Towards Lightweight and Efficient Distributed Intrusion Detection Framework

Shuai Yuan*, Hongwei Li(Corresponding author)*†, Rui Zhang*, Meng Hao*, Yiran Li* and Rongxing Lu‡
* School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
† Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China
‡ Faculty of Computer Science, University of New Brunswick, Canada

GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685953

***Abstract*—Federated learning (FL), as a promising distributed learning paradigm, has put many efforts into distributed intrusion detection systems (IDS), for defending against various malicious attacks, such as SQL injection and DDoS attacks. Compared with traditional IDS based on centralized deep learning (DL), FL-based solutions require not to share users' raw data while yielding better detection performance. However, state-of-the-art FL-based methods still suffer from two key limitations: 1) insufficient detection performance on non-independent and identically distributed (non-IID) data, and 2) high communication and computational overheads due to the utilization of large-scale neural network models. In this paper, we propose a lightweight collaborative intrusion detection framework, called CoLGBM, the first of its kind in the regime of decentralized IDS, where decision tree and light gradient boosting machine (LGBM) are combined for constructing the detection scheme. The main insight is that through combining user-trained decision trees (each user's decision tree is derived from its own data with unique distribution), our framework can perform effectively on non-IID data while working efficiently for handling enormous samples. Compared with the current FL-based methods, our CoLGBM achieves higher accuracy and lower overhead on both IID and non-IID data. Extensive experiment results demonstrate our scheme with high-level performance.**

***Index Terms*—Intrusion Detection, Light Gradient Boosting Machine, Collaborative Learning.**

I. Introduction

Deep learning (DL) has demonstrated tremendous success in constructing intrusion detection systems (IDS) [1] . For example, Wang et al. [2] utilized convolutional neural networks (CNN) to classify and identify malicious traffic by representing raw IP flow data as images. Novaes et al. [3] conducted a network operation detection scheme, in which Long-Short-Term-Memory (LSTM) models are used to predict the abnormal behavior of IP flows. However, centralized DL requires the raw data to be uploaded to the center server for achieving the training of neural networks, so that users' privacy cannot be guaranteed during the process [4]. This severely limits its scope of utilization, especially in scenarios where privacy is concerned.

Recently, federated learning (FL) has attracted widespread attention in both industry and academic [5]. Different from the centralized DL, there is no requirement to share users' original data to the server. Instead, just only a few gradients of the local trained model need to be uploaded. Due to such an advantage, FL has gradually been applied in the intrusion detection domain. For example, Nguyen et al. [6] introduced a self-learning system for detecting compromised devices in IoT networks using Gate Recurrent Unit (GRU) based FL. However, our experimental results found that their method is unable to effectively extend to non-IID scenarios. Abeshu et al. [7] proposed a distributed intrusion detection system based on the DL scheme using a pre-trained stacked autoencoder as feature extraction. Nevertheless, local training involving stacked autoencoder evaluation requires extensive computational resources, which is prohibitively expensive for resource-limited edge devices in a distributed setting. Besides, Diro et al. [8] proposed an LSTM network for distributed cyber-attack detection in fog-to-things communication. However, their scheme can just defend against only two categories of attacks, being noneffective for the problem with multiple malicious attacks. In summary, the research of distributed detection framework is still in its infancy, suffering from two key limitations: 1) insufficient detection performance on non-IID data; 2) high communication and computational overheads. Therefore, it is urgent to design a collaborative detection framework, which can improve detection performance on both IID and non-IID data with minimized communication and computation overhead.

To address the above issues, we propose CoLGBM, a hybrid method combining decision tree and light gradient boosting algorithm. Although the decision tree can effectively mine the relationship between features and labels based on the mechanism of automatical feature combination, using it alone may cause the model overfitting for the data with biased distribution. To solve the problem, we combine each user's trained decision tree, thereby effectively handling the non-IID data. Additionally, we apply the distributed LGBM for further improving the efficiency for processing a large amount of data and features. As a result, compared with state-of-the-art FL-based methods, our CoLGBM can perform better in terms of accuracy on both IID and non-IID data, as well as computation and communication overhead. We summarize our contributions as follows:

978-1-7281-8104-2/21/$31.00 ©2021 IEEE

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 10,2022 at 12:41:25 UTC from IEEE Xplore. Restrictions apply.

