Optimized Lightweight Federated Learning for Botnet Detection in Smart Critical Infrastructure

Segun I. Popoola, Ruth Ande, Aderemi A. Atayero, Mohammad Hammoudeh, *Senior Member, IEEE,* Guan Gui, *Senior Member, IEEE,* Bamidele Adebisi, *Senior Member, IEEE*

***Abstract*—The potentials of federated learning are currently being explored for privacy-preserving intrusion detection in smart critical infrastructure. However, previous works did not consider the determination of optimal model hyperparameters, class imbalance in the training data, high memory space requirement for network traffic data storage in resource-constrained edge nodes, and zero-day botnet attack scenarios. In this paper, we propose an optimized lightweight Federated Deep Learning (FDL) method for botnet attack detection in smart critical infrastructure. First, an optimization method is developed to determine the most appropriate combination of model hyperparameters for local Deep Learning (DL) at the edge nodes. Then, an oversampling algorithm is combined with the optimal DL model to improve the classification performance when the training data is highly imbalanced, without a significant increase in the overall computation time. Furthermore, a feature dimensionality reduction method is used to reduce the amount of memory space required to store the network traffic data at the edge nodes. Experiments are performed using the Bot-IoT and N-BaIoT datasets, and the FDL model achieved high classification performance with low memory space requirement.**

***Index Terms*—cybersecurity, Internet of Things, botnet detection, intrusion detection, deep learning, federated learning**

I. Introduction

THE Internet of Things (IoT) is one of the main technologies of smart critical infrastructure in the fourth industrial revolution (Industry 4.0) [1]. However, IoT has become the primary target of malicious botnet operators due to its proliferation and distributed nature [2]. A malicious botnet is a network of compromised computers known as bots. Hackers use this complex hacking technique to propagate malware and launch cyber attacks against IoT-enabled systems.

In a typical botnet, a cyber attacker, also known as a botmaster, controls the activities of the bots remotely using a Command and Control (C&C) communication channel. The life cycle of a botnet involves five phases, namely initial injection, secondary injection, connection, malicious activities, and maintenance and upgrading [3]. First, the botmaster infects IoT devices with malware. Then, the infected devices download malware binary files from a specific network database using Internet Relay Chat (IRC), File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP) or Peer-to-Peer (P2P) communication protocols. The new bots establish connections with the C&C server to receive instructions and updates. The botmaster instructs the bots to perform malicious activities. Finally, the botmaster maintains its hold on the bots by updating the malware frequently. A P2P botnet has no dedicated C&C server, while a hybrid botnet combines both centralised and P2P architectures. Botnets have significantly widened the attack vulnerability landscape of IoT [4].

Federated Learning (FL) is a privacy-preserving Artificial Intelligence (AI) method, and researchers have started exploring its application to cyber-attack detection in smart critical infrastructure [5], [6]. Although FL methods have been proposed to detect cyber attacks in different application domains, there are still some challenges that need to be addressed, which include the determination of optimal model hyperparameters, low classification performance due to imbalanced sample distribution in the training set, and high memory space requirement for training data storage. In this paper, we propose a Federated Deep Learning (FDL) method for efficient botnet attack detection in smart critical infrastructure. The main contributions of the paper are as follows:

1) A hyperparameter optimization method is used to determine the most appropriate combination of the numbers of hidden layers and hidden units, the learning rate, the optimizer, the activation function, the batch size, and the number of epochs for local Deep Learning (DL) at the IoT edge nodes.
2) A framework, which combines Synthetic Minority Oversampling Technique (SMOTE) with a Bidirectional Long Short-Term Memory (BLSTM) architecture, is proposed to improve the classification performance of the DL-based botnet attack detection models when the network traffic data in the training set is highly imbalanced.
3) A hybrid DL method, which employs LSTM Autoencoder (LAE) and BLSTM architectures, is proposed to reduce the feature dimensionality of the network traffic data without any significant adverse effect on the classification performance. Consequently, the amount of memory space required to store the training data is

Manuscript received October xx, 2020; revised December xx, 2020.

S. I. Popoola and B. Adebisi are with the Department of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom. E-mail: s.popoola@mmu.ac.uk; b.adebisi@mmu.ac.uk

R. Ande is with the Artificial Intelligence for Cybersecurity Research Team, Cyraatek Ltd., Manchester M5 3EZ, United Kingdom. E-mail: ruth@raait.com

A. A. Atayero is with the Department of Electrical and Information Engineering, Covenant University, P.M.B. 1023 Ota, Nigeria. E-mail: atayero@covenantuniversity.edu.ng

M. Hammoudeh is with the Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom. E-mail: m.hammoudeh@mmu.ac.uk

G. Gui is with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, 210003 China. E-mail: guiguan@njupt.edu.cn

reduced.

4) A FDL method, which combines FL and LAE-BLSTM algorithms, is proposed for zero-day botnet attack detection in IoT edge nodes to reduce data transmission cost, reduce network latency, and preserve the privacy of IoT network users.
5) Extensive experiments are performed to evaluate the effectiveness of the DL and FDL methods based on accuracy, precision, recall, F1 score, training time, and testing time using the Bot-IoT and N-BaIoT datasets.

II. Review of Related Works

In this section, we review the most recent FL methods that were proposed for cyber-attack detection in IoT and IoT-enabled critical infrastructure. Table I shows that none of the previous works addressed all the four challenges namely the determination of optimal model hyperparameters, class imbalance in the training data, high memory space requirement for network traffic data storage in resource-constrained edge nodes, and zero-day botnet attack scenarios.

