2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on

A Selective Model Aggregation Approach in Federated Learning for Online Anomaly Detection

1st Yang Qin
The University of Tokyo
Tokyo, Japan
shin@hal.ipc.i.u-tokyo.ac.jp

2nd Hiroki Matsutani
Keio University
Yokohama, Japan
matutani@arc.ics.keio.ac.jp

3rd Masaaki Kondo
The University of Tokyo
Tokyo, Japan
kondo@hal.ipc.i.u-tokyo.ac.jp

***Abstract*—Cloud computing has established a convenient approach for computing offloading, where the data produced by edge devices is gathered and processed in a centralized server. However, it results in critical issues related to latency. Recently, a neural network-based on-device learning approach is proposed, which offers a solution to the latency problem by relocating processing data to edge devices; even so, a single edge device may face insufficient training data to train a high-quality model, because of its limited available processing capabilities and energy resources. To address this issue, we extend the work to a federated learning system which enables the edge devices to exchange their trained parameters and update local models.**

However, in federated learning for anomaly detection, the reliability of local models would be different. For example, a number of trained models are likely to contain the features of anomalous data because of noise corruption or anomaly detection failure. Besides, as the communication protocol amongst edges could be exploited by attackers, the training data or model weights may have potential risks of being poisoned. Therefore, when we design a federated training algorithm, we should carefully select the local models that participate in model aggregation. In this work, we leverage an observed dataset to compute prediction errors, so that the unsatisfying local models can be excluded from federated training. Experimental results show that the federated learning approach improves anomaly detection accuracy. Besides, the proposed model aggregation solution achieves obvious improvement compared with the popular Federated Averaging method.

***Index Terms*—on-device learning, anomaly detection, federated learning, model aggregation**

I. Introduction

Anomaly detection is the identification of rare items from the majority of the data, which is applicable in a wide variety of application domains including cybersecurity, safety-critical systems, and enemy activities [1]. Neural network-based technique for anomaly detection has received increased attention in recent years, because of its strong ability to achieve high generalization performance for the given data [2]–[5]. However, the traditional cloud-based neural network approaches require the data to be centralized in a cloud server or data center, which results in critical issues related to latency. Besides, iteratively training a neural network with a large amount of input data, e.g., backpropagation, takes a considerable computation time, which makes it difficult to follow time-series changes in the distribution of normal data (i.e., concept drift). For the reasons above, the concept of pushing machine learning closer to where the data is gathered becomes a central point of modern edge devices [6].

Recently, a neural network-based on-device learning approach, which suggests performing both the training and inference computations on the edge device, is proposed for detecting anomalous data [7]. In this approach, the edge device performs fast sequential learning to process the time-series data at runtime; in addition, it detects anomalies independently without a connection to the server. However, since the training side occurs at the distributed edge devices in this approach, only a limited amount of training data can be processed due to the restricted computing capacity of edge devices. Problems arise if there is insufficient training data, as some normal patterns and trends in data may not be present in the prediction model. Hence, a single edge device would be unable to train a high quality neural network model which has a good detection accuracy for anomalous data.

The federated learning (FL), first proposed by Google [8], is a new paradigm that enables a collaborative training of machine learning models among different clients without sharing their privacy-sensitive data. In this paper, we focus on cross-device FL [9], where the clients are a number of edge devices. The framework of applying a horizontal FL system to on-device anomaly detection is shown in Fig.1. Firstly, the server selects a subset of typically normal data to initialize a shared global model. We call rendezvous between server and edge devices a round. Once a round is established, the server sends the metadata (i.e., model parameters) to each edge device. Then, each edge device performs online anomaly detection and updates its local model based on incoming data. After updating the models, edge devices upload their new model parameters to the server. Finally, the server aggregates the received model parameters to update the global model, and the process repeats. It is noteworthy that, due to resource limitations on edge devices, a light-weight neural network should be constructed to support real-time analysis of input data. The details of the neural network will be presented in Section

978-1-7281-7647-5/20/$31.00 ©2020 IEEE
DOI 10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics50389.2020.00119

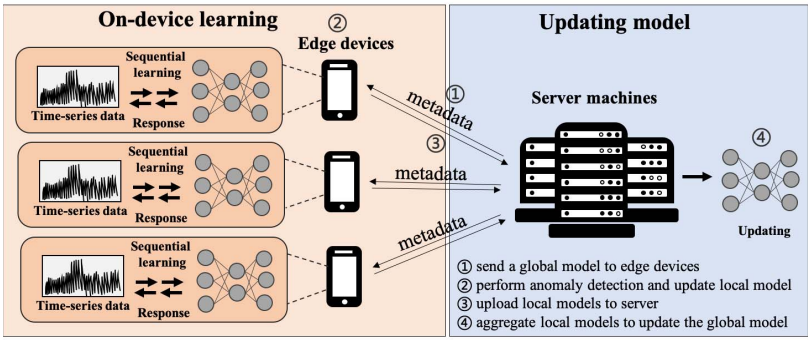


Fig. 1. Framework of cross-device Federated Learning for online anomaly detection.

