Efficiently Detecting Anomalies in IoT: A Novel Multi-Task Federated Learning Method

Junfeng Hao[1], Juan Chen[1], Peng Chen[1(✉)], Yang Wang[1], Xianhua Niu[1], Lei Xu[2], and Yunni Xia[3(✉)]

[1] School of Computer and Software Engineering, Xihua University, Chengdu, China
chenpeng@mail.xhu.edu.cn
[2] School of Emergency Management, Xihua University, Chengdu, China
[3] School of Computer Science, Chongqing University, Chongqing, China
xiayunni@hotmail.com

Abstract. With the development of IoT technology, a significant amount of time series data is continuously generated, and anomaly detection of this data is crucial. However, time series data in IoT is dynamic and heterogeneous, and most centralized learning also suffers from security and privacy issues. To address these issues, we propose a multi-task anomaly detection approach based on federated learning (MTAD-FL) to address these problems. First, we propose a distributed framework based on Multi-Task Federated Learning (MT-FL), which aims to solve multiple tasks simultaneously while exploiting similarities and differences between tasks; second, to identify complex anomaly patterns and features in the IoT environment, we construct a Squeeze Excitation (SE) based and External Attention (EA) based Enhance Dual Network (SE-EA-EDN) feature extractor to monitor real-time data features from IoT systems efficiently; finally, we design a Local-Global Feature-based Parallel Knowledge Transfer (LGF-PKT) to parallelize the updating of weights of local and global features. To validate the effectiveness of our approach, we conducted comparative experiments on three publicly available datasets, SMD, SWaT, and SKAB, and MTAD-FL improved F1 by 11%, 67.8%, and 27.5%, respectively, over the other methods.

Keywords: Internet of Things · Multi-Task Federated Learning · Anomaly Detection · Feature Extractor · Knowledge Transfer

1 Introduction

With the development of Internet of Things (IoT) technology, an increasing number of devices and sensors are being deployed, leading to a massive influx

This research is supported by the National Natural Science Foundation under Grant No. 62376043 and Science and Technology Program of Sichuan Province under Grant No. 2020JDRC0067, No. 2023JDRC0087, and No. 24NSFTD0025.

H. Gao et al. (Eds.): CollaborateCom 2023, LNICST 563, pp. 100–117, 2024.
https://doi.org/10.1007/978-3-031-54531-3_6

of heterogeneous time series data, and the urgency for extracting critical information from such data has become imperative [1]. Analyzing the quantitative performance plays an important role in understanding and improving the quality of cloud computing systems and cloud-based applications [2,7,9,10]. Anomaly Detection is a data analysis method used to identify behaviors or events that deviate from the expected pattern within the data. It has wide applications in many real-world domains, such as industrial manufacturing, network security, and financial fraud detection. Enhancing Quality of Service (QoS) in IoT cloud environments through anomaly detection techniques. However, the complexity and heterogeneity of multi-dimensional time series data pose challenges for traditional anomaly detection methods.

Federated Learning (FL) [3] effectively utilizes distributed resources to train machine learning models collaboratively. FL is a distributed machine learning approach where multiple edge devices co-train a model while keeping the original data dispersed and not moved to a single server or data center. FL efficiently trains a machine learning model using distributed resources and promises secure and privacy-preserving access to dispersed original data. In Federated Learning, the original data or data generated from the original data after secure handling is used as the training data. Federated Learning only allows intermediate data to be transmitted between distributed computing resources, avoiding data transmission of training data. Distributed computing resources refer to the mobile devices of terminal edge devices or servers of multiple organizations. Federated Learning brings the code to the data instead of bringing data to the code, resolving fundamental issues such as data privacy, ownership, and locality [4]. Therefore, Federated Learning allows multiple edge devices to train a model without leaking personal data.