TABLE I
Review of related works

Ref.	Model	Dataset(s)	A[1]	B[2]	C[3]	D[4]
[7]	ANN	NSL-KDD	✗	✗	✗	✗
[8]	CNN-GRU	GPWST	✗	✗	✗	✗
[5]	DNN, RNN, CNN	Bot-IoT, MQTTset, TON_IoT	✗	✗	✗	✗
[9]	CNN	NSL-KDD, UNSW-NB15	✗	✗	✗	✗
[10]	LAE-GRU	ToN_IoT	✗	✗	✓	✗
[11]	ANN	UNSW-NB15	✗	✓	✗	✗
[12]	ANN-RF	MQTT	✗	✗	✗	✗
[13]	CNN	Private	✗	✗	✗	✗
[14]	Transformer	ToN_IoT	✗	✗	✗	✗
[15]	DNN	Edge-IIoTset	✓	✓	✗	✗
[16]	GRU-SVM	KDD-Cup99, CICIDS2017, WSN-DS	✗	✗	✗	✗
[17]	AE-ANN	GPWST	✗	✗	✓	✗
[18]	LR	ToN_IoT	✗	✗	✗	✗
[19]	ANN	Bot-IoT	✗	✗	✓	✗
[20]	GRU-RF	Modbus	✗	✗	✗	✗
[21]	DQN, DNN	CICIDS2017	✗	✗	✗	✗
[22]	DNN	Bot-IoT, NBaIoT	✗	✗	✗	✓
This paper	LAE-BLSTM	Bot-IoT, NBaIoT	✓	✓	✓	✓

[1] Hyperparameter optimization method
[2] Class balance method
[3] Feature dimensionality reduction method
[4] Zero-day IoT botnet attack scenarios

Rahman et al [7] proposed a FL method for intrusion detection in IoT. This method uses Artificial Neural Network (ANN) model architecture, which has a single hidden layer with 288 hidden units, for binary classification. Li et al [8] proposed a FDL method for intrusion detection in industrial Cyber-Physical System (CPS). A hybrid of Convlutional Neural Network (CNN) and Gated Recurrent Unit (GRU) architectures were used for local model training in multiple industrial agents. The CNN model comprised three convolutional blocks, while the GRU model had two hidden layers. The outputs of the two models were concatenated and fed into an Multi-Layer Perceptron (MLP) module, which comprised two hidden layers. Ferrag et al [5] proposed a FDL method for cyber-attack detection in IoT. In this method, Deep Neural Network (DNN), Recurrent Neural Network (RNN), and CNN model architectures were used for local model training. Kumar et al [10] proposed a deep privacy-encoding-based FL framework for data security and privacy in smart agriculture. In this method, a perturbation-based encoding (feature mapping and feature normalisation) and an LAE-based transformation technique were used to prevent inference attacks.

Cheng et al [9] proposed a federated transfer learning method for intrusion detection in mobile edge computing. Transfer learning was employed in FL to speed up the model training, reduce computational cost, increase communication efficiency, and improve classification performance. This involves selecting a well-trained model in a particular source domain and transferring it to the edge server in the target domain. A CNN model architecture, which comprised three convolutional layers, two max-pooling layers, a batch normalisation layer, a dropout layer, and two dense layers, was used for binary classification. Attota et al [12] proposed an ensemble multi-view FL method for intrusion detection in IoT. For each client, three ANN models are trained with the bidirectional flow, unidirectional flow, and packet views of the network traffic features. Grey Wolf Optimization (GWO) technique is used to select the best set of network traffic features for the ANN model training. Chen et al [16] proposed a FL method for intrusion detection in wireless edge networks. This method uses the concept of attention mechanism to calculate the importance of uploaded model parameters, especially when limited bandwidth is available. A combination of GRU and Support Vector Machine (SVM) model architectures were used for local training.

Sedjelmaci and Ansari [11] proposed a cooperative federated Generative Adversarial Network (GAN) for attack detection in multi-access edge computing. The discriminator and generator of the GAN model were designed based on the ANN model architecture, which has five hidden layers. Sun et al [13] proposed segmented FL for adaptive intrusion detection in large-scale local area networks. This method uses a CNN model architecture, which has two convolutional layers, two max-pooling layers, and two dense layers with 200 hidden units each. Abdel-Basset et al [14] proposed a FDL method for security and privacy in heterogeneous blockchain-based smart transportation systems. A stack of context-aware transformer networks, which comprised an encoder and a decoder, was used to learn the spatial-temporal representations of vehicular traffic flows. Aouedi et al [17] proposed a federated semi-supervised learning method for attack detection in industrial IoT. This method uses both labelled and unlabelled data in federated approach since data labelling is costly and time-consuming.

Ruzafa-Alcazar et al [18] evaluated the performance of different differential privacy techniques for FL-based intrusion detection in industrial IoT. The techniques include Laplace, Laplace truncated, Laplace bounded domain, Laplace bounded noise, Gaussian, Gaussian analytic, and uniform. The authors

employed Logistic Regression (LR) model for the classification of network traffic samples. Huong et al [19] proposed an edge-cloud architecture for attack detection. To minimize the complexity of the detection model, Principal Component Analysis (PCA) was employed for feature dimensionality reduction. ANN model, which has a single hidden layer with 6-46 hidden units, was used for multi-class classification. Mothukuri et al [20] proposed a FL method for anomaly detection in IoT. The combination of GRU and RF models was used for local training. The performance of the method was evaluated with the Modbus dataset. Wei et al [21] proposed a FL-based end-edge-cloud cooperative method for attack detection in 5G heterogeneous networks. The authors employed Deep Q-Network (DQN) and DNN for local training in the end nodes and edge nodes, respectively.

Ferrag et al [15] proposed a FL method for cyber-attack detection in industrial IoT. They used the SMOTE method to oversample the minority classes. DNN model, which has two hidden layers with 90 hidden units each, was used for training. A grid search algorithm was used for model hyperparameter optimisation. Popoola et al [22] proposed a FDL method for zero-day botnet attack detection in IoT-edge devices. A DNN model, which has four hidden layers with 100 hidden units each, was used for local training.

