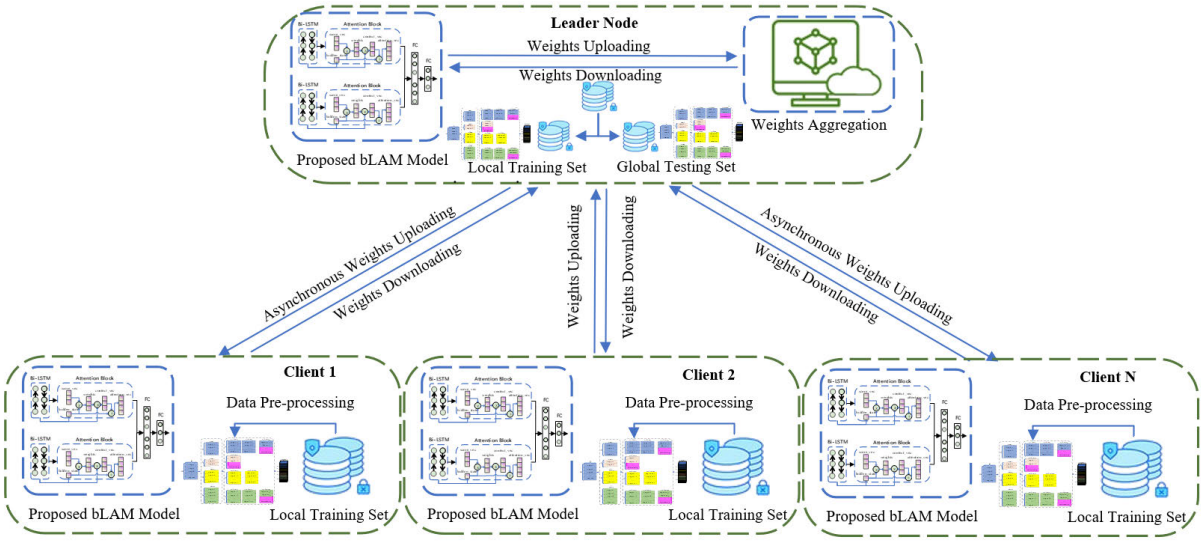

FIGURE 1. AsyncFL-bLAM: the asynchronous federated learning framework.

model to extract deeper spatial and temporal features of LDDoS attacks. Mohammad et al. [18] proposed a novel autoencoder-based anomaly detection system to leverage time-based features (TAE) over multiple time windows for efficiently detecting anomalous DDoS. Zhou et al. [19] proposed a variational LSTM (VLSTM) learning model for intelligent anomaly detection based on reconstructed feature representation. An encoder-decoder neural network associated with a variational reparameterization scheme was designed to learn the low-dimensional feature representation from high-dimensional raw data. Experiments using a public dataset demonstrated that the proposed VLSTM model could efficiently cope with high dimensional issues, significantly improve the accuracy, and reduce the false rate. Furthermore, researchers began to apply LSTM into FL methods. Zhao et al. [20] proposed an effective intelligent intrusion detection method based on federated learning aided long short-term memory framework (FL-LSTM), which can solve the problem of a powerful deep learning model training and intrusion risks at the central server and violate user privacy. On the other hand, the asynchronous FL methods are investigated. Lu et al. [21] performed a rating algorithm to incent powerful nodes asynchronously.

In this literature, we propose an asynchronous FL arbitration framework with local bi-LSTM network and attention mechanism, which can use the advantages of both LSTM-based models and FL frameworks to improve the performance of LDDoS detections and address LDDoS threats.

III. THE ASYNCHRONOUS FEDERATED LEARNING FRAMEWORK BASED ON BIDIRECTIONAL LSTM NETWORK AND ATTENTION MECHANISM

This section introduces the bi-LSTM and attention-based asynchronous federated learning (AsyncFL-bLAM) framework that we proposed for LDDoS attacks detection. Figure 1 illustrates the framework of the AsyncFL-bLAM, which is mainly divided into three parts. The first part is the data pre-processing by the proposed equal time step sliding window method. The proposed method helps to perform traffic analysis and feature extraction on the original dataset. After that, only the feature subset related to the LDDoS attacks is selected for training. The second part is the local arbitration model. An LDDoS defense model is built and trained locally. In it, the bi-LSTM network is used to learn the time sequenced features, and attention mechanism is used to judge which ones are available for detection. The last third part is the asynchronous global aggregation through the federated learning framework. It includes the leader node election algorithm and asynchronous method with weight correction.

A. PRE-PROCESSING: THE EQUAL TIME STEP SLIDING WINDOW METHOD FOR LDDoS FEATURES TRANSFORMATION

In the data pre-processing stage, the main work is to generate a bidirectional stream related parameter dataset, which is used as the input of the LDDoS detection model. Since the number of input neurons of bi-LSTM is fixed, the input feature-length must also be fixed. Therefore, we propose an equal time step sliding window method, as shown in Figure 2. By this, we can not only get the fixed-length chronological order samples, but also enlarge the training dataset.

Firstly, we analyze and extract the features. As known, the wave-shapes of LDDoS flooding attack traffic are rectangle, and they have observable burst features. That is, the normal network traffic is stable, while LDDoS network traffic is gusty. Hence, we can sample the network stream at 1-min intervals and define IP packet statistical features of network flow as $IP_s = \sum P_{\Delta t}$, where P is the number of packets.

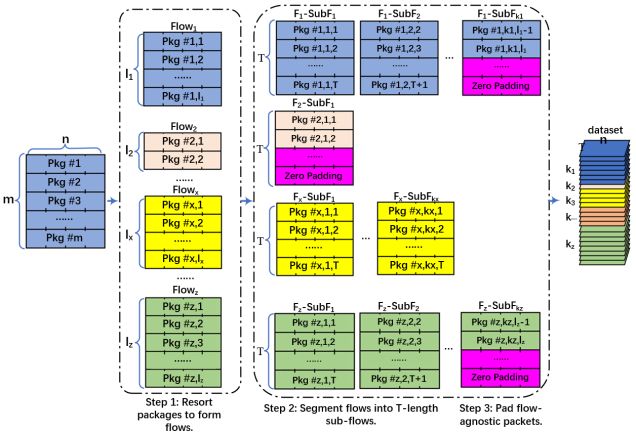

FIGURE 2. The equal time step sliding window method.

TABLE 1. The LDDOS time domain features based on network flow.

Num	Features	Descriptions
1	IP_s	The domain value of the IP package of flow at 1-min intervals based on a threshold.
2	FD	The domain value of Flow Duration based on a threshold.
3	LFP	The domain value of Length of First Package based on a threshold.

In the definition of IP_s, when multiple IP attack sources send useless packets to a destination IP address, the number of IP address packets increases in Δt time. When an IP attack source sends useless packets to multiple destination ports of a target host in Δt time, the number of different destination port numbers also increases abnormally. Including IP_s, some extra connection features and flow similarity features are extracted in Table 1. FD is used to indicate the communication time, which becomes one of the most common parameters for LDDoS botnet detection. LFP is the indicator of low-level protocol, which is an important parameter to different botnets.