Time series anomaly detection in IoT is to detect anomalies in the massive high-dimensional data collected, ensuring that the monitored objects are in a normal state and reducing unnecessary expenses in the future. Quantitative performance analysis is not easy because of the complexity of cloud provisioning control flows and the increasing scale and complexity of real-world cloud infrastructures [5]. There are many challenges in conducting time series anomaly detection in the IoT, one of which is how to jointly model data from different devices and locations. This has led to federated learning becoming a popular research direction to solve this problem [6]. Federated learning enables model training without exposing the raw data and can propagate updates from the global model to local models, thereby improving the accuracy and robustness of the model. Each time series dataset has specific characteristics, such as length and variance, which may be significantly different from other time series datasets, requiring multi-task anomaly detection methods [6]. Anomalous data in time series data is usually rare and overwhelmed by a large number of standard points. Therefore, in multi-task learning, it is generally believed that knowledge sharing between different tasks is helpful in improving the efficiency and accuracy of each task [8].

Therefore, we propose a multi-task anomaly detection method based on federated learning (MTAD-FL) to solve these problems. The approach consists of multiple edge nodes, each corresponding to an IoT edge device; the weight aggregation task is also a distributed node whose role is to process and compute the weights received from each node and then send them back to each edge node for the model update. Due to the highly heterogeneous nature of IoT time-series data, this poses a challenge for monitoring and performance modeling, so we first propose a distributed learning framework based on Multi-task Federated Learning to build anomaly detection models in different environments through the massive amount of data in the IoT environment; to identify complex anomaly patterns and features in the IoT environment To identify complex anomaly patterns and features in the IoT environment, we build a dual network feature extractor (SE-EA-EDN) based on Squeeze excitation (SE) and external attention (EA) to efficiently extract anomaly data features; anomalies in the IoT may have different definitions and diversity, different services may report different types of anomalies, and the definition of anomalies may vary depending on the business logic of the system. Therefore, designing a generic anomaly detection and diagnosis system becomes complex, and we design a parallel knowledge migration framework (LGF-PKT) based on local-global features to parallelize the weight update of local and global features.

Our main contributions are summarized as follows:

- To address the highly heterogeneous nature of time-series data in IoT systems, we propose a distributed learning framework based on Multi-task Federated Learning (MFL) to construct anomaly detection models in different environments through massive data in IoT environments;
- To identify complex anomaly patterns and features under IoT systems, we construct a dual network feature extractor (SE-EA-EDN) based on Squeeze Activation (SE) and External Attention (EA) to efficiently extract anomaly data features;
- To address the diversity of anomalies in the IoT environment. We designed a Local-Global Feature-based Parallel Knowledge Transfer Framework (LGF-PKT) parallelizing the implementation of local and global feature weight updates.

The rest of the paper is organized as follows. Section 2 briefly describes the algorithms related to time series anomaly detection; Sect. 3 describes our model architecture and the details of each component. Section 4 provides Sect. 4 provides detailed comparison results and experimental analysis. Section 5 summarises the main Sect. 5 summarises the main work of this paper and provides an outlook for future work.

2 Related Work

As mentioned earlier, the essence of anomaly detection in the IoT environment is to use time series anomaly detection techniques to analyze the time series

data of individual system monitoring performance indicators collected by system monitoring. In this chapter, we first introduce classical methods for time series anomaly detection. We then give a brief overview of deep learning-based methods and models. Finally, we briefly describe federated learning-based methods.

2.1 Classical Methods

The general idea of the statistical-based anomaly detection algorithm is to determine a reasonable range of fluctuations in the current data from the data distribution over a historical period. The model assumes that the statistical model generates ordinary data objects and that data that do not fit the model are outliers. However, the validity of the statistical method is very dependent on the validity of the statistical model assumptions for the given data. The most common methods for detecting time series anomalies using statistical methods are based on the K-sigma algorithm and the ARIMA [11] moving average autoregressive data prediction model. The basic idea of clustering-based anomaly detection algorithms determines whether the current data is anomalous by classifying the data, such as OCSVM [12]. However, there is a significant difference between clustering and anomaly detection, as the goal of anomaly detection is to find abnormal data, while the goal of clustering is to determine the class to which the data belongs. PCA [13], or principal component analysis, is a technique that aims to use the idea of dimensionality reduction to eliminate redundant features from high-dimensional data and retain good features. The deviation of each data point from the rest of the data. In 2014, Twitter also released a seasonal anomaly detection method using a seasonal mixed extreme research bias test (S-H-ESD) [14]. This method is also a practical method based on a robust statistical method.