III. Proposed Federated Deep Learning Method

The network traffic patterns and the nature of botnet attack that is launched against the IoT edge nodes are usually different. In this study, zero-day botnet attack scenarios are modelled using the 11-class Bot-IoT and the 10-class N-BaIoT datasets.

Table II presents the sample distribution of the zero-day botnet attack traffic data in ten IoT edge nodes based on the Bot-IoT dataset, and Table III presents the sample distribution in nine other IoT edge nodes based on the NBaIoT dataset. A class of botnet attack traffic was not included in each of the IoT edge nodes to model zero-day botnet attack scenario. For example, in Table II, there is no sample of DD-H attack in EN1, and there is no sample of KL attack in EN10. Similarly, in Table III, there is no sample of *g_junk* attack in EN3, and there is no sample of *m_syn* attack in EN7. In order to depict a real-life scenario, the distribution of botnet attack samples was unbalanced and non-identically distributed across the classes of network traffic and across the IoT edge nodes.

FDL method is proposed to detect zero-day botnet attacks in IoT-enabled critical infrastructure based on **Algorithm 1**. The FDL framework comprised of a model parameter server and K IoT edge nodes. The model parameter server coordinates the training of LAE-BLSTM [23], [24] models in the IoT edge nodes. Also, it determines the number of training iterations/epochs (E), the batch size of training data (B), and the number of communication rounds (R). In this method, K LAE-BLSTM models are trained separately with local training data that are privately held in K IoT edge nodes. After each training of E epochs, all the edge IoT devices send their local model updates to the model parameter server for aggregation using FedAvg algorithm [25]. Model aggregation is performed by model parameter server in R communication rounds.

Algorithm 1: FDL algorithm

Input: R, E, N, B, K
Initialization: $W = W_0$
Output: W_r

function `localUpdate`(W, k)**:**
  for $e = 1$ *to* E **do**
    for $b = 1$ *to* $\frac{N}{B}$ **do**
      $W_{k,b} = W_{k,b-1} - \gamma \Delta L(b, W_k)$
    end
  end
  return W_k
end function
for $r = 1$ *to* R **do**
  for $k = 1$ *to* K **do**
    $W_{r,k} = \text{localUpdate}(W_{r-1}, k)$
  end
  $W_r = \sum_{k=1}^{K} \frac{n_k}{N} W_{r,k}$
end

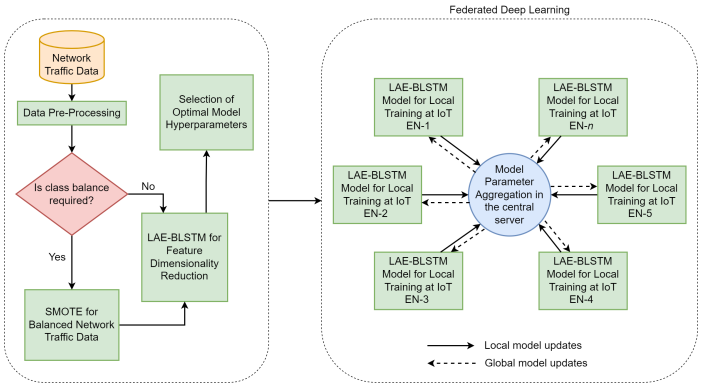

Fig. 1. FDL architecture for zero-day botnet attack detection in IoT edge nodes

The FDL method was simulated with the Bot-IoT and N-BaIoT data sets to evaluate the effectiveness of this method for zero-day botnet attack detection in IoT edge nodes, as shown in Fig. 1. The network traffic data in the IoT edge nodes was pre-processed as earlier described in [26]. The SMOTE method in [27] was used to achieve class balance when the network traffic data is highly imbalanced. The LAE-BLSTM method in [23], [24] was used for feature dimensionality reduction. The optimization method in [26] was used to select the most suitable set of hyperparameters for local model training in the IoT edge nodes. IBM FL framework was used to implement the proposed method. The deployment of the FDL model in IoT edge nodes was simulated using Linux terminals. The communication between the model parameter server and the IoT edge nodes was established using the Flask web framework.

The performance of the FDL method was compared with that of Centralized DL (CDL) and Localized DL (LDL) methods. For the CDL method, each of the IoT edge nodes transmitted its training data to a central server for aggregation.

TABLE II
SAMPLE DISTRIBUTION OF THE ZERO-DAY BOTNET ATTACK TRAFFIC DATA BASED ON THE BOT-IOT DATASET

Class	Edge Nodes									
	EN1	EN2	EN3	EN4	EN5	EN6	EN7	EN8	EN9	EN10
DD-H	0	539	1619	863	1295	215	971	1403	1731	2159
DD-T	117278	0	87958	46911	70367	11727	52775	76231	93827	29319
DD-U	113752	28438	0	45500	68251	11375	51188	73938	91004	85314
D-H	2159	539	1619	0	1295	215	971	1403	1731	863
D-T	73993	18498	55494	29597	0	7399	33296	48095	59198	44395
D-U	123882	30970	92912	49553	74329	0	55747	80523	99110	12388
Norm	2159	539	1619	863	1295	215	971	1403	1079	652
OSF	2159	539	1619	863	1295	215	0	1403	1731	971
SS	8789	2197	6592	3515	5273	878	3955	0	7037	5713
DE	2159	539	1619	863	1295	215	971	1403	0	1731
KL	2159	539	1619	863	1295	215	971	1403	1731	0