III.

However, in a cross-device Federated Learning system, the data across each edge device is usually unbalanced and non-IID (identically and independently distributed), which causes large variances among the local models. The Federated Averaging (FedAvg) [10] is a widely-used algorithm for federated model training, which takes the average of either model weight or gradient to train a deep neural network [11], [12]. Recently, the approach is also applied to updating Extreme Learning Machine (ELM) [13] models on edge devices [14], which is similar to our work. To be more precise, the model parameters are combined in the server with FedAvg aggregation, constructing a global model which is then sent back to each edge device for inference. However, it poses challenges to the FedAvg aggregation approach in the following cases.

(1) **The local models aim to perform online anomaly detection.** The local models train the time-series data using sequential learning, and output the prediction results at runtime (normal or abnormal). However, the data is collected on the edge devices, which tend to contain noise during data collection phase. Additionally, there is no perfect anomaly detection approach so that the model parameters are more or less affected by anomalous data.

(2) **The edge devices have potential risks of being attacked.** The server in FL system has no access to the data, nor control of the behaviors for edge devices. As a consequence, the edge devices may intentionally or unintentionally deviate from the prescribed course of federated training, such as being poisoned [15], or even worse turning into a malicious agent [16].

Given the fact that local models are likely to contain anomalous data and have potential risks of being attacked, we should carefully select members that participate in federated training; otherwise, aggregating inaccurate or insecure local models into the global model may inversely reduce the accuracy.

To deal with the underlying challenges, we propose a novel approach that leverages a prepared observed dataset to evaluate the reliability of local models. Specifically, after the server receives all model parameters from the edge devices, a test model is utilized to compute prediction errors (i.e., loss values) based on the observed dataset. In this scheme, for each uploaded model parameter, there is a prediction error that can be calculated. Particularly, the prediction errors are highly dependent on the detection accuracy of local models. For example, a large prediction error implies there is a high probability that the state of the chosen edge device is deviating from the normal states of other devices. Therefore, we should exclude it from model aggregation to avoid its adverse impact on the global model.

The novel contributions of this research are as follows:

(1) **Online anomaly detection in federated learning**: Inspired by on-device learning, we focus on machine learning methods for realizing local anomaly detection on edge devices. However, a single edge device may face insufficient training data to develop a high quality model, because of limited computation and energy resources. In this paper, we make use of a cross-device FL system, which enables all edge devices to exchange their trained results, thereby improving anomaly detection accuracy.

(2) **A selective model aggregation approach**: We addressed several practical problems of applying the FedAvg aggregation approach. To solve the problems, we suggest a selective model aggregation algorithm. The algorithm is designed based on an observed dataset, performing model selection and aggregation to build a higher accuracy global model.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes the background of anomaly detection model and federated learning system. Section IV discusses experimental results. Finally, Section V concludes the paper and presents our future directions.

II. Related Work

In recent years, federated learning has become a hot topic in machine learning for its capability to train a shared model without needing to access user data. Many efforts have recently been devoted to implementing federated learning algorithms to support effective machine learning models, including deep neural networks (NNs) [17], gradient boosted decision trees (GBDTs) [18], logistics regression [19] and support vector machines (SVMs) [20]. However, there is little work that examined how to run

federated learning with a client selection. In this section, we mainly focus on investigating the studies concerned with client selection.

In [21], Bonawitz et al. considered the participating clients in system design, however, they simply chose the devices that meet the eligibility criteria, e.g., charging and connected to an unmetered network. Recent studies related to client selection have specifically focused on defending against poisoning attacks or communication latency. For example, Li et al. suggested a detection-based approach aiming at detecting the anomalous clients (i.e., Byzantine attack), and eliminating their adverse impacts on the trained model [22]. Besides, Nishio and Yonetani discussed the client selection problem in practical mobile edge computing (MEC) frameworks. However, they devoted efforts in managing clients based on their computation and communication resource constraints to accelerate performance improvement in training models [23]. To our knowledge, this is the first work that discusses the issue of client selection in federated learning for improving anomaly detection. Furthermore, we propose a novel method to perform client selection by computing prediction errors based on an observed dataset.

III. Selective Model Aggregation in Federated Learning

In this section, we give detailed information about a selective model aggregation approach in federated learning. We first provide a brief introduction of anomaly detection model on edge devices. Then, we introduce a new aggregation algorithm that aims to exclude unsatisfying local models.