- We propose a novel collaborative intrusion detection framework, CoLGBM, which for the first time suggests leveraging the LightGBM to detect multiple malicious attacks.
- Compared with state-of-the-art FL-based methods, CoLGBM can achieve a better performance in terms of accuracy and overhead for both non-IID and IID data.
- We conduct extensive experiments to evaluate the performance of our scheme, where the CICDDoS2019 dataset is utilized for model learning and testing. As a result, our CoLGBM obtains an average 13% accuracy improvement, a 90% reduction in false positive rate and 15× less communication compared with the state-of-the-art FL-based work.

The remaining parts of this paper are organized as follows. In Section II, we state the problem. In Section III, we describe the details of our scheme. In Section IV, we conduct a series of experiments to evaluate the performance. Finally, Section V concludes the paper.

II. Problem Statement

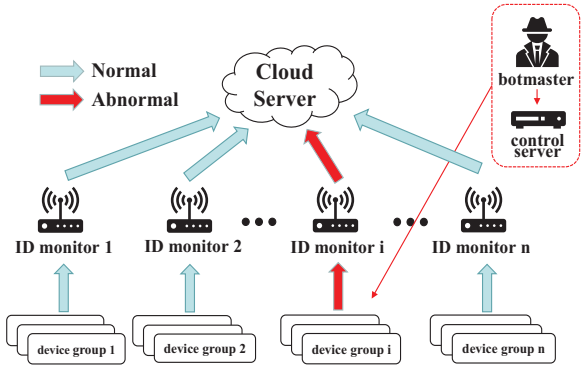


Fig. 1: Client-Server System

In the typical client-server system, as shown in Fig. 1, a botmaster may send malicious commands to the client through the controller, thus corrupting the client and even the cloud server. For guaranteeing the security of the system, it is essential to conduct an Intrusion Detection System (IDS) for detecting malicious intrusions and protecting the system from malware attacks. The main idea of IDS is to detect the malicious traffic (abnormal traffic) in real-time through cloud-based monitoring, thus activating attack warning and traffic interception.

For constructing the Intrusion Detection System, many DL-based methods [9] [10] have been proposed in various dimensions. Recently, FL-based methods [7] [8] [6] have been demonstrated more efficiency and higher accuracy than traditional centralized DL-based works. However, there are still 5 significant challenges faced by many researchers, as follows.

- **Non-IID data:** In the era of big data, the traffic data in the hands of each participant may be non-independent and identically distributed. So the devised framework should be able to deal with the intrusion detection problem on non-IID data.
- **Accuracy:** Compared with traditional centralized learning, the accuracy of the IDS model in distributed scenarios may drop significantly. Thus, the framework should not sacrifice the detection accuracy, no matter the model is trained with IID data or non-IID data.
- **False Positive Rate:** The false positive rate is a very important metric to measure an IDS framework. The high volume of false alarms will bring a heavy burden to IDS service providers, and even lead to their systems being paralyzed. Consequently, the designed framework should preserve a very low false positive rate.
- **Efficiency:** Participants in distributed IDS are often resource-constrained mobile devices or IoT devices. Therefore, the method should not cause excessive computation and communication overheads to the participants.
- **Diversified Deployment:** As technology evolves, the server is no longer the most important subject, and user involvement is often required when it comes to final predictions. Thereby, the new system should support multiple intrusion detection services, including service providers assisted detection and localized detection.

For addressing above issues, in this paper, we propose a novel framework (CoLGBM), which allows the server and separate monitors to jointly achieve the detection task. The details of this technology will be introduced in Section III.

III. Methodology

In this section, we first provide a high-level view of our scheme and then demonstrate an end-to-end intrusion detection framework consisting of training and prediction processes.