TABLE III
SAMPLE DISTRIBUTION OF THE ZERO-DAY BOTNET ATTACK TRAFFIC DATA BASED ON THE N-BAIOT DATASET

Class	Edge Nodes								
	EN1	EN2	EN3	EN4	EN5	EN6	EN7	EN8	EN9
Norm	33339	33339	33339	33339	33339	33339	33339	33339	33339
g_combo	37039	0	33953	33953	33953	33953	33953	33953	33953
g_junk	18832	17263	0	17263	17263	17263	17263	17263	17263
g_scan	18377	16846	16846	0	16846	16846	16846	16846	16846
g_udp	68094	62419	62419	62419	0	62419	62419	62419	62419
m_ack	46435	42566	42566	42566	42566	0	42566	42566	42566
m_scan	0	35511	35511	35511	35511	35511	35511	35511	35511
m_syn	52850	48446	48446	48446	48446	48446	0	48446	48446
m_udp	88577	81196	81196	81196	81196	81196	81196	0	81196
m_udpp	37643	34506	34506	34506	34506	34506	34506	34506	0

Therefore, the CDL model was trained with an aggregated data in the cloud. A copy of the CDL model was sent back to all the IoT edge devices for network traffic classification on the testing data. For the LDL method, model training was performed with the local training data in the edge IoT devices. Therefore, a unique LDL model was developed for each of the IoT edge devices. In the FDL method, a global LAE-BLSTM model was developed and a copy of this model was transmitted to all the IoT edge nodes for network traffic classification. The model parameter server receives further updates from the local models in the IoT edge nodes to improve the classification performance of the global FDL model.

IV. RESULTS AND DISCUSSION

In this section, we evaluate the effectiveness of the FDL models and compare it with that of the CDL and LDL models using the network traffic data in the testing sets of the Bot-IoT and N-BaIoT datasets based on the following: (a) classification performance, (b) computation efficiency, (c) memory efficiency, (d) data privacy preservation, (e) communication cost, and (f) network latency. The sample distribution of the testing sets for the Bot-IoT and N-BaIoT datasets is presented in Table IV.

TABLE IV
SAMPLE DISTRIBUTION OF BOTNET ATTACK TRAFFIC DATA IN THE TESTING SETS

Dataset	Class	No. of samples
Bot-IoT	DD-H	204
	DD-T	195274
	DD-U	190088
	D-H	268
	D-T	122974
	D-U	206789
	Norm	101
	OSF	3582
	SS	14413
	DE	1
	KL	11
N-BaIoT	Norm	111409
	g_combo	103019
	g_junk	52262
	g_scan	50904
	g_udp	189492
	m_ack	128690
	m_scan	107810
	m_syn	146022
	m_udp	246321
	m_udpp	104623

A. Model Hyperparameter optimization

The optimal sets of hyperparameters for the BLSTM models were determined based on the method proposed in [26]. For the Bot-IoT dataset, the optimised BLSTM model employed three hidden layers with 128 hidden units each, a learning rate of 0.001, Nadam optimizer, ReLU activation function, a batch size of 128, and 20 epochs. The optimal model achieved 99.97% accuracy, 85.84% precision, 87.12% recall, and 86.34% F1 score. Nearly all the samples in the DD-T, DD-U, D-T, D-U, Norm, OSF, and SS classes were classified correctly. However, less than 91% of the samples in the DD-H, D-H, DE, and KL classes were classified incorrectly. For instance, the only sample in the DE class was misclassified as a KL attack. This shows that the high class imbalance in the training set adversely affected the classification performance

of the model in the minority classes. It took 214.59 seconds to train the model with the network traffic samples in the training set, and the model spend 1.3 seconds to classify the network traffic samples in the testing set.

For the N-BaIoT dataset, the optimised BLSTM model employed two hidden layers with 128 and 32 hidden units respectively, a learning rate of 0.001, Nadam optimizer, ReLU activation function, a batch size of 512, and 15 epochs. The optimal model achieved 100% accuracy, 99.96% precision, 99.97% recall, and 99.97% F1 score. Nearly all the samples in each of the 10 classes were classified correctly. This implies that the model had a good classification performance. It took 123.41 seconds to train the model with the network traffic samples in the training set, and the model spend 1.23 seconds to classify the network traffic samples in the testing set.

B. Class Balance in the Training Set

Table V presents the sample distribution of the highly imbalanced as well as the balanced network traffic data in the training set of the Bot-IoT dataset. The SMOTE method was used to generate a total of 52139 synthetic samples to increase the low class imbalance ratio of the number of samples in a minority class to the number of samples in the majority class. For example, in the *DE* class, 10791 synthetic samples were generated, and this increased the class imbalance ratio from 1:154854 to 1:57. The increase in the class imbalance ratio helped the BLSTM model to achieve high classification performance. The sampling time for the synthetic data generation was 880 – 930 milliseconds. Therefore, the process did not increase the computation complexity of the DL-based botnet attack detection models. The SMOTE-BLSTM model achieved 100% accuracy, 99.32% precision, 99.92% recall, and 99.61% F1 score. In other words, the precision, recall, and F1 score of the SMOTE-BLSTM model were higher than those of the BLSTM model by 15.7%, 14.69%, and 15.37%, respectively. Nearly all the samples in the eleven classes were classified correctly. This shows that the class balance method improved the classification performance of the DL model, especially in the minority classes. It took 786 seconds to train the model with the network traffic samples in the training set, and the model spend 6.78 seconds to classify the network traffic samples in the testing set.