A. On-device Anomaly Detection Model

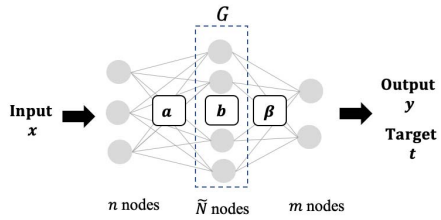
In this work, we take advantage of a single layer feedback neural network called "ONLAD" [7], which suggests a combination of Online Sequential Extreme Learning Machine (OS-ELM) [24] and Autoencoder [25] to perform fast sequential learning for anomaly detection. The details of the model are presented below.

1) OS-ELM: OS-ELM is a sequential implementation of batch learning ELM [13], which is originally developed from a single hidden layer feedforward network (SLFN), as shown in Fig.2-(a). Assuming an n-dimensional input chunk $x \in R^{k\times n}$ is given, where k denotes the batch size; then the m-dimensional output chunk $y \in R^{k\times m}$ with respect to input x can be computed as

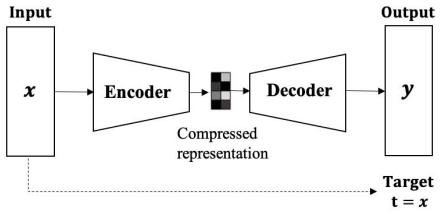
$$y = G(x \cdot \alpha + b)\beta, \tag{1}$$

where $G(\cdot)$ is an activate function, α is an input weight connecting input layer and hidden layer, b is a bias of the hidden node, and β is an output weight connecting the hidden layer and output layer. If SLFN can approximate an m-dimensional target chunk $t \in R^{k\times m}$ with zero error, it means we can find $\hat{\beta}$ that satisfies the following equation

$$G(x \cdot \alpha + b)\hat{\beta} = t. \tag{2}$$



(a) Single Hidden Layer Feedforward Network (SLFN)



(b) Autoencoder

Fig. 2. Basic neural network technologies behind [7].

Let $H \in R^{k\times \tilde{N}}$ be the hidden layer output matrix of the network, i.e., $H = G(x \cdot \alpha + b)$, then the optimal output weight $\hat{\beta}$ is formulated as

$$\hat{\beta} = H^{\dagger}t, \tag{3}$$

where $H^{\dagger}$ is Moore-Penrose generalized inverse of matrix. In particular, if $H^T H$ is non-singular, $\hat{\beta}$ can be further written as $\hat{\beta} = (H^T H)^{-1} H^T t$.

To suit the online learning scenario, OS-ELM which uses sequential learning to update the output weights instead of batch learning is implemented. On the basis, OS-ELM consists of an initialization phase and a sequential learning phase. In the initialization phase, the output weights for an initial training set are determined by

$$\begin{aligned} P_0 &= (H_0^T H_0)^{-1}, \\ \beta_0 &= P_0 H_0^T t_0. \end{aligned} \tag{4}$$

In the sequential learning phase, suppose the i-th training chunk $\{x_i \in R^{k_i\times n}, t_i \in R^{k_i\times m}\}$ is given, the output weights update equations are computed as follows

$$\begin{aligned} P_i &= P_{i-1} - P_{i-1}{H_i}^T(I + H_i P_{i-1}{H_i}^T)^{-1} H_i P_{i-1}, \\ \beta_i &= \beta_{i-1} + P_i {H_i}^T(t_i - H_i\beta_{i-1}). \end{aligned} \tag{5}$$

As seen from Equations (5), OS-ELM recursively updates the output weights on the intermediate results in the last iteration for the new training chunk. The chunk can be discarded immediately as soon as they have been learned without memory-restoring or retraining like ELM.

2) Autoencoder: Autoencoder is a type of artificial neural network which leverages unsupervised learning algorithms to learn a representation (encoding) for an input, as can be seen in Fig.2-(b). It consists of two parts, the encoder and decoder. The encoder tends to compress the input $x \in R^{k\times n}$ into a latent-space representation $\tilde{x} \in R^{k\times \tilde{N}}$, while the decoder takes in the reduced encodings to produce the output $y \in R^{k\times n}$. Generally, the hidden

nodes $\tilde{N}$ is less than input nodes n, i.e., $\tilde{N} < n$, referred to as undercomplete autoencoders. In the training phase, Autoencoder sets the target values to be equal to the inputs, i.e., $t = x$. The goal of Autoencoder is to generate a well-characterized dimensionality-reduced form so that it can reconstruct it as close as possible to its original input.