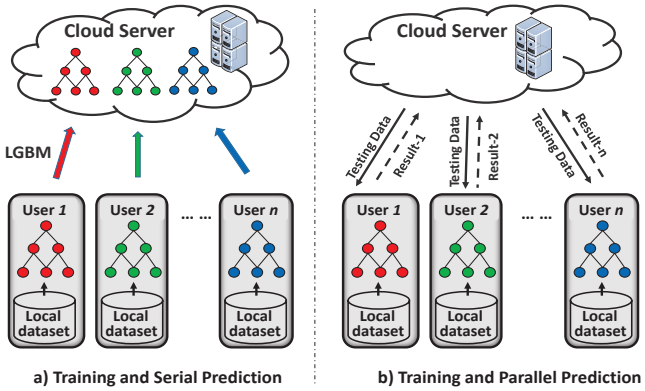


Fig. 2: System Overview

A. Overview of the proposed system

Traditional Federated Learning does not transmit data directly, which protects user privacy to a certain extent. However, it has a large time and communication overhead and cannot achieve the desired results that can be practically applied. Our

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 10,2022 at 12:41:25 UTC from IEEE Xplore. Restrictions apply.

proposed CoLGBM also does not transmit user data directly and contains test datasets in the server just like federated learning. However, our approach differs from existing federated learning methods in both the training and prediction phases. Specifically, in the training phase, we define the LGBM parameters instead of the neural network architecture, and each client trains its local LGBM using local data. Considering that federated learning requires uploading gradient information for each round, our approach does not require uploading any information in the training phase. In the prediction phase, based on efficiency and security considerations, we propose two schemes, as shown in Fig. 2. In this case, clients can send their LGBM models to the server, or the server can send the test set to all clients (this process can be performed simultaneously in the client training phase). Finally, the server obtains the final result by plurality voting.

Because LGBM uses a leaf-wise strategy to generate trees, and DT itself is characterized by fast prediction speed, less feature engineering required, and automatic combination of multiple features, LGBM also uses Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) to deal with the problem of excessive data volume and number of features, so LGBM can achieve higher accuracy with lower time overhead in distributed scenarios. In addition, the tree can be constructed quickly after specifying the parameters, and the training process does not require any data transfer, and only the tree or test dataset is transferred during the prediction phase, so it is more efficient than the traditional FL [11].

B. Training process

In our CoLGBM, we use a new GBDT variant called LightGBM [12], which contains two novel techniques: GOSS and EFB to deal with a large number of data instances and a large number of features, respectively. Specifically, we denote the total amount of data as A_0, the amount of non-zero data as A_1, the total amount of features as S_0, and the number of feature buckets as S_1. Due to estimating information gain of all splits by scanning all the data instances will cost much time, we utilize the GOSS and EFB algorithms for reducing the complexity of histogram building from $O(A_0 \times S_0)$ to $O(A_1 \times S_1)$, s.t. $S_1 << S_0$ [12], which significantly improves the efficiency of the training process. In addition, the LightGBM uses a leaf-wise strategy to grow trees and an additional parameter max_depth to limit the depth of the tree and avoid overfitting.

In the training process, as shown in Fig. 2, each client k has its local data D_k that is non-independent and identically distributed. Each client k firstly sets the training parameters locally and uses the LightGBM framework to train the decision tree T_k on the dataset D_k. Algorithm 1 shows the specific process of training. Here, we explain the symbols in Algorithm 1. There are K clients each with the private dataset $D_k, k \in K$; n denotes the boosting round; B denotes current batch of the test dataset; client k trains a tree model T_k locally and it is used to predict the label l_k of current batch B; the function of $mode()$ is to return the mode of $l_1, l_2 ... l_K$. Subsequently,

Algorithm 1 CoLGBM

procedure SERIAL PROCESS
  for each client k in parallel **do**
    $T_k \leftarrow ClientTrain(k, D_k, n)$
    pass T_k to Server
  end for
  Server executes:
    for $i = 1, 2...K$ **do**
      $l_i \leftarrow T_i(B)$
    end for
    $l \leftarrow mode(l_1, l_2 ... l_K)$
end procedure

procedure PARALLEL PROCESS
  Server executes:
    send B to each client k
  for each client k in parallel **do**
    $T_k \leftarrow ClientTrain(k, D_k, n)$
    $l_k \leftarrow T_k(B)$
    send l_k to Server
  end for
  Server executes:
    $l \leftarrow mode(l_1, l_2 ... l_K)$
end procedure

procedure $ClientTrain(k, D_k, n)$
  setting LGBM parameters
  $T \leftarrow LGB.train(D_k, n)$
  return T
end procedure

in the data preprocessing phase, both our CoLGBM and FL method need to do some feature selection. However, CoLGBM does not require other data preprocessing operations, while FL requires global data standardization or normalization.