After the above modeling, the flow data is sorted in a sequential time serial. Every time step is related to the front and back ones. That is, the LDDoS training dataset is a time-related information set, which can be represented by a matrix comprised of LDDoS training data in the same time step. The shape and content of representing the matrix of single-stream looks like (1), where IP_s, FD, LFP are various dimensions, and t represents the time step of the stream. The t may be different in different streams. If taking all the streams into account, the shape of the matrix will be $m * n$, where m represents the total number of packets in all streams, n means the dimension.

$$\begin{bmatrix} IP_{s0} & FD_0 & LFP_0 & \dots \\ IP_{s1} & FD_1 & LFP_1 & \dots \\ \dots & \dots & \dots & \dots \\ IP_{st} & FD_t & LFP_t & \dots \end{bmatrix} \tag{1}$$

Next, a sliding window with the same step size T is used to segment LDDoS time series blocks on the stream. If the number of messages in the flow is less than step T, it is filled with zero. This can facilitate bidirectional LSTM learning and has no impact on the final model. Since these padding data, same as the real noise data, can be treated as noise data to handle later. Till now, all the feature sequences are reshaped into the tensor of $\sum_{k=1}^{n} k_i * n * T$, where m and k_j is shown in (2).

$$\begin{aligned} m &= \sum_{i=1}^{n} l_i \\ k_j &= \begin{cases} l_j - T + 1 & l_j \geq T \\ 1 & l_j < T \end{cases} \end{aligned} \tag{2}$$

After the above conversions, the label in the original stream can be added to the corresponding feature block to form several smaller matrixes for training. From Figure 2, we can see that package-based features are converted to time series-based features. Under these operations, the model can not only learn the characteristics of the current package but also the characteristics of the previous (T-1) and subsequent (T-1) packets of the current package.

B. LOCAL ARBITRATION MODEL: THE LDDoS ATTACK DETECTION MODEL BASED ON BI-LSTM NETWORK AND ATTENTION MECHANISM

In reality, the collected LDDoS attack traffic is composed of a sequence of attack packets, but there are missing packets in this sequence. These missed packets are hurt to the final predicting result. As shown in Figure 3, the bidirectional LSTM network of the proposed bLAM model can refer and fill the right values in the placeholder of missed packets since it can learn from not only the previous values in the sequence but also the upcoming values. The attention mechanism in the bLAM model can improve the role of key features in the final decision-making through the redistribution of parameter weight. Thus, the hybrid of bi-LSTM network and attention mechanism can learn the time-domain features of forwarding and backward packets and bring the double improvement of precision and recall. In the data plane of Figure 3, the segmented LDDoS data is fed into bidirectional LSTM network, which is used to learn the valid information from noise data in their forward and backward hidden states. Sequentially, an attention mechanism as is used to merge important features together and choose the critical features by redistributing the weights. At last, two fully connected layers are used for classification. Some key technologies are used in the proposed model to avoid the over-fitting problem, such as the dropout layer with the dropout rate of 0.3 and the L2-normalization loss function. The detailed layers information, output shape, and parameter number of each layer are listed in Table 2. There are 111,873 parameters totally in the proposed model. Notes: the layers in attention mechanism and dropout, whose parameter number is 0, are hidden in the table.

1) THE BIDIRECTIONAL LSTM NETWORK

As Figure 4 shown, the bidirectional LSTM network in the proposed bLAM has two LSTM stacked on top of each other.

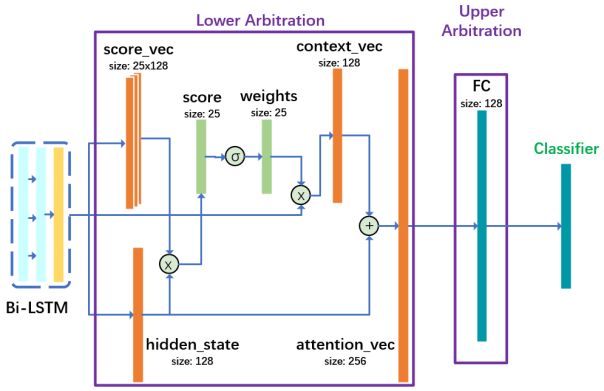

FIGURE 3. The layer-wise structure of proposed bLAM model.

TABLE 2. The layers of proposed bLAM model.

Layer (type)	Output Shape	Param Number
bidirectional (Bi-LSTM)	(None, 25, 128)	46,080
attention_score_vec (Dense)	(None, 25, 128)	16,384
attention_vector (Dense)	(None, 128)	32,768
dense (Dense)	(None, 128)	16,512
dense_1 (Dense)	(None, 1)	129

Each of the LSTM layers consists of 128 neurons. One of them is used to learn from the forward direction, while the other is learning from the inverse direction. The outputs of the two layers are integrated into the subsequent layer so that each neuron in the subsequent layer can get the complete sequence context information. Equation (3) explains the operations performed in bidirectional LSTM cells, where x_t is the input at timestamp t. w's are the weights of gates of LSTM cells. h_t and h'_t are the forward and backward output, respectively. The output gate og_t keeps the information about the bidirectional steps. Based on this, bidirectional LSTM network solves the problem of vector gradient disappearing in the time domain during the LDDoS learning process, realizes the preservation of a large range of temporal information, and improves the ability to extract association features of long-time-interval context messages in the case of packet loss.

$$\begin{aligned} h_t &= tanh(w_i x_t + w_2 h_{t-1}) \\ h'_t &= tanh(w_3 x_t + w_5 h'_{t-1}) \\ og_t &= tanh(w_4 h_t + w_6 h'_t) \end{aligned} \tag{3}$$

After the generated vector of chosen features is fed to the bidirectional LSTM network, an L2 regularizer and an activation *tanh* (*tanh*(.) is a hyperbolic tangent function) are adopted to finish normalize to help reduce over-fitting. When running the network, the vector with a size of $T * o$ is combined by the next layer, where o is the number of the hidden neurons of each bidirectional LSTM cells. In our experiment, after feature extraction of the bidirectional LSTM layer, the feature dimension is expanded to 128. However, the contributions of these 128 features to the detection results are not equal. Therefore, this paper

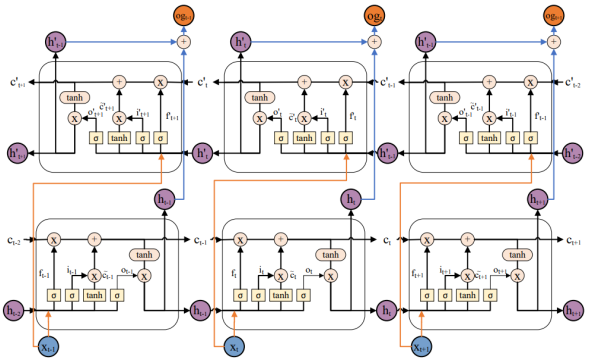

FIGURE 4. The basic structure of three-stage bi-LSTM networks.

introduces an attention mechanism and a fully connected layer as lower and upper arbitrations.