Autoencoder is implemented with OS-ELM to perform anomaly detection for edge devices, as presented in paper [7]. When it is combined with OS-ELM, the encoding result for input x is represented as $H = G(x \cdot \alpha + b)$, and the decoding result that for decompressing the reduced form is $y = H \cdot \beta$. Thus, when the model is trained with a normal input pattern, the output tends to have a relatively large difference with the input in the case of anomaly.

B. Selective Model Aggregation

In the previous section, we introduced the construction of local model, which combines two classic neural networks to detect anomalous data on edge devices. In this section, we describe a federated training algorithm to support better detection accuracy.

Assuming K edge devices send their local model parameters to the server. Each edge device has a number of n_k training data points. The goal is to detect inaccurate local models (e.g., mixed with a high percentage of anomalous data) and insecure parameters (e.g., under poisoning attacks), to avoid their inverse consequences of the global model.

In the beginning, the server selects a small normal dataset to initialize the input weight α and bias b for the global model. It is notable that once α and b are decided, they will not change during the training process, the same for local models. After deciding the initial parameters, the server computes the output weight β_0 and meta weight P_0 as Equations (4), and distributes them to each edge device for local learning. As long as the edge devices receive the global model, they collect time-series data from the real-world and begin sequential learning. It is because the local models perform unsupervised anomaly detection, that incoming data with large prediction errors are rejected automatically to be trained into the model. More concretely, if input data is identified as abnormal, the device responds to the user that anomalous data is detected; otherwise, the data is fed into the model, that is, updated to model parameters β and P as Equations (5). Furthermore, as frequent participation in federated learning (i.e., after each batch training) may impair the throughput of anomaly detection, we recommend that all edge devices learn locally for a period of time, then upload their model parameters to the cloud server.

The structure of federated learning is shown in Fig.3. After each edge device completes local sequential learning, the model parameters β^k and P^k will be sent to the server. The server collects the model parameters, and performs "parameter selection" and "parameter aggregation" to update the global model. By applying FedAvg algorithm [10], [14], all model parameters will be selected, and the global parameters in the t-th round are updated as the following formula

$$\begin{aligned} P_t &= \sum_{k=1}^{K} \frac{n_k}{N} P_t^k, \\ \beta_t &= \sum_{k=1}^{K} \frac{n_k}{N} \beta_t^k, \end{aligned} \tag{6}$$

where β_t^k and P_t^k represent the local model parameters uploaded from edge k in round t, β_t and P_t represent the global parameters, n_k represents the number of training data points on edge device k, and N represents the total number of data points on all K edge devices. However, as we stated above, averagely aggregating all parameters into the global model may inversely impair the accuracy, as some of β_t^k and P_t^k may contain the features of anomalous/poisoned data. Therefore, we propose a selective parameter aggregation method for federated training.

Algorithm 1 A Selective Parameter Aggregation Approach

Input:
an observed dataset: X;
local model parameters in round t: $\{\beta_t^1, \beta_t^2, ..., \beta_t^K\}$, $\{P_t^1, P_t^2, ..., P_t^K\}$;

Output:
global parameters in round t: β_t, P_t;

for $k = 1$ to K **do**
  $loss_t^k \longleftarrow L(X, G(X \cdot \alpha + b)\beta_t^k)$
end for
for $k = 1$ to K **do**
  if $loss_t^k > \lambda$ **then**
    credit score $c_t^k \longleftarrow 0$
  else
    credit score $c_t^k \longleftarrow \frac{n_k}{N} \frac{1/loss_t^k}{M_t} / \sum_{k=1}^{K} \frac{n_k}{N} \frac{1/loss_t^k}{M_t}$
  end if
end for
$P_t = \sum_{k=1}^{K} c_t^k P_t^k$
$\beta_t = \sum_{k=1}^{K} c_t^k \beta_t^k$

Algorithm 1 gives a general description of our method. We first select a subset of representative normal data as an observed dataset X. Specifically, for each received model parameter, a test ONLAD model can be used to compute

$$loss_t^k = L(X, G(X \cdot \alpha + b)\beta_t^k), \tag{7}$$

where L denotes a loss function, and $loss_t^k$ denotes a prediction error corresponding to β_t^k. Then, we can proceed to compute a credit score by considering the data size and loss values: $c_t^k = \frac{n_k}{N} \frac{1/loss_t^k}{M_t} / \sum_{k=1}^{K} \frac{n_k}{N} \frac{1/loss_t^k}{M_t}$, where M_t represents the sum of reciprocal of the loss values, i.e., $M_t = \sum_{k=1}^{K} 1/loss_t^k$. Note that $\sum_{k=1}^{K} c_t^k = 1$, the credit score c_t^k therefore computes the fraction of each local parameter over the global parameter. Besides, unsatisfying parameters are rejected to be trained to the global model via thresholding. Specifically, if $loss_t^k > \lambda$, we assume the parameters obtained from edge device k are not reliable.