C. Prediction process

In this subsection, we introduce two prediction methods for different application scenarios.

1) Serial process: As we discussed in Section III-C, the clients get the final LightGBM models after training on their local data. When requiring to make predictions on the test dataset, the server collects all clients' models and makes a prediction using each model. Finally, the prediction result is generated by plurality voting as shown in procedure *serial process* of Algorithm 1. The advantage of this algorithm is that clients' private original data will not be disclosed, since the IDS service provider cannot immediately recover users' original data according to the rules of decision trees. Moreover, by analyzing the transmitted data size, we found our method had a much lower communication overhead. In the traditional FL-based method, the total transmitted data size is approximately equal to $O(2 \times n \times r \times d_1)$, while the data size our method transmits is $O(n \times d_2)$, where n denotes the

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 10,2022 at 12:41:25 UTC from IEEE Xplore. Restrictions apply.

number of clients, r the round of iteration, d_1 the size of neural network gradients and d_2 the size of the decision tree. (see our experimental results in Section IV). But it also has some flaws. Specifically, when there are too many users, all trees will be stored on the server as shown in Fig. 2 (a), which poses a challenge to the storage capacity of the server. Besides, the server can only calculate the result of each tree serially, which would cost a lot of time.

2) Parallel process: Another method for prediction is that the IDS service provider sends its test dataset to all clients. After that, as shown in procedure *parallel process* of Algorithm 1 and Fig. 2 (b), each client makes prediction on test datasets using its local LightGBM model. Note that the information transmitted through this method still does not reveal the users' original data. All information the server received is only the prediction result from every client. Moreover, this method can be run in parallel, which is why we call it parallel process. In addition, to deal with the problem of some users offline or network delay, the server can set a time threshold T. Once the waiting time exceeds the time threshold T, the server can decide to stop receiving any results. Despite these advantages, this method is more dangerous to reveal IDS service provider's privacy than sending the user's rules of LightGBM models.

IV. Evaluation and results

A. Experimental Setup

We utilize the public dataset called CICDDoS2019 [13] from the University of New Brunswick's Canadian Institute for Cybersecurity. The dataset contains 11 different types of DDoS attack traffic and 1 type of benign traffic. For each type of traffic, we extracted 300,000 pieces of data samples to construct the training dataset. In the test dataset, there are 50,000 pieces of benign traffic and 10,000 pieces of each malicious traffic. For comparing the performance of FL and LightGBM, we make a feature selection. In LightGBM, we use the EFB algorithm [12] to bundle many exclusive features, which significantly decreases the computational cost. In FL, we manually select features to train the neural network model. Similar to [13], we choose the 20 best features for each type of traffic data.

Additionally, we respectively set model parameters for both FL and our CoLGBM, as follows.

- **FL setting.** In FL, we used the Fully Connected (FC) layer and the Gated Recurrent Units (GRU) to construct models FL_FC and FL_GRU, respectively. For the FL_FC model, there have three hidden layers with 512, 256, and 128 neurons, followed by the ReLU activation function. The number of nodes in the output layer is 12, and the activation function is changed to softmax accordingly. For the FL_GRU model, we set a lookback history of $k = 20$ and use a GRU network with three layers of size 256 neurons each.
- **CoLGBM setting.** In CoLGBM method, we set the parameters of LGBM as shown in Table I.

Fianlly, we list all the evaluation metrics, including accuracy, false positive rate (FPR), recall, and F1_score. Note that F1_score calculates the unweighted mean of the metrics for each label, without considering label imbalance.