TABLE V
NEW TRAINING SET FOR THE BOT-IOT DATASET

Class	Training data		
	Original	Generated	New
DD-H	588	10207	10795
DD-T	586393	0	586393
DD-U	568760	0	568760
D-H	906	9889	10795
D-T	369965	0	369965
D-U	619414	0	619414
Norm	290	10505	10795
OSF	10795	0	10795
SS	43949	0	43949
DE	4	10791	10795
KL	48	10747	10795

C. Feature Dimensionality Reduction

The training loss of the LAE-BLSTM model when it was trained with the low-dimensional network traffic feature sets of the Bot-IoT dataset. The training loss reduced as the number of epochs increased from 1 to 15. Specifically, the training loss reduced by 35.92%, 84.02%, 88.68%, 94.02%, and 91.63% when LAE-BLSTM model reduced the feature dimensionality of the network traffic data from 37 to 2, 4, 6, 8, and 10, respectively. The LAE-BLSTM model achieved the lowest training loss of 7.81×10^{-3} when the feature dimensionality of the data was reduced from 37 to 8. Therefore, the LAE-BLSTM model did not under-fit the low-dimensional network traffic data in the 11-class Bot-IoT dataset.

The validation loss of the LAE-BLSTM model when it was trained with the low-dimensional network traffic feature sets of the Bot-IoT dataset. The validation loss reduced as the number of epochs increased from 1 to 15. Specifically, the validation loss reduced by 28.15%, 78.76%, 83.29%, 92.44%, and 87.29% when LAE-BLSTM model reduced the feature dimensionality of the network traffic data from 37 to 2, 4, 6, 8, and 10, respectively. The LAE-BLSTM model achieved the lowest validation loss of 3.79×10^{-3} when the feature dimensionality of the data was reduced from 37 to 8. Therefore, the LAE-BLSTM model did not over-fit the low-dimensional network traffic data in the 11-class Bot-IoT dataset.

Table VI presents the classification performance of the 11-class LAE-BLSTM model based on the balanced Bot-IoT dataset. The BLSTM model, which was developed with the 37-dimensional network traffic data, achieved the best classification performance with 100% accuracy, 99.32% precision, 99.92% recall, and 99.61% F1 score. However, large memory spaces of 666.96 MB, 217.18 MB, and 217.18 MB are required to store the data on a central server or IoT edge node for model training, validation, and testing, respectively. On the other hand, the LAE-BLSTM model, which was developed with 8-dimensional network traffic data, reduced the memory space requirements by 89.19%, without a significant decrease in the classification performance. The model achieved 99.98% accuracy, 99.03% precision, 99.53% recall, and 99.25% F1 score. It took 212.99 ± 0.66 seconds to train the LAE-BLSTM model, and the model spent 1.19 ± 0.01 seconds to classify the network traffic data in the testing set.

TABLE VI
PERFORMANCE OF THE LAE-BLSTM MODEL BASED ON THE ORIGINAL BOT-IOT DATASET

Metrics		Feature Dimensionality					
		2	4	6	8	10	37
(%)	A	97.02	99.80	99.91	99.98	99.96	100.00
	P	71.40	94.31	97.04	99.03	97.80	99.32
	R	83.97	99.04	98.52	99.53	99.44	99.92
	F1	74.93	96.23	97.72	99.25	98.55	99.61
(MB)	Train	18.03	36.05	54.08	72.10	90.13	666.96
	Val	5.87	11.74	17.61	23.48	29.35	217.18
	Test	5.87	11.74	17.61	23.48	29.35	217.18
(s)	Train	1007.39	869.59	945.42	870.72	939.75	786.00
	Test	0.89	0.92	0.90	0.89	0.89	6.78

TABLE VII
PERFORMANCE OF THE LAE-BLSTM MODEL BASED ON THE N-BAIOT DATASET

Metrics		Feature Dimensionality					
		2	4	6	8	10	115
(%)	A	96.30	98.56	99.62	99.90	99.90	100.00
	P	81.94	94.02	98.34	99.50	99.47	99.96
	R	80.96	94.27	98.14	99.45	99.15	99.97
	F1	80.87	94.13	98.23	99.47	99.30	99.97
(MB)	Train	29.77	59.55	89.32	119.09	148.87	3423.92
	Val	9.92	19.85	29.77	39.70	49.62	1141.31
	Test	9.92	19.85	29.77	39.70	49.62	1141.31
(s)	Train	1007.39	869.59	945.42	870.72	939.75	786.00
	Test	0.89	0.92	0.90	0.89	0.89	6.78

Fig. 2 shows the training loss of the LAE-BLSTM model when it was trained with the low-dimensional network traffic feature sets of the N-BaIoT dataset for 10-class classification. The training loss reduced as the number of epochs increased from 1 to 15. Specifically, the training loss reduced by 24.49%, 57.65%, 80.13%, 89.98%, and 91.59% when LAE-BLSTM model reduced the feature dimensionality of the network traffic data from 115 to 2, 4, 6, 8, and 10, respectively. The LAE-BLSTM model achieved the lowest training loss of 2.53×10^{-2} when the feature dimensionality of the data was reduced from 115 to 10. Therefore, the LAE-BLSTM model did not under-fit the low-dimensional network traffic data in the 10-class N-BaIoT dataset.

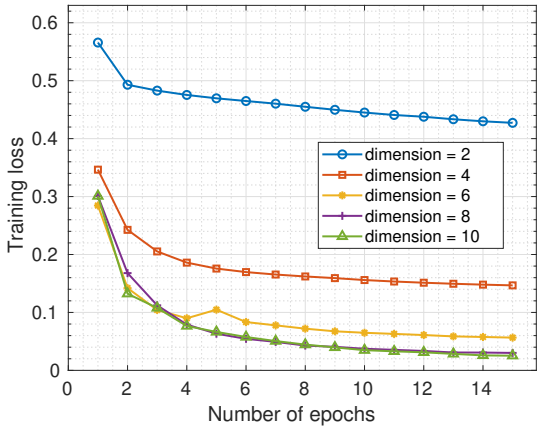

Fig. 2. Training loss of the 10-class LAE-BLSTM model based on the N-BaIoT dataset

Fig. 3 shows the validation loss of the LAE-BLSTM model when it was trained with the low-dimensional network traffic feature sets of the Bot-IoT dataset for 10-class classification. The validation loss reduced as the number of epochs increased from 1 to 15. Specifically, the validation loss reduced by 15.24%, 46.70%, 69.10%, 88.10%, and 89.40% when LAE-BLSTM model reduced the feature dimensionality of the network traffic data from 115 to 2, 4, 6, 8, and 10, respectively. The LAE-BLSTM model achieved the lowest validation loss of 1.79×10^{-2} when the feature dimensionality of the data was reduced from 115 to 10. Therefore, the LAE-BLSTM model did not over-fit the low-dimensional network traffic data in the 10-class N-BaIoT dataset.