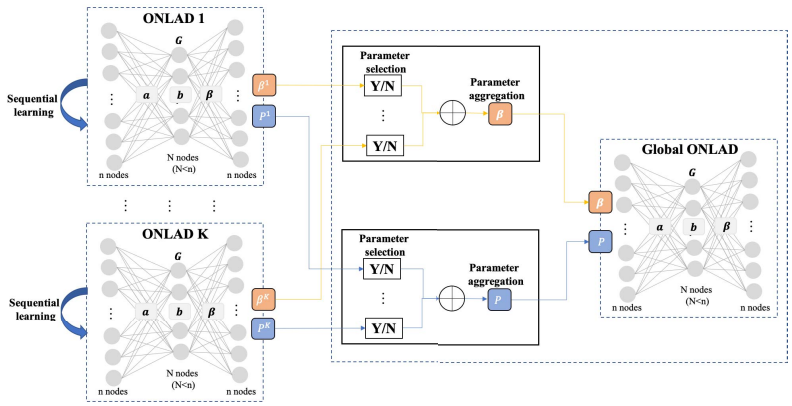


Fig. 3. Structure of federated learning based on ONLAD model.

Thus we should better exclude them from the aggregation operation, i.e., $c_t^k = 0$. Afterwards, the parameters of the global model in the t-th round are updated as follows

$$P_t = \sum_{k=1}^{K} c_t^k P_t^k, \qquad \beta_t = \sum_{k=1}^{K} c_t^k \beta_t^k. \tag{8}$$

IV. Experiment

To demonstrate the efficiency of our proposal, simulation experiments based on MNIST [26] and Fashion MNIST datasets [27] are conducted. Suppose the digits in MNIST are normal, while the images in Fashion MNIST are abnormal. Note that we choose digit datasets as the raw data to simulate an anomaly detection environment, because the datasets include various categories of images that can be used to simulate non-IID data among clients [28]. In this section, we present the experimental setup, procedures, and results of the proposed approaches.

A. Experimental settings

To simulate a federated learning scenario, we assume five edge devices. Each edge is assigned a separate dataset, in which both the number of normal classes and data points are randomly picked. Since we consider online sequential learning for local models, the data arrives in a stream manner. For a comprehensive evaluation, three federated training scenarios with respect to different input datasets are considered.

Scenario 1: all five datasets are normal. This simulates a general case that the server and edge devices collaboratively train/update a shared prediction model. Specifically, all the training data is perfectly normal without any noise or external disturbance. The purpose of this experiment is to measure the generalization capability of our proposed approach.

Scenario 2: some datasets are mixed with anomalies. As this work aims at online anomaly detection, the local models not only perform sequential learning but also reject anomalous data to be trained into the model. However, there is hardly a perfect detection method which achieves 100% accuracy. Moreover, online detection is usually influenced by noise, external environment, and device conditions. As it is difficult to simulate the results of online detection, we simply assume that all the data is fed into local models, while some of them are mixed with anomalies. For simplicity, we assume that four of the datasets are normal, while one of them is a mixture of normal and abnormal data.

Scenario 3: some datasets are injected by poisoned data. Data security has been demonstrated as a big threat to machine learning models. Given that we utilize a federated learning system, the communication protocol amongst different edge devices could be exploited by attackers to launch data poisoning [15]. In this paper, we assume one dataset is poisoned by a Gaussian Attack Model, where each data follows Gaussian Distribution $N(0, 1)$. The purpose of this testbed is to evaluate the robustness of the proposed approach.

Besides the training data, we randomly select 1000 images from MNIST as an observed dataset, which is used for evaluating the reliability of local models. We also prepare a test dataset, where 9000 images are from MNIST as normal data, while another 1000 images are from Fashion MNIST as abnormal data. Afterward, we compare three different model aggregation algorithms: FedAvg [10], [14], "score-based", and "score-based + threshold". For the remainder, we call our proposal "score-based + threshold", as we compute credit scores to determine the dominance of local models, and exclude the unsatisfying models via thresholding. It is worth noting that, in order to illustrate the effectiveness of parameter selection, we present the results of "score-based" separately.

B. Experimental Results

In this section, we describe the experimental results of different federated learning algorithms. All results are averaged over ten trials.