2.2 Deep Learning Methods

Classical methods cannot meet the requirements of complex, dynamic cloud computing systems, and deep learning approaches benefit from the powerful learning capabilities of neural networks. VAE [15] is an unsupervised anomaly detection algorithm where the algorithm determines anomalous data by reconstructing the error. DeepSVDD [16] learns a spherical boundary by mapping the data into a spherical hyperspace and using support vector machines to separate average data from anomalous data. CGNN-MHSA-AR [17] is a new method for detecting performance anomalies in fluctuating cloud environments that uses an interpretable approach based on neural graph networks (GNNs) and correlation analysis. HTA-GAN [18] is a predictive model based on generative adversarial networks (GANs) that can effectively detect operational anomalies in large-scale IoT. USAD [19] uses an auto-encoder with two decoders and an adversarial game-like training framework to classify normal and abnormal data. CausalRCA [20] enables fine-grained, automated, and real-time root cause localization. TranAD [21] is an anomaly detection and diagnosis model based on deep Transformer networks that use an attention-based sequence encoder to quickly infer information about temporal trends.GDN [22] combines structural learning methods

with neural graph networks, using attention weights to explain detected anomalies. MSCRED [23] is a multi-scale convolutional recursive encoder-decoder for anomaly detection and diagnosis on multivariate time series data. ELBD [24] is a new framework based on integrated learning for robust and accurate performance anomaly detection and prediction. The framework combines machine learning algorithms and models to improve detection and prediction performance.

These detection algorithms implement multivariate anomaly detection through advanced deep-learning methods to improve detection accuracy. However, almost all of these algorithms are centralized single-task anomaly detection, and the training process for single-task centralized methods is bandwidth intensive and has significant privacy implications. The large amount of data generated in IoT systems at any given time would be catastrophic in the event of a data breach.

2.3 Federated Learning Methods

Recently, federated learning has emerged as a viable and compelling alternative to centralized learning methods. Rather than aggregating an increasing amount and type of data into a central location, federated learning distributes the global model training process so that the data from each participating distributed node can be used in situ to train local models [25]. DÏoT [26] is an anomaly detection system that uses federated learning to detect compromised IoT by aggregating anomaly detection profiles for intrusion detection devices. A self-encoder-based anomaly detection method is proposed on the server side [27] for detecting anomalous local weight updates from clients in federated learning systems. PFNM [28] is a probabilistic federated learning framework with a particular emphasis on training and aggregating neural network models by decoupling the learning of local models from their aggregation to global federated models. MT-DNN-FL [6] is a multi-task federated learning approach for anomaly detection in computer networks, traffic recognition, and classification tasks. FATHOM [29] is a federated multi-task hierarchical attention model for activity recognition and environmental monitoring using multiple sensors. Because detecting anomalies in centralized systems is often plagued by significant delays in response times, FSLSTM [25] is a novel privacy design federated learning model using stacked long short-term memory (LSTM) models, which is more than twice as fast during training convergence as centralized LSTMs. It is also essential for model updates in federated learning that FedAvg [30] calculates the average weights of all node models and shares the weights with each node in the federated learning system. The convergence of FedAvg on Non-IID Data (non-linear data) was analyzed by multiple nodes learning a model together [31]. FedTL introduces learning techniques to facilitate knowledge migration between nodes and improve system accuracy. Yang et al. [32] developed FedTL [33], a framework FedSteg for secure image privacy analysis. Unlike FedAvg and FedTL, FedKD takes the average of all node weights as the weights of all teachers and transfers each teacher's knowledge to the corresponding students through Knowledge Distillation (KD) [34]. A population knowledge migration training algorithm was used to train

small convolutional neural networks (CNNs) and transfer their knowledge to a prominent server-side CNN [35]. By finding that existing federated learning methods typically employ a single global model to capture the shared knowledge of all users by aggregating their gradients, regardless of the differences between their data distributions. However, due to the diversity of user behavior, assigning users' gradients to different global models (i.e., centers) can better capture the heterogeneity of data distribution among users, and a multi-center federated learning FeSEM was proposed [36].