2) THE ATTENTION MECHANISM

An attention function can be described as mapping a query and a set of key-value pairs to an output. Thus, we use it to redistributed the weights of features for filtering the important features to classify. As Figure 5 shown, the attention mechanism in the bLAM model is used to extract single feature attention and feature relation attention so as to select the features that have a significant influence on the results. Firstly, attention mechanism calculates a *score* of a group of *score_vec* by dot-product function for each feature. And then, all the *scores* are running by *softmax* function for scaled *scores*, and stored into *weights*. Next, these *weights* are aligned and summed up for *context_vector* as (4). μ_t means the representation information of hidden layer, μ_d is the similarity of feature vectors. α_t is the normalized weight. Here, the reason why the dot-product attention is chosen instead of additive attention is that the former has advantages in computing and saving memory.

$$\begin{aligned} \mu_t &= tanh(w_d h_t + b_d) \\ \alpha_t &= \frac{exp(\mu_t \mu_d)}{\sum_t exp(\mu_t \mu_d)} \end{aligned} \tag{4}$$

3) THE OTHER LAYERS OF bLAM MODEL

After the above transformation, there are two fully connected layers in the end of bLAM model (as seen in Figure 3). They are playing different roles. One fully connected layer is as upper arbitration to reduce the dimension of the features, while the other one is as classifier to judge whether the traffic is an LDDoS attack or not and produce the classification outcome. Thus, *sigmoid* and *softmax* activations are used for different purposes.

To avoid the over-fitting problem, a dropout layer with a 0.3 dropout rate is followed by the bidirectional LSTM network, which permits the hidden layer to drop out certain neurons during training phase randomly. During model compiling, the binary cross-entropy loss function is used to calculate the loss in the training dataset and validation dataset.

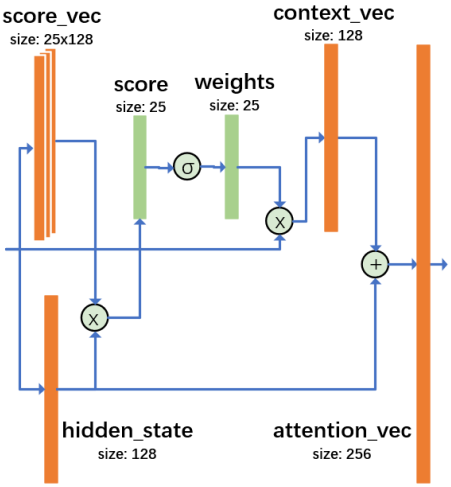

FIGURE 5. The attention mechanism (lower arbitration) of proposed bLAM model.

And Adam optimizer is used to adjust weights and biases through backpropagation.

C. AsyncFL: THE NOVEL ASYNCHRONOUS FEDERATED LEARNING FRAMEWORK

As known, federated learning can protect data privacy. But the traditional federated learning frameworks are easily impacted by abnormal parameters or gradients during global aggregation. A stable model cannot be trained availably sometimes, once the abnormal parameters or gradients are too many. The proposed AsyncFL framework reduces the impact of abnormal parameters on the overall parameter aggregation and improves the robustness of the algorithm. It has two main improvements compared with traditional ones. The one is the leader node election algorithm. Through this algorithm, the best node is chosen for global aggregation. The other one is the asynchronous FL method with weight correction. It helps to finish asynchronous global aggregation.

1) THE LEADER NODE ELECTION ALGORITHM

Federated learning happens among multiple cloud computing data centers, which belong to different autonomous organizations. Each data center has a different dataset size. As a result, it is difficult to forcibly specify a data center as the leader node for parameter summarization. Therefore, it is necessary to select an appropriate data center as the leader node through negotiation among these participants. In this paper, the proposed election conditions are that the node that has a large IP address and more latest training data will be the leader node.

To do this, the ballot with the content of (*IP*, *datasize*, *newdatasize*) needs to be passed during the election phase. The *datasize* means the count of training data in the local current data center. The *newdatasize* means the count of up-to-date training data according to the preset time threshold. When the election begins, each node first calculates its own *newdatasize* and votes for itself. Next, it broadcasts the ballot to other nodes. When other nodes receive the ballot, they record the carried *datasize* for that node immediately. It begins to calculate and compare the ratio by (5). If the ratio exceeds or equals the preset threshold, it will insist on electing itself; otherwise, it will record and elect the opposite node in the next round. If it does not exceed the preset threshold, the *IP* address is compared, and the one with a larger *IP* address is selected. In this way, after several rounds of the election, the node that finally obtains more than half of the ballots is elected as the leader node. Pseudocode is shown in Algorithm 1.

$$ratio = (100\% - \frac{NewDataSize_r}{NewDataSize_l + NewDataSize_r}) \quad (5)$$

Algorithm 1 The Leader Node Election Algorithm

Input: (*IP*, *DataSize*, *NewDataSize*)
Output: The elected leader node
Global Parameters: gRatioThresh

The process of receiving ballot:

```
Calculate the new data ratio by (5);
if ratio > gRatioThresh then
    Cache the ticket (IP, DataSize, NewDataSize);
else {IP < incoming IP}
    Cache the ticket (IP, DataSize, NewDataSize);
end if
Count the number of tickets selected;
if a number of tickets selected > 50% then
    Mark as master node;
end if
```

The process of sending ballot:

```
if not exist cached tickets then
    vote self by (IP, DataSize, NewDataSize);
else
    send the cached tickets;
end if
```

At the same time as completing the leader node election, the leader node also knows the dataset size D of all participating nodes by summarizing the datasize field in the ballot. If the data size of the leader node is big enough, it can segment its own dataset into the training and testing part. In this way, the leader node can not only act as the computing node to participate in the overall training but also complete the parameter update, which speeds up the generation of the detection model.

2) THE ASYNCHRONOUS METHOD WITH WEIGHT CORRECTION

During federated learning, the accuracy of uploading parameters and learning progress of different nodes are not the same, which is caused by the different number of training samples

and the basic computing power of each node. Thus, using the traditional synchronous methods to update the global parameters equally in the leader node is not reasonable. The asynchronous parameters obtained in the leader node should be modified according to the detection effect, which can solve the imbalance problem of parameters updating in asynchronous federated learning.

Considering that the number of samples of each node has been clearly known during the process of leader node election, the sample weight is introduced. In detail, the sample weight γ_s^i is the ratio of the number of samples owned by the submitted parameter node to the total number of node samples. The sample weight solves the problem of sample imbalance in asynchronous training. The calculation formula is shown in (6). Among them, D_i is the sample number of the i^th node.

$$\gamma_s^i = \frac{D_i}{D},\ D = \sum_{i=1}^{n} D_i \tag{6}$$

During electing, partial test dataset is also reserved in the leader node; thus, the normalized test weights are introduced. That is, when the leader node receives the updated parameters, these parameters are loaded into the local model of the leader node, and the detection accuracy is calculated in the reserved test set as the test weights. The test weights reflect the contribution of nodes to the whole model and also solve the problem of inconsistent learning progress caused by asynchronous training. After this, the test weights of different nodes are processed by *softmax*, and the exponential normalized test weights γ_t^i are obtained.

Based on the sample weight and normalized test weights, the model parameters θ' are modified. The calculation formula is shown in (7).

$$\theta' = \theta * \frac{(\gamma_s^i + \gamma_t^i)}{2} \tag{7}$$

During the process of asynchronous learning, the global parameters are optimized according to the FedAvg algorithm [13], [22] after the parameters are modified by weights. After optimization, the leader node sends the updated parameters to all other nodes for the next iteration. After applying the asynchronous update algorithm based on weight correction, the sample imbalance problem and the training iteration speed problem are resolved so that the obtained model also has certain robustness.

IV. EXPERIMENT EVALUATION

A. LDDoS DATASET INTRODUCTION AND REVISE

The ISCX-2016-SlowDos [23] and the Friday dataset of the 1st week of 1999 DARPA [24] are chosen as attack and normal traffic. The ISCX-2016-SlowDos is captured in a testbed environment with a victim web server running Apache Linux v.2.2.22, PHP5, and Drupal v.7 as a content management system. DARPA samples were provided by the Air Force Research Laboratory for the real-time evaluation of network security. Considering that we only need to detect low-volume DDoS attacks, we remove the high-volume ones and intermix the left ones with DARPA samples.