(1) Capability of observed dataset: since the server in a federated learning system has no access to the data on edge devices, it becomes challenging to evaluate the

reliability of local models. To address this issue, we first suggest leveraging an observed dataset to compute their prediction errors (i.e., loss values). To demonstrate the capability of our proposal, in this experiment, we design a dataset mixed with different proportions of abnormality, varying from 0% to 100%. After the dataset is fed into a local model, the model parameters will be computed and sent to the server. We compute loss values of the received parameters based on an observed dataset (see Equation (7)), and draw the bars as Fig.4. From the figure, we observe an obvious increasing tendency of loss value with abnormal data grows, which proves the feasibility of our proposal.

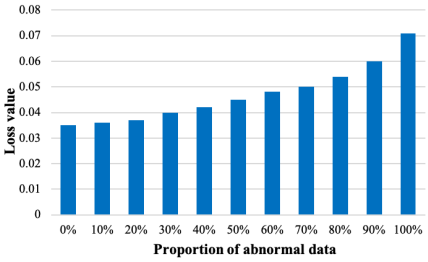
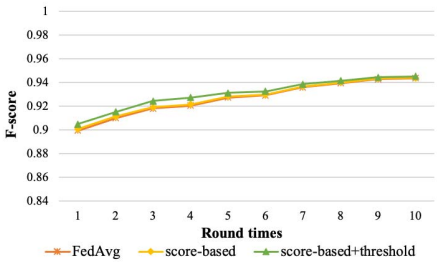


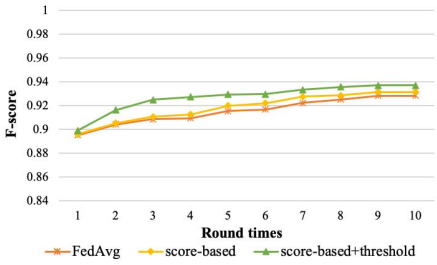
Fig. 4. Loss values of different training data based on an observed dataset.

(2) Effectiveness of model aggregation algorithm: In the second experiment, we compare different model aggregation algorithms. For simplicity, we set the updating time as 1, which represents that all edge devices participate in federated learning for one time. Table 1 shows the overall results obtained using different aggregation methods. We measure $Precision$, $Recall$, $Accuracy$ and $F\text{-}score$, to evaluate the anomaly detection performance of global model. $Precision$ quantifies the number of correctly predicted normal data divided by all normal data. $Recall$ quantifies the number of predicited normal data divided by the total data. $Accuracy$ computes a ratio of correctly predicted data, including both normal and abnormal, to the total data. Finally, $F\text{-}score$ computes the harmonic mean of the $Precision$ and $Recall$, which balances two concerns in one number. It can be observed that our proposal outperforms FedAvg in all three scenarios. The reason for the limited improvement in scenario 1 is that our algorithm is designed for excluding inaccurate/insecure local models, whereas all models in scenario 1 are trained with normal data. Furthermore, we find the "score-based + threshold" always achieves better results than "score-based", which also demonstrates the advantage of parameter selection.

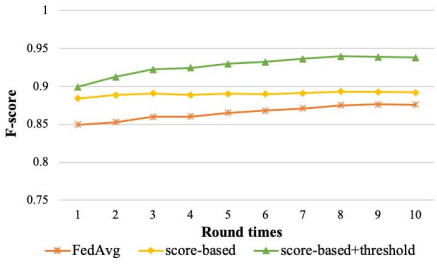
(3) Impact of round times: In the following experiments, we analyze the effectiveness of federated learning. Assume the updating time is 10, and the training data for each edge device is divided into ten small datasets on average. We compare the $F\text{-}score$ of global model after each round of federated learning. In Fig.5, we find that with the increase of round times, the $F\text{-}score$ with respect to all aggregation algorithms in all scenarios shows an increasing tendency. There are two reasons for this: (a) more training data is fed into local models; (b) more exchanges are activated between edge devices and server. It is also observed that the gap between our proposal and FedAvg (green line and red line) varies in three scenarios, minimum in scenario 1, and maximum in scenario 3. This observation implies that when the variances among local models become larger, our method is supposed to achieve better results. Also, the improvement in scenario 1 and scenario 3 certifies the generalization capability and robustness of our approach, respectively.



(a) F-score in Scenario 1



(b) F-score in Scenario 2



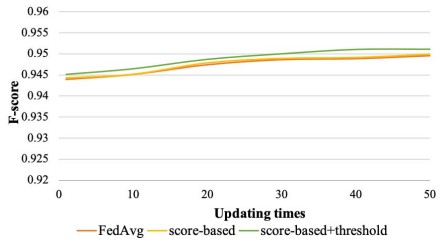
(c) F-score in Scenario 3

Fig. 5. Impact of round times (updating time=10).