The more dynamic and volatile nature of the IoT environment poses challenges for monitoring and performance modeling; and the fact that anomalies in the IoT may have different definitions and diversity; as well as the need for efficient anomaly detection and diagnosis in the IoT environment, and the fact that rapid detection and response to anomalies is critical for system stability and reliability. The above federated learning approach brings us new ideas and directions. Therefore, we propose a Multi-Task Anomaly Detection Based on Federated Learning (MTAD-FL).

3 Method

This chapter first introduces the overall architecture of Multi-Task Anomaly Detection Based on Federated Learning (MTAD-FL), followed by a detailed description of each part.

3.1 Overview

In MTAD-FL, we assume there are t distributed nodes, where $t = 1, 2, ...,$ I_{con}(I_{con} denotes the number of connected distributed nodes), each micro-distributed node has local system monitoring real-time data D_t, D_t is not shared with other nodes, the weight distance calculation is also a distributed node, the detailed system structure is shown in Fig. 1.

First, each running distributed node sends the local system monitoring real-time data as input to a dual network feature extractor (SE-EA-EDN) composed of Local and Transfer models based on Squeeze Excitation and External Attention, where the Local model starts the first round of training. The trained feature weights are sent to the distributed node for weight distance calculation. After all the connected nodes have After all connected nodes have uploaded the feature weights, the weight distance calculation decision is initiated for parallel knowledge migration based on local-global features; after all connected nodes receive the sent-back feature weights, they are loaded onto the Transfer model for the model update, local anomaly detection and diagnosis is performed, and then anomaly categories are output, after which a new round of training is initiated.

3.2 Multi-task Federated Learning Framework(MT-FL)

Suppose we have t tasks, where $t = 1, 2, \ldots, I_{con}$($I_{con}$ denotes the number of connected distributed nodes), (X^t, y^t) denotes the training data for the task,

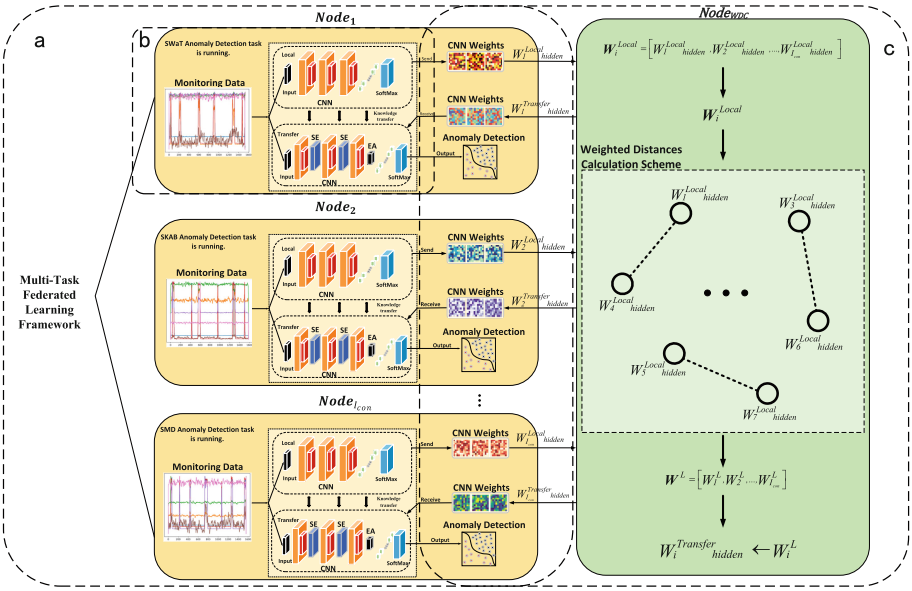

Fig. 1. MTAD-FL overall architecture. (a) denotes the Multi-Task Federated Learning framework; (b) denotes the Double Network Feature Extractor based on Squeeze Excitation and External Attention; (c) denotes the Local-Global Feature-based Parallel Knowledge Transfer.

the local system monitoring system generates X^t in real-time, and y^t denotes the truth label or output vector. Multi-task federated learning aims to minimize the objective function on all nodes to learn the feature weights.