TABLE I: The parameters of CoLGBM

Parameters	Values
boosting_type	gbdt
objective	multiclass
num_leaves	40
learning_rate	0.05
feature_fraction	0.9
bagging_fraction	0.8
bagging_freq	5
num_class	12

B. Experiment Implementation

1) Constructing data with different distributions: To illustrate the versatility of our method, we simulate a variety of data distributions including IID data and non-IID data, as shown below:

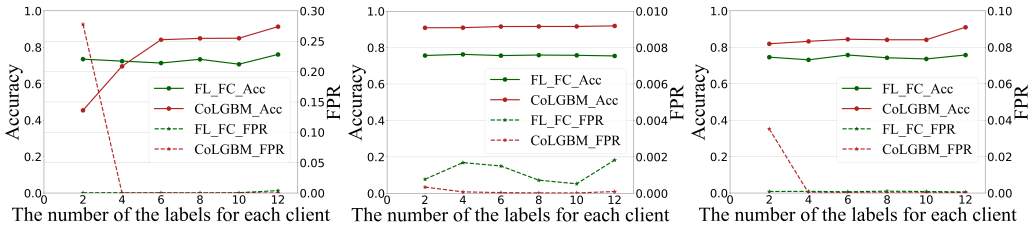
- **Non-IID data.** In this setting, we construct an unbalanced dataset with only a few traffic types for each client. To simulate the real situation, each client has the same amount of data, but different traffic types. In the evaluation, we change the number of traffic types for each client from 2 to 12 to observe the effects of an unbalanced degree on detection performance.
- **Server-aided non-IID data.** This setting is the same as *non-IID data* setting, except for a small public dataset. Specifically, the server constructs the public dataset that contains all types of traffic. After that, clients download the public dataset and combine it with their local datasets (e.g., non-IID data), to generate the combined datasets for training. Note that although the combined dataset of each client contains all types of traffic data, the amount of data for each traffic type is still unbalanced such that it is different from the *IID data* below.
- **IID data.** In this case, we uniformly construct a dataset for each client, which includes all traffic types with the same amount of data. In our experiments, we modify the number of clients to observe the impact on detection performance.

2) Experiments for centralized setting: In order to comprehensively analyze our framework, we first conduct experiments on the centralized scenario, where the training data is collected centrally by the server. We compare our CoLGBM with the FL_GRU method [14] and the FL_FC method that can be seen as an improved version of FL_GRU. In addition, the training dataset of the server contains all traffic types and the amount of data for each type is almost the same. As shown in Table II, our CoLGBM outperforms the other two methods in terms of the accuracy, FPR, Recall, and F1_score in the centralized scenario. Moreover, the performance of FL_GRU is inferior to the improved method FL_FC. Therefore, in the subsequently distributed scenario, we will use FL_FC as the benchmark for experimental comparison.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 10,2022 at 12:41:25 UTC from IEEE Xplore. Restrictions apply.

TABLE II: The experimental results for different methods

Methods	Accuracy	FPR	Recall	F1_score
FL_FC	$74.81 \pm 1.72\%$	$00.34 \pm 0.09\%$	$52.30 \pm 2.13\%$	$64.59 \pm 2.03\%$
FL_GRU	$66.25 \pm 0.47\%$	$12.69 \pm 1.02\%$	$63.66 \pm 1.67\%$	$17.95 \pm 1.35\%$
CoLGBM	$\mathbf{86.27 \pm 1.34\%}$	$\mathbf{00.04 \pm 0.02\%}$	$\mathbf{67.39 \pm 1.79\%}$	$\mathbf{75.09 \pm 2.31\%}$



(a) (b) (c)

Fig. 3: Comparison with different data distributions. (a) Non-IID dataset. (b) Server-aided Non-IID dataset. (c) Client-aided Non-IID dataset.

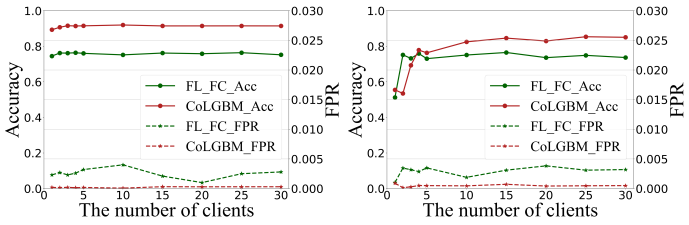
3) Experiments for distributed setting: In the distributed setting, we compare our CoLGBM with the FL_FC method on non-IID and IID data.