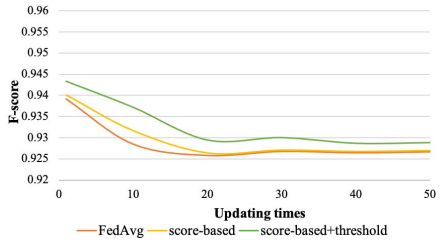
(4) Impact of updating times: In this experiment, different updating times through the whole local data are applied. The more the updating times, the fewer data in each round. Note that we verify the last global model which has successfully completed all rounds' federated learning. The test results are shown in Fig.6. We can observe when the training data is normal without abnormity mixed in or attack, the updating times do not have a large impact on detection precision, as in Fig.6-(a). The slight improvement is because as the updating time increases, the exchange between edge devices and server increases, thus the devices are more capable of cooperatively training a satisfactory model. However, in Fig.6-(b), when the training data is mixed with the abnormality, an unexpected decreasing tendency is first observed, and then stabilizes. We believe the reason for the decreasing tendency is, when

TABLE I
PRECISION OF ANOMALY DETECTION WITH DIFFERENT FEDERATED LEARNING ALGORITHMS

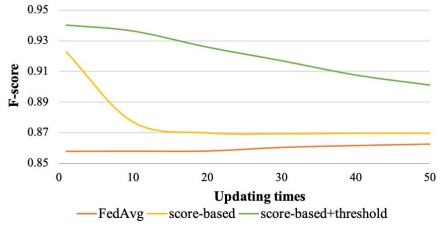
	FL Algorithm	*Precision*	*Recall*	*Accuracy*	*F-score*
Scenario 1	***FedAvg***	0.982	0.911	0.904	0.945
	Score-based	0.982	0.911	0.905	0.945
	Score-based + threshold	**0.982**	**0.912**	**0.905**	**0.946**
Scenario 2	***FedAvg***	0.968	0.908	0.889	0.937
	Score-based	0.971	0.911	0.894	0.94
	Score-based + threshold	**0.977**	**0.912**	**0.9**	**0.943**
Scenario 3	***FedAvg***	0.977	0.791	0.794	0.866
	Score-based	0.98	0.882	0.876	0.928
	Score-based + threshold	**0.982**	**0.91**	**0.9**	**0.942**



(a) F-score in Scenario 1



(b) F-score in Scenario 2



(c) F-score in Scenario 3

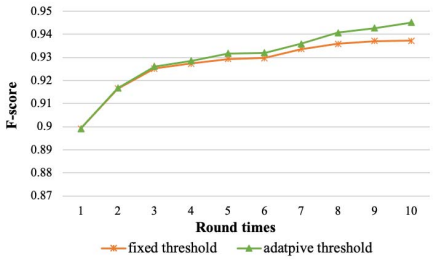
Fig. 6. Impact of updating times.

the exchange between edge devices and server increases, the "normal devices" who perform accurate local detection at first are influenced by "abnormal devices" gradually. Similarly, while the "normal devices" are deteriorating, the "abnormal devices" are under correction. In conclusion, with the updating time increases, all characteristics of local models tend to be similar, which leads to stable detection accuracy. In Fig.6-(c), we find the curve of "score-based + threshold" always shows a decreasing tendency. This is because with the execution of federated learning, the attacked local models become difficult to be detected. The insecure models that failed to be excluded from global aggregation cause a negative impact on the overall accuracy. On the contrary, we observe that the curve of FedAvg shows a slight upward trend. This is because the attack domains the detection accuracy, and the effect of updating times can be ignored. Besides, we believe the reason explained in Fig.6-(a) also leads to the slight increasing tendency in Fig.6-(c).

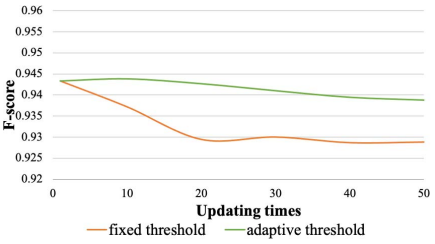
(5) Adaptive threshold method: Inspired by the analysis results in the last experiment, we suggest an adaptive threshold method to help detect "abnormal devices" in case the local models tend to be similar. Specifically, as the federated learning is performed, the threshold should be adaptively decreased. In the current step, we simply assume that the threshold reduces linearly after every execution of federated learning. Take scenario 2 for example. We evaluate the adaptive threshold method with varying round times and updating times, as shown in Fig.7-(a) and Fig7-(b), respectively. Experimental results show that adaptively reducing the threshold helps further improve the detection accuracy. Note that the same conclusions can be drawn in scenario 3. Due to the limited space, the results are not presented in this paper.