Anomaly detection can be seen as a dichotomous task model where only "normal" and "abnormal" data are judged. For the anomaly detection task, we use the minimized cross-entropy as the loss function, which can be defined as:

$$L\left(y^t, \hat{y}^t\right) = -\frac{1}{m}\sum_{i=1}^{m} y^t log\left(\hat{y}^t\right) \tag{1}$$

where m is the number of input vectors, where are the truth labels and prediction probabilities for the t-th task.

3.3 Dual Network Feature Extractor Based on Equeeze Excitation and External Attention(SE-EA-EDN)

For complex system anomaly patterns and features under IoT, and the fast discovery and response of anomalies is crucial for the stability and reliability of the system, we, therefore, propose a dual network feature extractor. The dual network feature extractor consists of a Local model and a Transfer model deployed on each node; the model is shown in Fig. 2. Firstly, the Local model

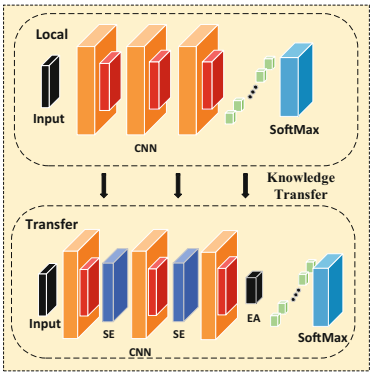

Fig. 2. Structure of a dual network feature extractor based on Squeeze Activation and External Attention.

trains the data of the current node, sends the weights with the characteristics of the node to the weight aggregation node to initiate weight distance calculation for weight update, and then performs parallel knowledge migration to send the matched feature weights back to the Transfer model for further training Squeeze Activation (SE) module and External Attention (EA) module are added to the Transfer model on top of the Local model.

Local Model. The Local model consists of a convolutional layer, an adaptive maximum pooling layer, and a fully connected layer. Each convolutional layer consists of a 1D-CNN module, a batch normalization module, and a ReLU activation function, defined as:

$$f_{conv}(x) = f_{relu}\left(f_{bn}\left(W_{conv} \otimes x + b_{conv}\right)\right) \tag{2}$$

In the formula, W_{conv} and b_{conv} are the weight and bias matrices of the CNN, respectively. $\otimes$ denote the convolution operation. f_{bn} and f_{relu} denote the batch normalization layer and the ReLU activation function, respectively.

Let $X_{bn} = x_1, x_2, ..., x_m$ be denoted as the input to the batch normalization layer, where x_i and m denote the i-th instance and batch size, respectively, defined as :

$$f_{bn}\left(X_{bn}\right) = f_{bn}\left(x_1, x_2, \ldots, x_m\right) = \left(\alpha\frac{x_1-\mu}{\delta-\varepsilon} + \beta, \alpha\frac{x_2-\mu}{\delta-\varepsilon} + \beta, \ldots, \alpha\frac{x_m-\mu}{\delta-\varepsilon} + \beta\right) \tag{3}$$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{4}$$

$$\delta = \sqrt{\sum_{i=1}^{m}\left(x_i - \mu\right)^2} \tag{5}$$

where $\alpha \in R+$ and $\beta \in R$ are the parameters to be learned in training.

Transfer Model. The Transfer model consists of a convolutional layer, an adaptive mean pooling layer, a Squeeze Excitation (SE) module, and an External Attention (EA) module, which enhances the feature extraction capability of the Transfer model by selectively focusing on the contextual features of other external nodes.