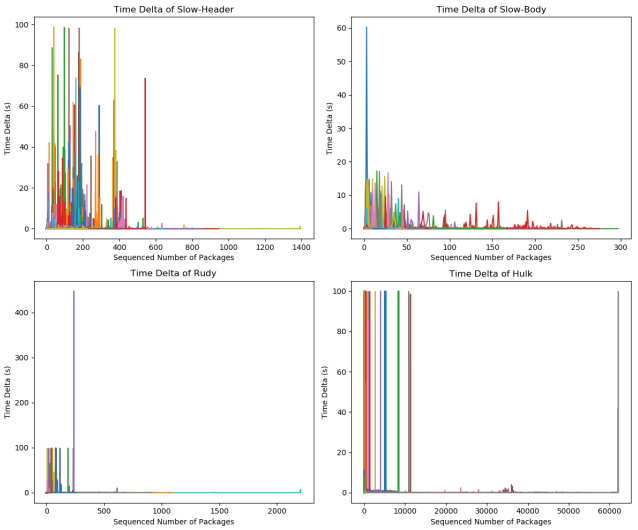

FIGURE 6. The time delta of sequenced packages of LDDoS attack flows.

In order to verify that whether the proposed model is available in a smaller dataset with noise data, we take 80,000 samples (40,000 is for LDDoS data, while 40,000 is for normal traffic. This partition can avoid the problem of data imbalance). In addition, 10% values in both training and testing are changed to random ones, which is simulating the noise data. And then, we check the time deltas of the LDDoS (slow-header, slow-body, hulk, and rudy) flows and normal traffics. As indicated in Figure 6, different variants of LDDoS attacks have different attack time deltas and packet numbers. For all, the time deltas of LDDoS attacks are greater than 30 seconds, and the packet number is bigger than 10. Figure 7 shows the normal traffics. Their access times are less than 20 seconds, and the packet number is smaller than 5. Furthermore, we make a comparison of the mean value between the original dataset and the revised dataset. We take the average package size as an example to draw their boxplot, as indicated in Figure 8. The minimum, maximum, and Q2 are changed, but they can still be used to classify attacks.

In order to verify the proposed federated learning method, we set up three data centers; the data volume of the three data centers is 30,000, 50,000, and 70,000, respectively. The data centers compete for the leader node. According to the leader node election algorithm, the third data center is selected as the leader node, and its dataset will be divided into 40,000 training data and 30,000 testing data.

B. EXPERIMENT ENVIRONMENT SETUP

The experiment was conducted on a laptop with Intel(R) Core (TM) i5-7200U CPU. GPU is not used here. The library Keras with TensorFlow as the backend was imported into the proposed model implementation.

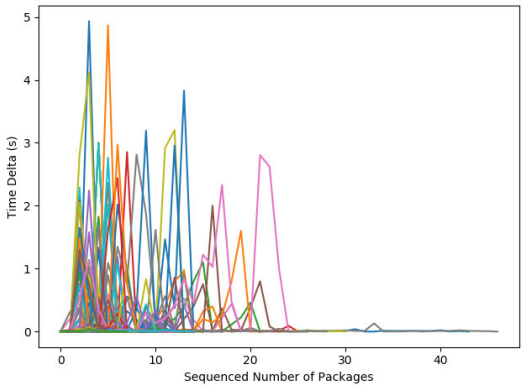

FIGURE 7. The time delta of sequenced packages of normal flows.

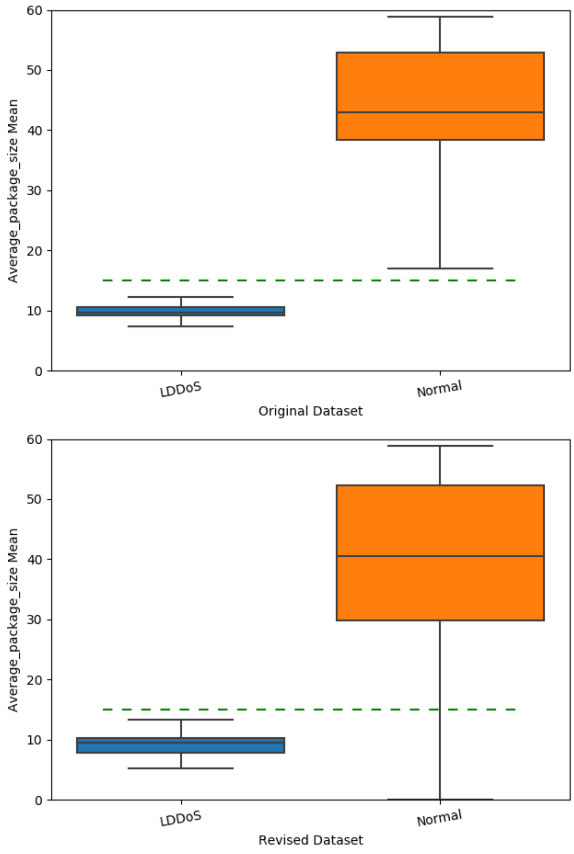

FIGURE 8. The boxplots of average package size of original and revised dataset.

C. ABLATION EXPERIMENT AND HYPER-PARAMETERS TUNING

1) EQUAL TIME STEP SLIDING WINDOW SIZE TUNING

Different sizes of sliding windows represent information of different time spans. But as the size of the sliding window increases, the data correlation becomes weaker. Therefore, the size of the sliding window needs to be set with an upper threshold. Once the upper threshold is exceeded, it will lead to a decrease in accuracy and an increase in loss. In order

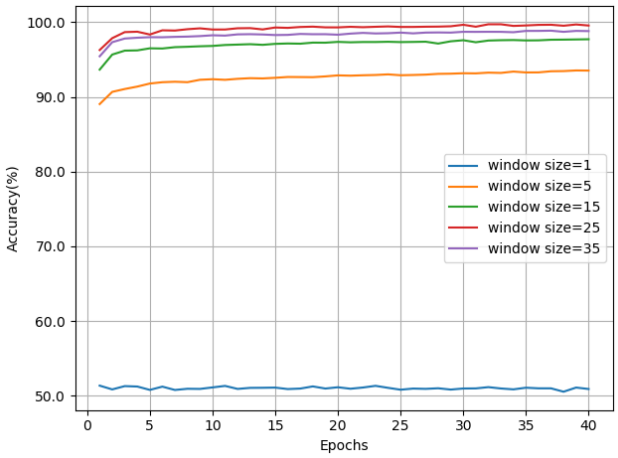

FIGURE 9. The accuracy comparison of bLAM model in different window size.

to obtain the optimized value of the sliding window size, we compared the detection accuracy and loss under the size range of [1, 5, 15, 25, 35] in the experiment.