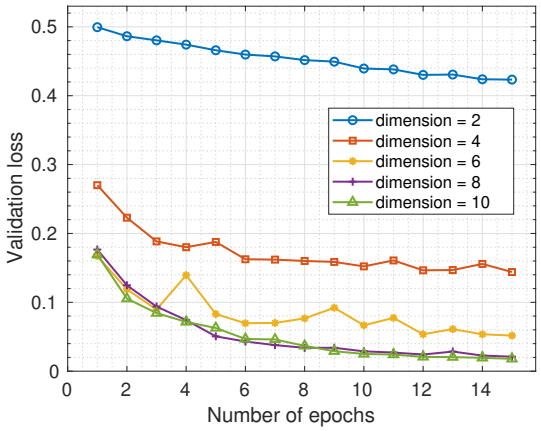

Fig. 3. Validation loss of the 10-class LAE-BLSTM model based on the N-BaIoT dataset

Table VII presents the classification performance of the binary LAE-BLSTM model based on the N-BaIoT dataset. The BLSTM model, which was developed with the 115-dimensional network traffic data, achieved the best classification performance with 100% accuracy, 99.96% precision, 99.97% recall, and 99.97% F1 score. However, large memory spaces of 3.42 GB, 1.14 GB, and 1.14 GB are required to store the data on a central server or IoT edge node for model training, validation, and testing, respectively. On the other hand, the LAE-BLSTM model, which was developed with 8-dimensional network traffic data, reduced the memory space requirements by 96.52%, without a significant decrease in the classification performance. The model achieved 99.90% accuracy, 99.50% precision, 99.45% recall, and 99.47% F1 score. It took 926.57±57.94 seconds to train the LAE-BLSTM model, and the model spent 900 ± 11 milliseconds to classify the network traffic data in the testing set.

D. Localised Deep Learning Models

Ten LDL-based botnet attack detection models, which employed LAE-BLSTM architecture, were trained and tested with the Bot-IoT network traffic data that are located in ten edge nodes (EN1-EN10), respectively.

TABLE VIII
CLASSIFICATION PERFORMANCE OF THE LDL MODELS BASED ON THE BOT-IOT DATASET

Edge Node	Classification performance (%)			
	Accuracy	Precision	Recall	F1 Score
EN1	99.69	78.15	89.89	81.77
EN2	94.27	58.64	84.32	63.13
EN3	94.93	77.84	85.67	80.84
EN4	99.14	72.75	87.88	76.88
EN5	96.77	64.76	85.42	69.61
EN6	94.49	60.57	83.42	66.14
EN7	99.34	67.57	85.87	72.73
EN8	99.10	61.49	85.53	66.19
EN9	99.55	76.39	87.77	80.53
EN10	98.09	69.52	83.38	72.71

Table VIII shows that the LDL models achieved a low classification performance with 97.54 ± 2.23% accuracy, 68.77 ±

7.34% precision, 85.91 ± 2.07% recall, and 73.05 ± 6.77% F1 score. None of the models was able to detect any of the zero-day botnet attacks at the IoT edge nodes. The LDL models, which were developed based on the Bot-IoT dataset, had a faster training time than the CDL models. It took 48.94 ~ 127.88 seconds to train the models with training sets of different sizes, as shown in Table II. However, the LDL models spent more time to classify the network traffic samples in the testing set, compared to the CDL models. The LDL models spent 2.28 ~ 2.54 seconds to classify 733705 network traffic samples in the testing set. The LDL method required a smaller memory space of 2.1 ~ 28.7 MB to store the network traffic data in the IoT edge nodes.

Another nine LDL-based botnet attack detection models were trained and tested with the N-BaIoT network traffic data that are located in nine edge nodes (EN1-EN9), respectively.

TABLE IX
CLASSIFICATION PERFORMANCE OF THE LDL MODELS BASED ON THE N-BAIOT DATASET

Edge Node	Classification performance (%)			
	Accuracy	Precision	Recall	F1 Score
EN1	97.33	81.06	84.03	81.55
EN2	97.61	80.15	86.92	81.86
EN3	98.37	83.31	86.32	84.47
EN4	98.30	81.46	85.16	83.06
EN5	95.58	81.46	82.78	77.65
EN6	97.30	82.57	86.01	83.82
EN7	96.75	76.88	83.74	79.20
EN8	95.59	78.58	86.84	81.23
EN9	97.72	82.62	86.25	83.95

Table IX shows that the LDL models achieved a low classification performance with 97.17 ± 1.03% accuracy, 80.90 ± 2.07% precision, 85.34 ± 1.49% recall, and 81.87 ± 2.28% F1 score. None of the models was able to detect any of the zero-day botnet attacks at the nine IoT edge nodes. The LDL models, which were developed based on the N-BaIoT dataset, had a faster training time than the CDL models. It took 43.47 ~ 53.87 seconds to train the models with training sets of different sizes, as shown in Table III. However, the LDL models spent more time to classify the network traffic samples in the testing set, compared to the CDL models. The LDL models spent 3.77 ~ 4.64 seconds to classify 733705 network traffic samples in the testing set. The LDL method required a smaller memory space of 20.8 ~ 25.7 MB to store the network traffic data in the IoT edge nodes.