Hu et al. [37] proposed a Squeeze Excitation module as a computational unit for arbitrary transformations. $\mathbf{F}_{tr} : \mathbf{X} \rightarrow \mathbf{U}$, where $\mathbf{X} \in \mathcal{R}^{W' \times H' \times C'}$, $\mathbf{U} \in \mathcal{R}^{W \times H \times C}$, and $\mathbf{F}_{tr}$ is represented as $\mathbf{U} = [u_1, u_2, \cdots, u_c]$, where:

$$\mathbf{u}_c = \mathbf{V}_c \otimes \mathbf{X} \tag{6}$$

In the given expression, $\otimes$ denotes a convolution operation.

The squeezing operation utilizes contextual information beyond the local receptive field by using global average pooling to generate channel-wise statistical information. The transformed output $\mathbf{U}$ undergoes contraction along the spatial dimensions $\times H$ to compute the channel-wise statistics $z \in \mathbf{R}^C$. The c-th element of z is calculated by computing $\mathbf{F}_{sq}(\mathbf{u}_C)$, where $\mathbf{F}_{sq}(\mathbf{u}_C)$ is the channel-wise global average value over the spatial dimensions $W \times H$, defined as:

$$\mathbf{Z}_C = \mathbf{F}_{sq}\left(\mathbf{u}_C\right) = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} \mathbf{u}_C\left(i, j\right) \tag{7}$$

The excitation operation follows the aggregated information obtained from the squeeze operation, with the goal of capturing channel dependencies. To achieve this, a simple gating mechanism is applied with the Sigmoid activation, as shown below:

$$s = \mathbf{F}_{ex}\left(\mathbf{z}, \mathbf{W}\right) = \sigma\left(g\left(\mathbf{z}, \mathbf{W}\right)\right) = \sigma\left(W_2 \delta\left(W_1 z\right)\right) \tag{8}$$

where $\mathbf{F}_{ex}$ is parameterized as a neural network, σ is the Sigmoid activation function, δ is the ReLU activation function, $W_1 \in \mathcal{R}^{\frac{C}{r} \times C}$ and $W_2 \in \mathcal{R}^{\frac{C}{r} \times C}$ are the learnable parameters of $\mathbf{F}_{ex}$, and r is the reduction ratio. W_1 and W_2 are used to constrain the complexity of the model and aid in generalization. W_1 is the parameter of the dimensionality reduction layer, while W_2 is the parameter of the dimensionality expansion layer.

Finally, the output of the Squeeze Excitation module is rescaled as follows:

$$\overline{X_c} = F_{scale}\left(u_c, s_c\right) = s_c \bullet u_c \tag{9}$$

In the equation, $\overline{X} = [\overline{x_1}, \overline{x_2}, \ldots, \overline{x_c}]$ and $F_{scale}(u_c,\ s_c)$ represents the multiplication operation of feature map $u_c \in \mathcal{R}^C$ and scale s_c across channels. That is, each channel in u_c is multiplied by the corresponding value in s_c to obtain a new feature map.

External Attention [38] is a mechanism in machine learning models that improves performance on a given task by selectively focusing on certain parts of the input data or features. This mechanism allows the model to focus on relevant

information while ignoring irrelevant or redundant information. The formula for the external attention mechanism can be expressed as follows:

$$A = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{10}$$

In the formula, Q, K, and V respectively represent the Query, Key, and Value matrices, and d_k is the dimension size of the Key matrix.

3.4 Local-Global Feature-Based Parallel Knowledge Transfer(LGF-PKT)

Most existing approaches in federated learning frameworks [25–27,30,32,34] use average weight decisions to aggregate weights and thus update models without considering the differences between data distributions on nodes, which is more evident in the IoT environment, while anomalies in the IoT environment have different definitions and diversity. Therefore, we propose a parallel knowledge migration framework based on local-global features to perform local-to-global model updates using weight distance calculation decisions.