Figure 9 and Figure 10 show the detection accuracy and cross-entropy loss under different sliding window sizes, respectively. When the sliding window size is set to 1, the training data does not contain time domain-related information, and the model also degenerates into a BP-like network. It is difficult for such a model to learn the difference of strong correlation in the time domain between LDDoS attack flows and normal traffics. As a result, when the sliding window size is 1, the accuracy of the model is very low, maintained at about 53% (the blue line in Figure 9), but the cross-entropy loss of the model is as high as 0.7 (the blue line in Figure 10). As the size of the sliding window increases, the accuracy first increases rapidly and then slows down. When the sliding window size is 25, the maximum accuracy and minimum loss are reached. However, when the sliding window size further increased, the accuracy decreases and the cross-entropy loss continues to increases. Analysis shows that the large sliding window introduces useless long-term dependencies into the model, resulting in the above scenarios. Thus, this paper uses the sliding window with a value of 25 as the tuning size.

2) BI-LSTM AND ATTENTION ABLATION EXPERIMENT AND HYPER-PARAMETERS TUNING

The validation of the bidirectional LSTM networks and attention mechanism was tested and compared through ablation experiment of bLAM model. In Figure 11 and Figure 12, we compare the accuracy and loss among LSTM, bi-LSTM, bi-LSTM with dropout, and bi-LSTM with attention (proposed bLAM model). The training is carried out in 40 epochs. Obviously, the proposed bLAM model has the best performance, with an accuracy up to 98.00%, and cross-entropy loss is less than 0.1. The epoch 40 with the

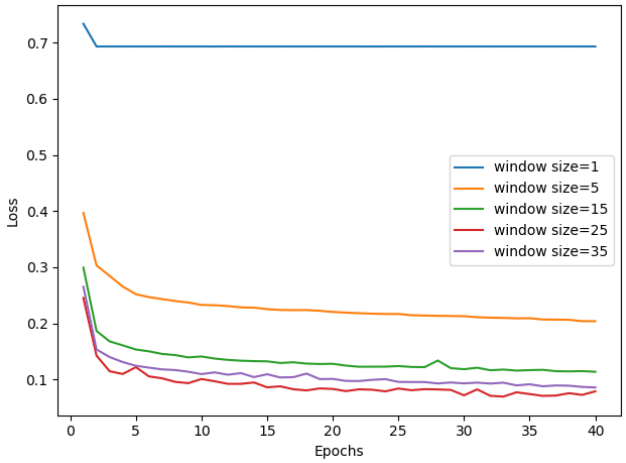

FIGURE 10. The loss comparison of bLAM model in different window size.

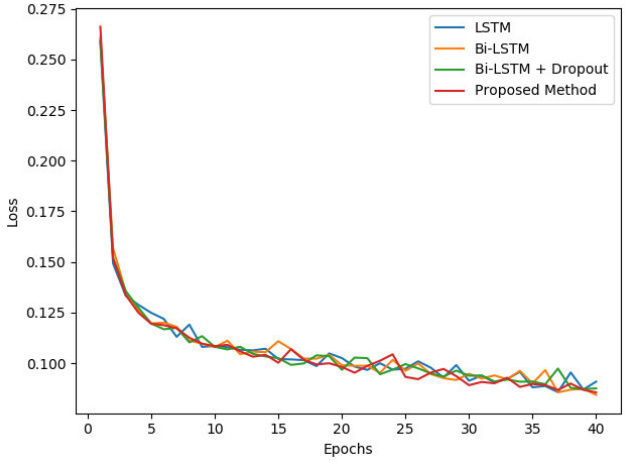

FIGURE 12. The loss comparison of ablation experiment in validation set.

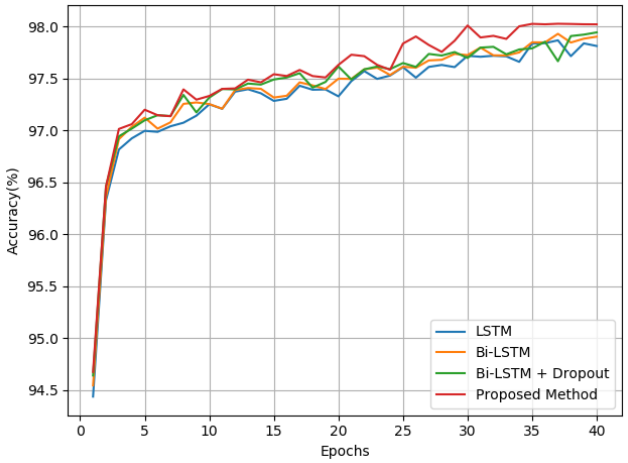

FIGURE 11. The accuracy comparison of ablation experiment in validation set.

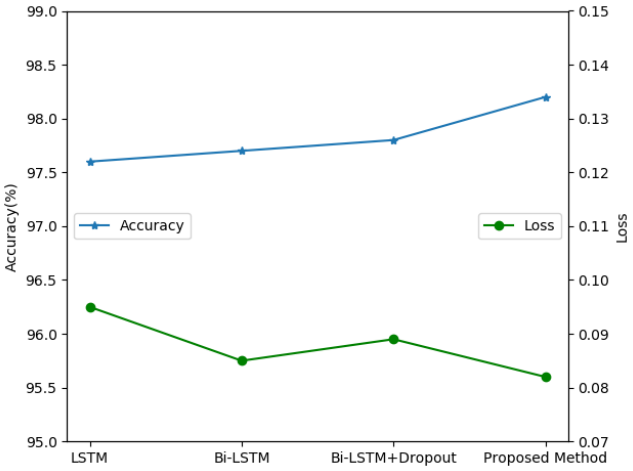

FIGURE 13. The accuracy and loss comparison of ablation experiment in test set.

best performance is chosen to validate in the test dataset. Figure 13 presents the results in the test dataset. The proposed method still has the best performance while considering both accuracy and loss. Its accuracy is 98.20%, which is a little higher than other RNN methods, and its loss is also better than others. Thus, bi-LSTM, Dropout method, and Attention mechanism are available and added to our proposed bLAM local model.

After the model is obtained, the hyper-parameters are well-tuned through testing the output of different values. A greedy-wise tuning method named grid search is used to achieve as close to the optimal performance as possible. The different dropout values and the number of neuron units are testing in the provided dataset. As shown in Figure 14, when the dropout rate is 0.3, both the loss and accuracy have the best performance. Thus, we use 0.3 as the dropout rate in the proposed model.

After applying the selected dropout rate, a different number of neuron units in the bidirectional LSTM layer is tested by grid search. As shown in Figure 15, the best number is 128.

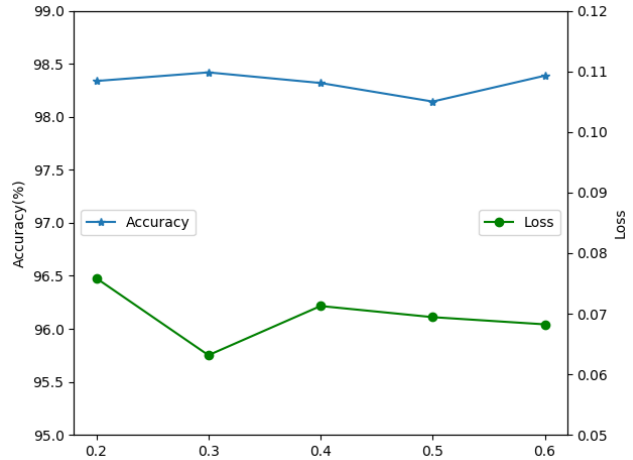

FIGURE 14. The accuracy and loss comparison in different dropout rate.

It can learn the representation features and get an accuracy of 98.87%.