In the LDL method, the network traffic features of the IoT edge nodes were not shared with a third-party central cloud server to preserve the data privacy of IoT-enabled critical infrastructure users. The LDL models required a shorter training time and a lower memory space for data storage, and they incurred lower communication overhead. However, the classification performance of the LDL models was lower than that of the CDL models because each of former was trained with insufficient private network traffic and fewer botnet attack scenarios in a single IoT edge node. Therefore, the LDL method is not suitable for zero-day botnet attack detection in IoT-enabled critical infrastructure.

E. Federated Deep Learning Models

A FDL-based botnet attack detection model, which employed LAE-BLSTM architecture, was trained and tested with Bot-IoT network traffic data at ten IoT edge nodes.

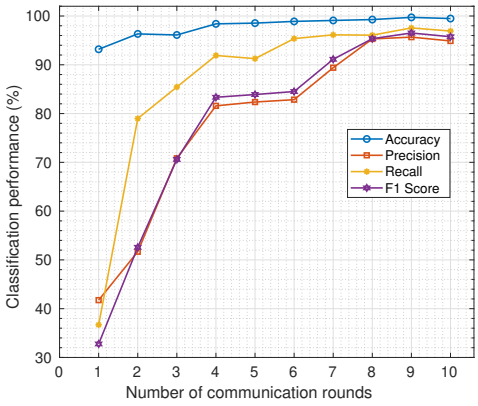

Fig. 4. Classification performance of the FDL model based on the Bot-IoT dataset

Fig. 4 shows that the classification performance of the FDL model improved as the number of communication rounds increased from 1 to 10. Specifically, the accuracy, precision, recall, and F1 score of the model increased by 6.54%, 53.92%, 60.88%, and 63.75%, respectively. The FDL model achieved the best classification performance at the end of the ninth communication round with 99.72% accuracy, 95.67% precision, 97.56% recall, and 96.52% F1 score. All the network traffic samples in the *DD-T*, *DD-U*, *D-U*, *Norm*, *OSF*, *SS*, *DE*, and *KL* classes were classified correctly. This means that the FDL model can distinctively detect benign network traffic as well as DD-T, DD-U, D-U, OSF, SS, DE, and KL attack traffic in IoT-enabled critical infrastructure with 100% accuracy and zero false alarm rate.

Although the FDL model could not classify all the network traffic samples in the *DD-H*, *D-H*, and *D-T* classes correctly, the detection rates were very high and the false alarm rates were very low. In the DD-H class, 90.7% of the samples were classified correctly, 5.9% were misclassified as D-H attack, and 3.4% were misclassified as DD-T attack. In the *D-H* class, 91.8% of the samples were classified correctly, 6.3% were misclassified as DD-H attack, and 1.9% were misclassified as DD-T attack. In the *D-T* class, 91.3% of the samples were classified correctly, and 8.7% were misclassified as DD-T attack. Therefore, the FDL model can also distinctively detect DD-H, DD-U, D-H, OSF, and SS attack traffic in IoT-enabled critical infrastructure with high accuracy and low false alarm rate. The time required to train the FDL model increased from 426 to 3853.64 seconds as the number of communication rounds increased from 1 to 10. The training time of the FDL model which achieved the best classification performance at the end of the ninth communication round was 3491.72 seconds. The FDL model spent 3.7 – 5.3 seconds to classify 733705 network traffic samples in the testing set.

Another FDL-based botnet attack detection model, which employed LAE-BLSTM architecture, was trained and tested

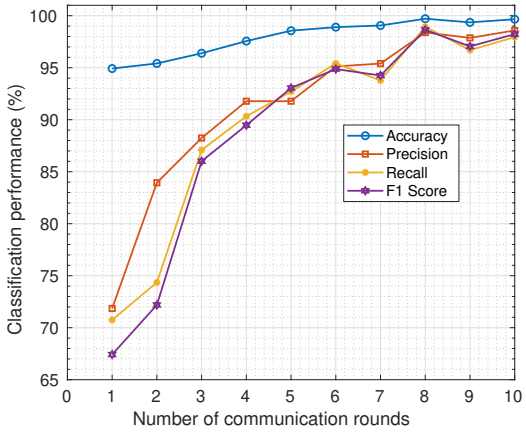

Fig. 5. Classification performance of the FDL model based on the N-BaIoT dataset

with N-BaIoT network traffic data at nine IoT edge nodes. Fig. 5 shows that the classification performance of the FDL model improved as the number of communication rounds increased from 1 to 10. Specifically, the accuracy, precision, recall, and F1 score of the model increased by 4.78%, 26.54%, 28.19%, and 31.22%, respectively. The FDL model achieved the best classification performance at the end of the eighth communication round with 99.71% accuracy, 98.39% precision, 98.94% recall, and 98.64% F1 score. All the network traffic samples in the *Norm*, *g_junk*, *g_scan*, *g_udp*, *m_ack*, *m_scan*, and *m_syn* classes were classified correctly. This means that the FDL model can distinctively detect benign network traffic as well as *g_junk*, *g_scan*, *g_udp*, *m_ack*, *m_scan*, and *m_syn* attack traffic in IoT-enabled critical infrastructure with 100% accuracy and zero false alarm rate.

Although the FDL model could not classify all the network traffic samples in the *g_combo*, *m_udp*, and *m_udpp* classes correctly, the detection rates were very high and the false alarm rates were very low. Therefore, the FDL model can also distinctively detect *g_combo*, *m_udp*, and *m_udpp* attack traffic in IoT-enabled critical infrastructure with high accuracy and low false alarm rate. The time required to train the FDL model increased from 319.95 to 3074.85 seconds as the number of communication rounds increased from 1 to 10. The training time of the FDL model which achieved the best classification performance at the end of the eighth communication round was 2460.92 seconds. The FDL model spent 5.4 – 6.4 seconds to classify 733705 network traffic samples in the testing set.