V. CONCLUSIONS

Federated learning is originally proposed to enable clients to collaboratively train a machine learning model while preserving client privacy. In this work, we propose a cross-device federated learning system that enables the edge devices to exchange their trained results, thereby improving anomaly detection accuracy. Experimental results show obvious improvement in detection accuracy by using federated learning. Specifically, ten rounds' federated learning achieves an average 4.4% improvement of $F\text{-}score$ than one round. Moreover, we concern the reliability of local models that participate in federated learning, as some of them may contain the features from anomalies data, and have potential risks of being attacked. Therefore, we suggest evaluating all local models and exclude the unsatisfactory ones from the federated training. To our best knowledge, this is the first work that employs an observed dataset to select members that participate in federated learning. Compared with the popular FedAvg aggregation approach, our method shows better generalization



(a) varying round times



(b) varying updating times

Fig. 7. Effectiveness of adaptive threshold method in scenario 2.

performance in anomaly detection, and it is robust against malicious attackers. The approach is supposed to be applied to various anomaly detection applications, for instance, cyber-security, driverless cars, autonomous driving, and avionics domain.

In future work, we will examine if our proposal performs well in large-scale issues. For example, we will evaluate it with Federated Extended MNIST (FEMNIST) dataset [29], which has a collection of 62 different classes distributed unevenly among 3500 users. Besides, we notice the benefit of our proposal relies heavily on the observed dataset. If the normal pattern changes over time, i.e., concept drift happens, it becomes challenging to prepare a representative observed dataset. In the future, we will discuss how to deal with the problem of concept drift in federated learning.

VI. Acknowledgments

This work was supported by JST CREST Grant Number JPMJCR20F2, Japan.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[2] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, "A survey of deep learning-based network anomaly detection," *Cluster Computing*, pp. 1–13, 2017.

[3] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1409–1416.

[4] R. Chalapathy, A. K. Menon, and S. Chawla, "Anomaly detection using one-class neural networks," *arXiv preprint arXiv:1802.06360*, 2018.

[5] S. Garg, K. Kaur, N. Kumar, G. Kaddoum, A. Y. Zomaya, and R. Ranjan, "A hybrid deep learning-based model for anomaly detection in cloud datacenter networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 924–935, 2019.

[6] H. Khelifi, S. Luo, B. Nour, A. Sellami, H. Moungla, S. H. Ahmed, and M. Guizani, "Bringing deep learning at the edge of information-centric internet of things," *IEEE Communications Letters*, vol. 23, no. 1, pp. 52–55, 2018.

[7] M. Tsukada, M. Kondo, and H. Matsutani, "A neural network based on-device learning anomaly detector for edge devices," *arXiv preprint arXiv:1907.10147*, 2019.

[8] F. Learning, "Collaborative machine learning without centralized training data," 2016.

[9] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, and B. He, "A survey on federated learning systems: vision, hype and reality for data privacy and protection," *arXiv preprint arXiv:1907.09693*, 2019.

[10] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[11] H. Yu, S. Yang, and S. Zhu, "Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5693–5700.

[12] T. T. Phuong *et al.*, "Privacy-preserving deep learning via weight transmission," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 11, pp. 3003–3015, 2019.

[13] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)*, vol. 2. IEEE, 2004, pp. 985–990.

[14] Y.-T. Chen, Y.-C. Chuang, and A.-Y. A. Wu, "Online extreme learning machine design for the application of federated learning," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, pp. 188–192.

[15] G. Sun, Y. Cong, J. Dong, Q. Wang, and J. Liu, "Data poisoning attacks on federated machine learning," *arXiv preprint arXiv:2004.10020*, 2020.

[16] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," *arXiv preprint arXiv:1811.12470*, 2018.

[17] Y. Liu, T. Chen, and Q. Yang, "Secure federated transfer learning," *arXiv preprint arXiv:1812.03337*, 2018.

[18] L. Zhao, L. Ni, S. Hu, Y. Chen, P. Zhou, F. Xiao, and L. Wu, "Inprivate digging: Enabling tree-based distributed data mining with differential privacy," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 2087–2095.

[19] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-preserving ridge regression on hundreds of millions of records," in *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013, pp. 334–348.

[20] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

[21] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecny, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[22] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, "Abnormal client behavior detection in federated learning," *arXiv preprint arXiv:1910.09933*, 2019.

[23] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[24] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on neural networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[25] G. Hindon and R. Salakhutdinov, "Reducing the dimensionality of data with neural network," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[26] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," 2010.

[27] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[28] Y. Deng, M. M. Kamani, and M. Mahdavi, "Adaptive personalized federated learning," *arXiv preprint arXiv:2003.13461*, 2020.

[29] S. Caldas, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," *arXiv preprint arXiv:1812.01097*, 2018.