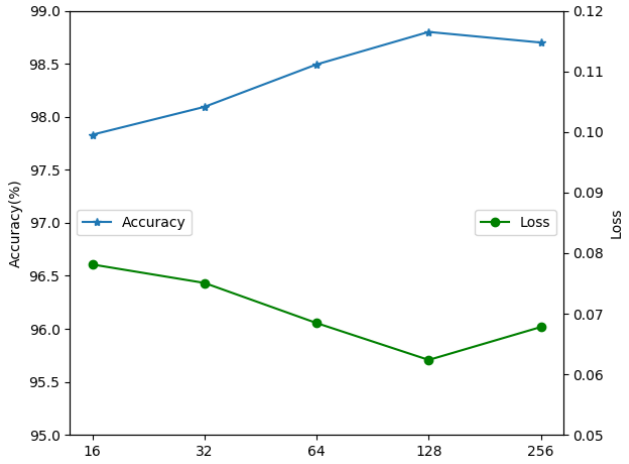

FIGURE 15. The accuracy and loss comparison in different neuron units.

TABLE 3. The table of hyper-parameters.

Hyper-parameters	Values
Dropout rate	0.3
Neuron Units of bidirectional LSTM	128
Dominant Layers	Bidirectional LSTM, Attention, Dropout

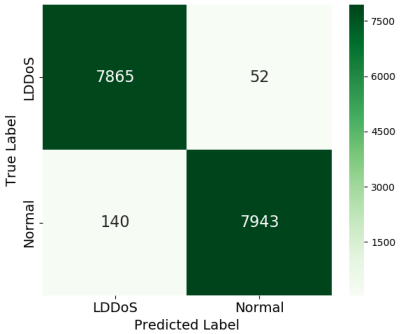

FIGURE 16. The heatmap of classification of bLAM model.

Learning rate is not necessary since Adam is used during model compiling. The optimizer Adam will help to the best learning rate automatically.

Thus, when the above exploration is done, the hyper-parameters are gotten. Table 3 lists the value of the tuned hyper-parameters.

D. EXPERIMENTAL RESULTS

1) EXPERIMENTAL RESULTS OF bLAM MODEL

Figure 16 shows the confusion matrix of the revised ISCX-2016-SlowDos dataset. We can see that the prediction accuracy is 98.80%, the precision is 99.34%, and the recall is 98.25%, which indicated that the proposed model has good performance for classifying the network LDDoS attacks. In addition, we also compare shallow classifiers and other classics CNN/RNN models.

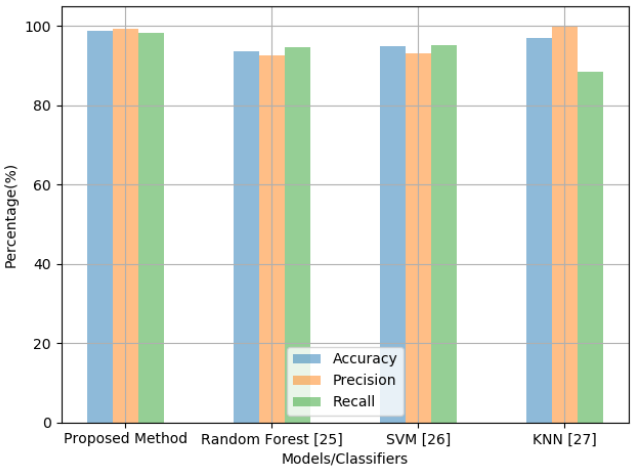

FIGURE 17. The performance comparison with shallow classifiers.

In Figure 17, for the purpose of comparing with shallow machine learning classification, this paper sets the sliding window size to 1. Since the shallow classifier does not have the ability to analyze time steps. The experimental result shows that Random Forest [25], SVM [26] and KNN [27] indexes are significantly lower than the proposed classifier. This verifies a huge gap between the shallow classifiers and the proposed one in the aspect of historical feature extraction.

As shown in Figure 18, all RNN models, including TAE [18], VLSTM [19], and FL-LSTM [20], have high precision and recall rate, indicating that the RNN model can learn the representation of time sequence-related features and shows that the time sequence-related features of LDDoS are the main influencing factors. Compared with the LSTM-like models, the indexes of the CNN-LSTM model [28] are slightly lower. The reason is that the dataset does not contain spatial information, making the CNN layer unable to play its advantages in spatial feature extraction. Based on same idea, the hybrid CNN+GRU model (HY-CNN+GRU) [29] also used CNN and GRU, but it tried to optimize the hyper-parameters of model by improved sailfish algorithm. From the results, the performance was not as expected. The FSL-SCNN model [30] has an improvement compared with CNN methods. This is because siamese encoding network helps to measure distances of input samples based on their feature representations, and alleviate the loss of key features. The proposed bLAM model has better performance than other models. This is because the proposed method further enhances the bidirectional LSTM-based model through an attention mechanism. It improves the importance of critical features in the final decision-making by redistributing weights, which brings about double improvement of precision and recall rate.

2) EXPERIMENTAL RESULTS OF AsyncFL FRAMEWORK

In order to verify the effectiveness of the proposed asynchronous federated learning framework, the accuracies and

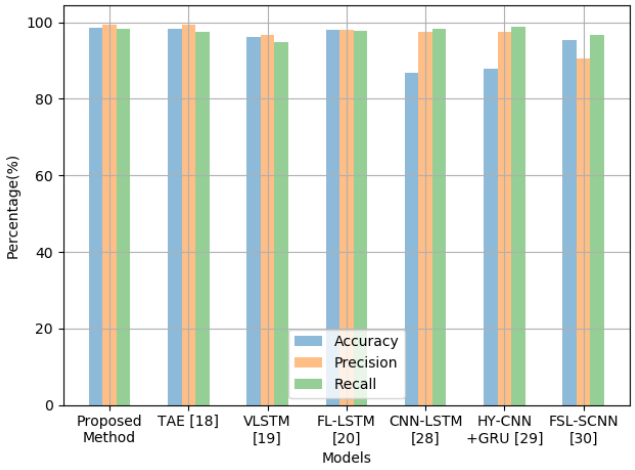

FIGURE 18. The performance comparison among different RNN models.

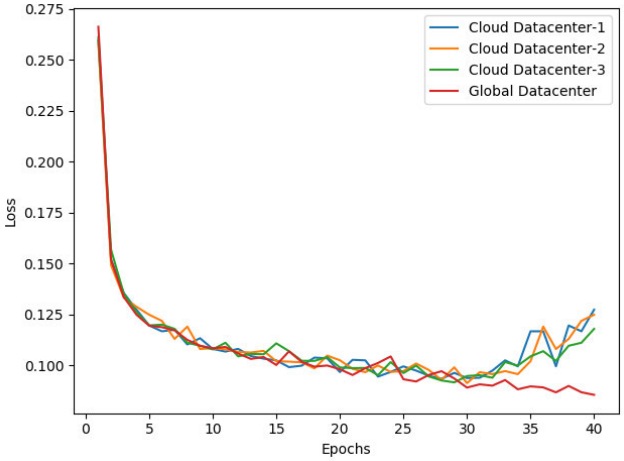

FIGURE 20. The loss comparison of AsyncFL and local learning (5% dataset).