V. Conclusion

In this paper, an optimised lightweight FL method is proposed for efficient botnet attack detection in smart critical infrastructure. FDL model was developed with the Bot-IoT and N-BaIoT data sets, and its effectiveness was compared with the CDL and LDL models. The optimization method helped in choosing the best combination of the models' hyperparameters for optimal BLSTM model. Considering the highly imbalanced Bot-IoT data set, the proposed oversampling method improved the precision, recall, and F1 score of the BLSTM model. The LAE-BLSTM model reduced the feature dimensionality of the network traffic data without any significant adverse effect on the classification performance. Consequently, the amount of memory space required to store the private training data on local IoT edge nodes also reduced. In the future, efforts will be made to protect the FDL models against adversarial attacks such as backdoor and poisoning attacks.

References

[1] G. Aceto, V. Persico, and A. Pescape, "A survey on information and communication technologies for industry 4.0: state-of-the-art, taxonomies, perspectives, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3467–3501, 2019.

[2] G. Vormayr, T. Zseby, and J. Fabini, "Botnet communication patterns," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2768–2796, 2017.

[3] S. S. Silva, R. M. Silva, R. C. Pinto, and R. M. Salles, "Botnets: A survey," *Computer Networks*, vol. 57, no. 2, pp. 378–403, 2013.

[4] N. Koroniotis, N. Moustafa, and E. Sitnikova, "Forensics and deep learning mechanisms for botnets in internet of things: A survey of challenges and solutions," *IEEE Access*, vol. 7, pp. 61 764–61 785, 2019.

[5] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, "Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis," *IEEE Access*, vol. 9, pp. 138 509–138 542, 2021.

[6] M. Alazab, S. P. RM, M. Parimala, P. Reddy, T. R. Gadekallu, and Q.-V. Pham, "Federated learning for cybersecurity: concepts, challenges and future directions," *IEEE Transactions on Industrial Informatics*, 2021.

[7] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of things intrusion detection: Centralized, on-device, or federated learning?" *IEEE Network*, 2020.

[8] 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 Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615–5624, 2020.

[9] Y. Cheng, J. Lu, D. Niyato, B. Lyu, J. Kang, and S. Zhu, "Federated transfer learning with client selection for intrusion detection in mobile edge computing," *IEEE Communications Letters*, 2022.

[10] P. Kumar, G. P. Gupta, and R. Tripathi, "Pefl: Deep privacy-encoding based federated learning framework for smart agriculture," *IEEE Micro*, 2021.

[11] H. Sedjelmaci and N. Ansari, "On cooperative federated defense to secure multi-access edge computing," *IEEE Consumer Electronics Magazine*, 2022.

[12] D. C. Attota, V. Mothukuri, R. M. Parizi, and S. Pouriyeh, "An ensemble multi-view federated learning intrusion detection for iot," *IEEE Access*, vol. 9, pp. 117 734–117 745, 2021.

[13] Y. Sun, H. Esaki, and H. Ochiai, "Adaptive intrusion detection in the networking of large-scale lans with segmented federated learning," *IEEE Open Journal of the Communications Society*, 2020.

[14] M. Abdel-Basset, N. Moustafa, H. Hawash, I. Razzak, K. M. Sallam, and O. M. Elkomy, "Federated intrusion detection in blockchain-based smart transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[15] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning," *IEEE Access*, 2022.

[16] Z. Chen, N. Lv, P. Liu, Y. Fang, K. Chen, and W. Pan, "Intrusion detection for wireless edge networks based on federated learning," *IEEE Access*, vol. 8, pp. 217 463–217 472, 2020.

[17] O. Aouedi, K. Piamrat, G. Muller, and K. Singh, "Federated semi-supervised learning for attack detection in industrial internet of things," *IEEE Transactions on Industrial Informatics*, 2022.

[18] P. Ruzafa-Alcazar, P. Fernandez-Saura, E. Marmol-Campos, A. Gonzalez-Vidal, J. L. H. Ramos, J. Bernal, and A. F. Skarmeta, "Intrusion detection based on privacy-preserving federated learning for the industrial iot," *IEEE Transactions on Industrial Informatics*, 2021.

[19] T. T. Huong, T. P. Bac, D. M. Long, B. D. Thang, N. T. Binh, T. D. Luong, and T. K. Phuc, "Lockedge: Low-complexity cyberattack detection in iot edge computing," *IEEE Access*, vol. 9, pp. 29 696–29 710, 2021.

[20] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated learning-based anomaly detection for iot security attacks," *IEEE Internet of Things Journal*, 2021.
[21] Y. Wei, S. Zhou, S. Leng, S. Maharjan, and Y. Zhang, "Federated learning empowered end-edge-cloud cooperation for 5g hetnet security," *IEEE Network*, vol. 35, no. 2, pp. 88–94, 2021.
[22] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in iot edge devices," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3930–3944, 2022.
[23] S. I. Popoola, B. Adebisi, R. Ande, M. Hammoudeh, and A. A. Atayero, "Memory-efficient deep learning for botnet attack detection in iot networks," *Electronics*, vol. 10, no. 9, p. 1104, 2021.
[24] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the internet-of-things networks," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4944–4956, 2021.
[25] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.
[26] S. Popoola, B. Adebisi, G. Gui, M. Hammoudeh, H. Gacanin, and D. Dancey, "Optimizing deep learning model hyperparameters for botnet attack detection in iot networks," 2022. [Online]. Available: https://bit.ly/3K6GZWD
[27] S. I. Popoola, B. Adebisi, R. Ande, M. Hammoudeh, K. Anoh, and A. A. Atayero, "Smote-drnn: A deep learning algorithm for botnet detection in the internet-of-things networks," *Sensors*, vol. 21, no. 9, p. 2985, 2021.