a) Comparison on the non-IID data: Fig. 3a shows that the accuracy of FL_FC is stable at about 73.37%, indicating that the stability of FL is good under non-IID data. Moreover, we found that there is strong relation between the accuracy of the CoLGBM and the number of labels owned by clients, i.e., the more labels a client has, the higher the accuracy can be obtained. The accuracy of CoLGBM is close to that of FL_FC in the case where the client data contains 4 types of labels. When the client has more than 4 types of labels, the accuracy of the CoLGBM model is improved by 13.44% on average compared to FL_FC. Finally, when users have all 12 labels, i.e., each user's data is IID, CoLGBM achieves an accuracy rate of 91.37%. Observing the false positive rate (FPR), when the client's label types are more than or equal to 4, the FPR is lower than FL_FC, and it is especially obvious in the case of IID data distribution.

We found that in the case of the non-IID dataset, the accuracy of the CoLGBM model does not perform well with fewer labels. In response to this result, we propose a feasible solution as follows.

b) Comparison on the server-aided non-IID data: In this setting, the clients' dataset consists of local datasets and the public dataset provided by the server. As shown in Fig. 3b, benefit from the public dataset, the accuracy of our CoLGBM is improved by about 15.38% compared with that of the FL_FC model. Compared to the experiment on the non-IID dataset, the accuracy of the CoLGBM model performs better in this setting with fewer types of labels, and the accuracy of the model is relatively stable. Moreover, the FPR of CoLGBM is much lower than that of the FL_FC method in all cases.

In addition, we consider a reasonable alternative when the server is unwilling to disclose the dataset to clients. Specifically, as a special participant, the server participates in the training process with all types of traffic data, called *client-aided non-IID data*. Such a setting is feasible in practical scenarios and easy to implement. In Fig. 3c, the experimental results show that the accuracy of the CoLGBM model is significantly higher than that of the FL_FC model. The results of the FPR are similar to Fig. 3a. The difference is that the FPR of the CoLGBM model decreases with less than 4 tags (although it is still higher compared to the FL_FC model), and the FPR of our scheme is much smaller compared to the FL_FC model when the number of tags is greater than 4.



(a) (b)

Fig. 4: The accuracy with different client numnbers. (a) IID dataset. (b) Non-IID dataset.

c) The impact of the number of clients: We further discuss the impact of the number of clients on the accuracy and FPR of the model. There are two situations for discussion, namely that clients' data is IID and non-IID. In the former case, as shown in Fig. 4a, CoLGBM has an average of 14% higher accuracy than FL_FC. For the FPR, CoLGBM is not only very low but also stable, while on the contrary FL_FC fluctuates more. In the latter case, we set the number of traffic types owned by each client to 6 on the client-aided non-IID data setting. The experimental results are shown in Fig. 4b.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 10,2022 at 12:41:25 UTC from IEEE Xplore. Restrictions apply.

TABLE III: The communication and computation overhead

Methods	Time		Transmitted data size	
	IID Dataset	Non-IID Dataset	IID Dataset	Non-IID Dataset
FL_FC	156.35 ± 3.77s	162.90 ± 4.43s	660.00 ± 1.00MB	660.00 ± 1.00MB
Serial_CoLGBM	208.84 ± 24.84s	192.27 ± 14.47s	$\mathbf{49.45 \pm 0.51}$**MB**	$\mathbf{40.9 \pm 0.30}$**MB**
parallel_CoLGBM	$\mathbf{145.13 \pm 8.33}$**s**	$\mathbf{147.03 \pm 15.80}$**s**	$1487.85 \pm 0..10$MB	1487.85 ± 0.10MB

In terms of accuracy, CoLGBM performs better than FL_FC when the number of clients is greater than 4. Besides, the FPR is always lower than FL_FC. Adding one user does not have much impact on the final result, but gives our solution a better performance compared to FL_FC. Therefore, our solution on both IID and non-IID datasets is feasible.

d) Comparison on communication and computation complexity: We conduct experiments on the communication and time costs of FL_FC serial CoLGBM and parallel CoLGBM. In the experimental setup, 15 users are involved and we set the FL's iteration as 200 epochs. The size of the transmitted test dataset is 97.6 M. As we can see, the time in the second column of Table III is the sum of the time consumed in the training and prediction phases, and the communication cost is the sum of the size of the data transmitted by all client.