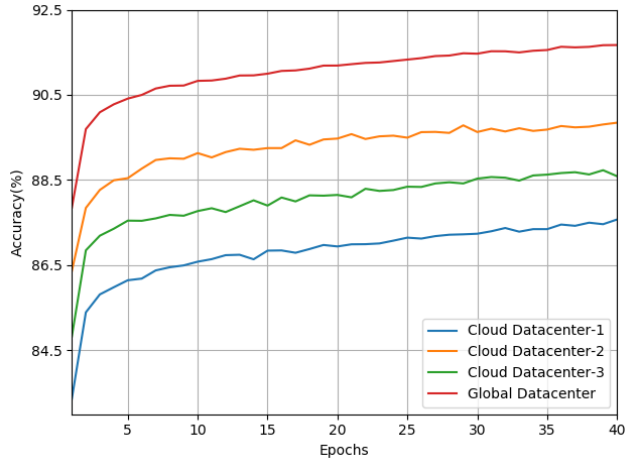

FIGURE 19. The accuracy comparison of AsyncFL and local learning (5% dataset).

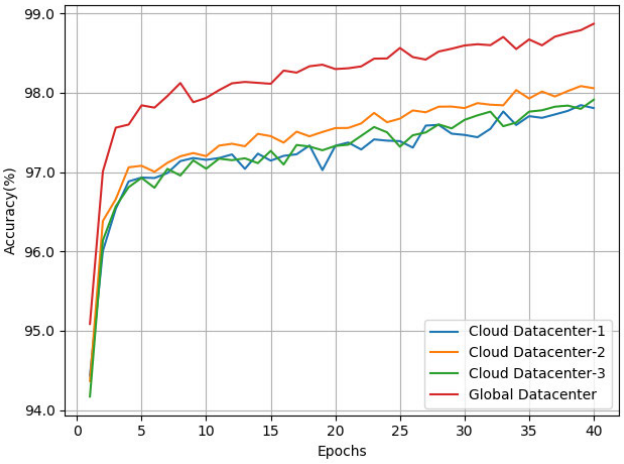

FIGURE 21. The accuracy comparison of AsyncFL and local learning (full dataset).

cross-entropy losses of each node in different epochs under 5% dataset and fully dataset are compared.

In the case of the 5% dataset, the data used for training in the three cloud centers is 1,500, 2,500, and 3,500 (of which 2000 are used for local learning and 1500 for federal learning testing), respectively. The experimental results are shown in Figure 19 and Figure 20. Figure 19 shows the accuracy of different epochs of the bLAM model in local learning and federated learning. Even if the accuracies of the local nodes are not high, the global accuracy is greatly improved after applying the proposed federated learning framework. Figure 20 shows their cross-entropy loss. It can be seen that the losses of local nodes tend to grow up with epochs increasing. The reason is that the size of training data of each node is small and over-fitting occurs. On the contrary, the cross-entropy loss of the proposed AsyncFL framework is smooth, indicating that the AsyncFL framework avoids over-fitting through weight correction.

In the case of the full dataset, the data used for training in the three cloud centers are 30,000, 50,000, and 70,000, respectively (40,000 for local learning and 30,000 for federal learning testing). The experimental results are shown in Figure 21 and Figure 22. Figure 21 shows the accuracy of different epochs of local learning and federated learning under a large amount of data. Similarly, the accuracies of local nodes are not high, and the accuracy is greatly improved by the proposed AsyncFL framework. Figure 22 shows their cross-entropy loss. Compared with local learning, the cross-entropy loss of the bLAM model is smoother and lower. It verifies that the AsyncFL framework improves the weight of effective parameters and makes the classification more accurate.

To verify that the AsyncFL framework is better than other federated learning frameworks, we compared it with synchronous (DeepFed) [16] and classics asynchronous (PAFL) [21] frameworks. As mentioned above, FedAvg is adopted to do parameters aggregating. We add random delay in asynchronous updating cluster frameworks when local models' weights are uploading. This makes the experimental

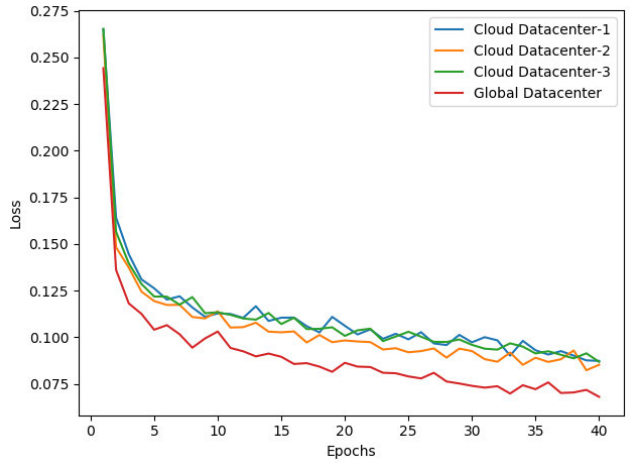

FIGURE 22. The loss comparison of AsyncFL and local learning (full dataset).

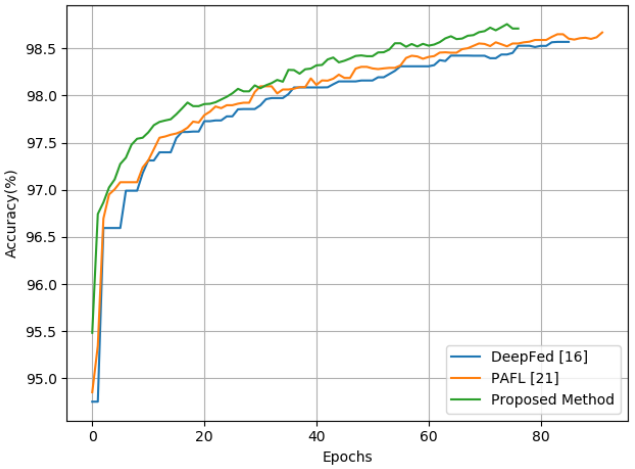

FIGURE 23. The accuracy comparison among federated learning frameworks.

results more clearly. As shown in Figure 23, the bLAM model's accuracy under the AsyncFL framework is better than ones under other frameworks in the same epoch, and bLAM model's accuracy grows smoother. The weighted correction method acts as a regularization part, which helps prevent the over-fitting of learning. Thus, these abnormal parameters have little influence during the global parameters aggregating. This also shows that the training model under AsyncFL framework has high robustness. At the beginning of the curve under AsyncFL framework, there is a significant oscillation. The reason is that the initial parameters of a data center arrive late but are unconditionally aggregated into the global parameters, which brings the robustness problem of the framework. Per the final accuracy, the bLAM model under AsyncFL framework has the highest accuracy of 98.68%, which is better than synchronous DeepFed update (98.6%) and asynchronous PAFL update (98.6%). The reason is related to the accuracy of the bi-LSTM networks in bLAM model. The optimal accuracy of the bi-LSTM networks on a single small dataset can reach 98.5%.

V. CONCLUSION

In this paper, we have addressed the problems of LDDoS attack detection over multiple cloud datacenters using AsyncFL-bLAM framework. The proposed framework is partitioned into three stages: pre-processing, local training based on bLAM model and global aggregation based on AsyncFL framework. In the pre-processing stage, we propose an equal time step sliding window method feature splitting in order to get a good feature engineering. In the local training stage, the bLAM models are trained in each local datacenter. After models are trained, the models' parameters are asynchronously uploaded to the central participant for the joint training. Experiments demonstrate that our proposed bLAM model has good effectiveness in both detection accuracy and time consumption for IoT network-based attacks. The comparison among different classifiers/models demonstrated that the proposed bLAM model has a higher accuracy. And federated learning experiments show that the models under the proposed AsyncFL framework can get higher accuracy with the least epochs.