Weighted Distance Calculation Scheme. Let FL_{iter} denote the maximum number of cycles of federated learning. Let$W_i^{Local,k}$ and $W_i^{Transfer,k}$ be the k-th previously trained weight uploaded to the server and the weight sent from the server after weight matching, $k = 1, 2, \ldots, FL_{iter}$. The weights of their hidden layers are denoted by $W_i^{Local_{hidden},k} \subset W_i^{Local,k}, W_i^{Transfer_{hidden},k} \subset W_i^{Transfer,k}$ respectively.

Specifically, $W_i^{Local_{hidden},k}$ consists of Conv1, Conv2 and Conv3, i.e. $W_i^{Local_1,k}$, $W_i^{Local_2,k}$ and $W_i^{Local_3,k}$. Then we have $W_i^{Local_{hidden},k} = W_i^{Local_1,k}$, $W_i^{Local_2,k}, W_i^{Local_3,k}$. In the k-th federated learning phase, nodes $T_i, i = 1, 2, \ldots, I_{con}$ upload $W_i^{Local_{hidden},k}$ to the weight distance calculation node. The node then stores the uploaded weights in the set of weights defined in Eq. (11).

$$\mathbf{W} = \left[W_1^{Local_{hidden},k}, W_2^{Local_{hidden},k}, ..., W_{I_{con}}^{Local_{hidden},k}\right] \tag{11}$$

The server then computes the set of weight distances d, d defined by $\mathbf{W}$:

$$d = \begin{bmatrix} d_1 \\ d_2 \\ \cdots \\ d_{I_{con}} \end{bmatrix} = \begin{bmatrix} d_{1,2} & \cdots & d_{1,I_{con}} \\ d_{2,1} & \cdots & d_{2,I_{con}} \\ \cdots & \cdots & \cdots \\ d_{I_{con},1} & \cdots & d_{I_{con},I_{con}-1} \end{bmatrix} \tag{12}$$

where $d_{i,j}\,(i, j \in 1, ..., I_{con}, i \neq j)$ is the weight distance between $W_i^{Local_{hidden},k}$ and $W_j^{Local_{hidden},k}$ obtained by the combination of Euclidean and Cosine distances, as defined in Eq. (15):

$$d_{i,j}^{eu} = \sqrt{\sum_{n=1}^{3} \left\| W_i^{Local_n,k} - W_j^{Local_n,k} \right\|^2} \tag{13}$$

$$d_{i,j}^{cos} = \frac{\sum_{n=1}^{3} W_i^{Local_{hidden},k} \bullet W_i^{Local_{hidden},k}}{\sqrt{\sum_{n=1}^{3} W_i^{Local_{hidden},k^2}} \bullet \sqrt{\sum_{n=1}^{3} W_i^{Local_{hidden},k^2}}} \tag{14}$$

$$d_{i,j} = \frac{d_{i,j}^{eu}}{d_{i,j}^{cos}} \tag{15}$$

Euclidean distances focus on the differences between feature vectors in each dimension. In contrast, Cosine distances focus on the angles between vectors, and we can better express the similarity between feature vectors by using the two together.

Parallelization of Knowledge Transfer. Parallelization of Knowledge Transfer aims to solve the problem of abnormal data scarcity. Through parallelized knowledge migration, data and knowledge accumulated can be used and migrated to other tasks where data is scarce, thus compensating for the lack of data and improving the model's learning efficiency and generalization ability.

We obtain the **ID** list by weight distance calculation. The **ID** is a ranked list of the most similar feature weights to the current node; the higher the similarity, the highest-ranked index is returned each time, and the most similar feature weights are returned through the index. **ID** is defined in Eq. (16).

$$\mathbf{ID} = [ID_1, ID_2, ..., ID_{I_{con}}] \tag{16}$$

where ID_i is the index of the T_i distance.