The experimental results are shown in Table III. Specifically, the FL_FC takes less time but transmits a larger data size. Our proposed serial_CoLGBM reduces the communication cost by $13\times$-$15\times$ compared to FL_FC. However, the serial_CoLGBM takes a longer time, mainly because it is very time-consuming for the server to make predictions using the received trees in the prediction phase. In parallel_CoLGBM, the time spent is minimal because the server sends the test dataset to each client and the client can make predictions in parallel and pass the prediction results to the server. But the shortcoming of parallel_CoLGBM is that more data is transferred, which depends mainly on the size of the test dataset.

So, the serial_CoLGBM can reduce the transmission cost by $13\times$-$15\times$ to achieve higher accuracy than FL_FC, although it will take some extra time. The parallel_CoLGBM can reduce the training and prediction time compared to FL_FC, and the size of its transmitted data mainly depends on the size of the test dataset.

V. Conclusion and future work

In this paper, we use the distributed decision tree to conduct a lightweight collaborative intrusion detection framework, called CoLGBM. We address two key challenges in existing FL-based intrusion detection methods, e.g., insufficient detection performance on non-IID data and high communication and computational overheads. Extensive evaluations on the CICDDoS2019 dataset show that CoLGBM outperforms prior FL-based works in terms of accuracy, FPR on both IID and non-IID data. In addition, we propose two schemes to reduce the communication overhead and computation overhead respectively. In the future, we will investigate our CoLGBM's performance on different datasets and compare it with other neural network models. We also want to design a lightweight secure protocol tailored to decision tree evaluation.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grants 62020106013, 61972454, 61802051, 61772121, and 61728102, Sichuan Science and Technology Program under Grants 2020JDTD0007 and 2020YFG0298, Chongqing Science and Technology Commission under Grants cstc2018jcyjAX0703, the Fundamental Research Funds for Chinese Central Universities under Grant ZYGX2020ZB027.

References

[1] K. Wolsing, E. Wagner, and M. Henze, "Facilitating protocol-independent industrial intrusion detection systems," in *Proceedings of CCS*, 2020, p. 21052107.

[2] Y. Wang, J. An, and W. Huang, "Using cnn-based representation learning method for malicious traffic identification," in *Proceedings of ICIS*. IEEE, 2018, pp. 400–404.

[3] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proença, "Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment," *IEEE Access*, vol. 8, pp. 83 765–83 781, 2020.

[4] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2020.

[5] Y. Li, H. Li, G. Xu, T. Xiang, X. Huang, and R. Lu, "Toward secure and privacy-preserving distributed deep learning in fog-cloud computing," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 460–11 472, 2020.

[6] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A. Sadeghi, "Dot: A federated self-learning anomaly detection system for iot," in *Proceedings of ICDCS*, 2019, pp. 756–767.

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

[8] A. Diro and N. Chilamkurti, "Leveraging lstm networks for attack detection in fog-to-things communications," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 124–130, 2018.

[9] A. E. Cil, K. Yildiz, and A. Buldu, "Detection of ddos attacks with feed forward based deep neural network model," *Expert Systems with Applications*, p. 114520, 2020.

[10] M. S. Elsayed, N. A. Le-Khac, S. Dev, and A. D. Jurcut, "Ddosnet: A deep-learning model for detecting network attacks," in *Proceedings of WoWMoM*, 2020, pp. 391–396.

[11] Y. Li, H. Li, G. Xu, S. Liu, and R. Lu, "Epps: Efficient privacy-preserving scheme in distributed deep learning," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[12] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[13] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (ddos) attack dataset and taxonomy," in *Proceedings of ICCST*, 2019, pp. 1–8.

[14] F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, and G. A. Shah, "Iot dos and ddos attack detection using resnet," *arXiv preprint arXiv:2012.01971*, 2020.

Authorized licensed use limited to: University of Electronic Science and Tech of China. Downloaded on May 10,2022 at 12:41:25 UTC from IEEE Xplore. Restrictions apply.