Nevertheless, the LDDoS attacks are composed of multi-variate flows. Thus, they can be handled by a multi-head arbitration. In the near future, we will design a novel multi-head bi-LSTM arbitration model to down the detection time. With the application of edge computation, some LDDoS attack datasets are in the edge nodes. We will also further expand the AsyncFL model for involving edge nodes.

REFERENCES

[1] CNCERT. (2022). *Overview of China's Internet Security Situation in 2021*. [Online]. Available: https://www.cert.org.cn/

[2] Fortinet. (2018). *Fortinet Predicts Highly Destructive and Self-Learning 'Swarm' Cyberattacks in 2018*. [Online]. Available: https://www.fortinet.com/cn/corporate/about-us/newsroom/press-releases/2017/predicts-self-learning-swarm-cyberattacks-2018

[3] Z. Liu, X. Yin, and Y. Hu, "CPSS LR-DDoS detection and defense in edge computing utilizing DCNN Q-learning," *IEEE Access*, vol. 8, pp. 42120–42130, 2020.

[4] Z. Liu and X. Yin, "LSTM-CGAN: Towards generating low-rate DDoS adversarial samples for blockchain-based wireless network detection models," *IEEE Access*, vol. 9, pp. 22616–22625, 2021.

[5] A. Madane, M.-d. Dilmi, F. Forest, H. Azzag, M. Lebbah, and J. Lacaille, "Transformer-based conditional generative adversarial network for multivariate time series generation," 2022, *arXiv:2210.02089*.

[6] Blacknurse. (2018). *Blacknurse It Can Bring You Down*. [Online]. Available: http://www.blacknurse.dk/

[7] P. Kairouz, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, Jun. 2021.

[8] A. Abeshu and N. Chilamkurti, "Deep learning: The frontier for distributed attack detection in fog-to-things computing," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 169–175, Feb. 2018.

[9] M. Yue, L. Liu, Z. Wu, and M. Wang, "Identifying LDoS attack traffic based on wavelet energy spectrum and combined neural network," *Int. J. Commun. Syst.*, vol. 31, no. 2, Jan. 2018, Art. no. e3449.

[10] K. Hong, Y. Kim, H. Choi, and J. Park, "SDN-assisted slow HTTP DDoS attack defense method," *IEEE Commun. Lett.*, vol. 22, no. 4, pp. 688–691, Apr. 2018.

[11] H. Song and X. Wang, "LDDoS emulation technologies based on lightweight virtualization," *Comput. Eng.*, vol. 46, no. 3, pp. 105–113, 2019.

[12] J. Charlier, A. Singh, G. Ormazabal, R. State, and H. Schulzrinne, "SynGAN: Towards generating synthetic network attacks using GANs," 2019, *arXiv:1908.09899*.

[13] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.
[14] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of Things intrusion detection: Centralized, on-device, or federated learning?" *IEEE Netw.*, vol. 34, no. 6, pp. 310–317, Nov. 2020.
[15] R. Wang, C. Ma, and P. Wu, "An intrusion detection method based on federated learning and convolutional neural network," *Netinfo Secur.*, vol. 20, no. 4, pp. 47–54, 2020.
[16] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5615–5624, Aug. 2021.
[17] W. Sun, S. Guan, P. Wang, and Q. Wu, "A hybrid deep learning model based low-rate DoS attack detection method for software defined network," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 5, May 2022, Art. no. e4443.
[18] M. A. Salahuddin, V. Pourahmadi, H. A. Alameddine, M. F. Bari, and R. Boutaba, "Chronos: DDoS attack detection using time-based autoencoder," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 1, pp. 627–641, Mar. 2022.
[19] X. Zhou, Y. Hu, W. Liang, J. Ma, and Q. Jin, "Variational LSTM enhanced anomaly detection for industrial big data," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3469–3477, May 2021.
[20] R. Zhao, Y. Yin, Y. Shi, and Z. Xue, "Intelligent intrusion detection based on federated learning aided long short-term memory," *Phys. Commun.*, vol. 42, Oct. 2020, Art. no. 101157.
[21] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, Mar. 2020.
[22] B. McMahan, E. Moore, and D. Ramage, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.
[23] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling," *Comput. Netw.*, vol. 121, pp. 25–36, Jul. 2017.
[24] MIT. (1999). *DARPA Intrusion Detection Evaluation Dataset*. [Online]. Available: http://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset
[25] A. Tesfahun and D. L. Bhaskari, "Intrusion detection using random forests classifier with SMOTE and feature reduction," in *Proc. Int. Conf. Cloud Ubiquitous Comput. Emerg. Technol.*, Nov. 2013, pp. 127–132.
[26] N. Pritesh, M. H. Kumar, and T. Manish, "Novel approach of intrusion detection classification deeplearning using SVM," in *Proc. 1st Int. Conf. Sustainable Technol. Comput. Intell.* Singapore, 2020, pp. 365–381.
[27] V. M. Rios, P. R. M. Inácio, and D. Magoni, "Detection of reduction-of-quality DDoS attacks using Fuzzy logic and machine learning algorithms," *Comput. Netw.*, vol. 186, Feb. 2021, Art. no. 107792.
[28] P. Sun, P. Liu, Q. Li, C. Liu, X. Lu, R. Hao, and J. Chen, "DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system," *Secur. Commun. Netw.*, vol. 2020, pp. 1–11, Aug. 2020.
[29] W. Sun, S. Guan, P. Wang, and Q. Wu, "A hybrid deep learning model based low-rate DoS attack detection method for software defined network," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 5, May 2022, Art. no. e4443.
[30] X. Zhou, W. Liang, S. Shimizu, J. Ma, and Q. Jin, "Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5790–5798, Aug. 2020.

ZENGGUANG LIU received the B.S. and M.S. degrees from the University of Shanghai for Science and Technology, in 2005 and 2008, respectively, and the Ph.D. degree from the Shandong University of Science and Technology, in 2022. From 2008 to 2018, he was a Senior Architecture Engineer with the Core Network Department, Alcatel-Lucent. Currently, he is an Assistant Professor with the School of Information Engineering, Shandong Vocational College of Science and Technology, Shandong, China. His research interests include big data, AI, and network security areas.

CUIYUN GUO received the B.S. degree from Shandong Normal University, in 2005, and the M.S. degree from the Ocean University of China, in 2011. She is currently a Lecturer with the School of Computer Science and Technology, Weifang University of Science and Technology, Shandong, China. Her research interests include computer networks and blockchain.

DEYONG LIU received the B.S. degree from Shandong Normal University, in 1996, and the M.S. degree from the Ocean University of China, in 2010. He is currently a Professor with the Weifang University of Science and Technology. He is also the Tutor of Shandong Normal University and the Torch High-Tech Industry Development Center of the Ministry of Science and Technology and the Director of the Shandong Software Industry Association. His research interests include cloud computing, the IoT, and smart city big data research.

XIAOCHUN YIN (Member, IEEE) received the B.S. degree from Qufu Normal University, in 2004, the M.S. degree from Nanjing Normal University, in 2007, and the Ph.D. degree from Dongseo University, in 2015. Since 2007, she has been an Associate Professor with the Weifang University of Science and Technology. Her research interests include network security, the IoT security, blockchain, and AI areas. She was a recipient of the Science Research Innovation Award from the Weifang University of Science and Technology, in 2017 and 2018.