According to the ID, it is easy to obtain the set of weights based on the union equation $\mathbf{W}^L$ from $\mathbf{W}$, $\mathbf{W}^L$ is defined in Eq. (17):

$$\mathbf{W}^L = \left[W_1^{L,k}, W_2^{L,k}, ..., W_{I_{con}}^{L,k}\right] = \left[\mathbf{W}\left(ID_1\right), \mathbf{W}\left(ID_2\right), \ldots, \mathbf{W}\left(ID_{I_{con}}\right)\right] \tag{17}$$

where $W_i^{L,k}$ are the weights matching T_i at the k-th federated learning cycle.

Once I_i has received $W_i^{L,k}$ from the server, T_i loads these weights into $W_i^{Transfer_{hidden},k}$ at the beginning of the following federated learning cycle. As defined in Eq. (18).

$$W_i^{Transfer_{hidden},k+1} \leftarrow W_i^{L,k} \tag{18}$$

The feature weights extracted from the Local model are migrated to the Transfer model to perform a local-global parallel weight update. Specifically, we first upload the feature weights from the Local model of all connected nodes to the node where the weight distance is calculated. After uploading the feature weights of all connected nodes, the Weight Distance Computation Scheme is launched, and the matching feature weights are passed back to the nodes for the Local-Global model update, which can be expressed as:

$$W_j^{Transfer} \leftarrow W_i^{Local} \tag{19}$$

4 Experiments and Analysis

Here is a description of three open experimental datasets, as shown in Table 1.

4.1 Dataset

Table 1. Settings and anomaly rates for the three datasets.

Dataset	SWaT	SMD	SKAB
Dimension	51	38	8
Train	7000	70843	12712
Test	3000	30361	5448
Abnormality Rate	29.2%	4.21%	35.1%

- SMD (Server Machine Dataset): It is a public data set for monitoring the performance and operation of servers in the data center. The SMD consists of data from 28 different machines.
- SWaT (Secure Water Treatment): It is a security testing platform for a simulated water treatment plant. Developed by the National University of Singapore, it is used to test and evaluate the network security performance of water treatment plants.
- SKAB (Skoltech Anomaly Benchmark): It consists of a water circulation system, its control system, and a data processing and storage system. The anomalies it generates include a partially closed valve, an unbalanced connecting shaft, re-duced motor power, cavitation, and flow disturbance.

4.2 Evaluation Metrics

For the anomaly detection experiments we used Precision(Pre), Recall(Rec), F1 Score(F1), and Matthews Correlation Coefficient (MCC) as the evaluation metrics.

Precision (Pre) indicates the proportion of samples that are actually abnormal to those that are detected as abnormal and is calculated as:

$$Pre = \frac{TP}{TP + FP} \times 100\% \tag{20}$$

Recall (Rec) represents the proportion of correctly detected anomalies to anomalous samples and is calculated as:

$$Rec = \frac{TP}{TP + FN} \times 100\% \tag{21}$$

F1 Score (F1) is the reconciled average of precision and recall, which is used to evaluate the overall performance of the model:

$$F1 = \frac{2 \times Pre \times Rec}{Pre + Rec} \tag{22}$$

Matthews Correlation Coefficient (MCC) [39] is a correlation coefficient that describes the correlation between the actual classification and the predicted classification, which can take values ranging from, a value of 1 indicates perfect prediction of the subject, a value of 0 indicates that the predicted result is not as good as the result of the random prediction, and –1 means that the predicted classification does not coincide at all with the actual classification:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{23}$$

Where: $TP+FP$ is the number of abnormal samples; $TN+FN$is the number of normal samples; TP is the number of samples correctly detected as abnormal; TN indicates the number of samples correctly detected as normal; FP indicates the number of samples incorrectly detected as abnormal; FN indicates the number of samples incorrectly detected as normal.

4.3 Experiment Setup

Dataset Preprocessing: We performed MinMax normalization on each dataset and compressed it to [0,1].

Parameter Settings: We set the parameters for batch normalization layers and attention mechanisms to their default values and set the decay value of the Squeeze Excitation (SE) module to 16. At the same time, we used AdamW with PyTorch as the optimizer, with an initial learning rate of 0.02, a batch size of 128, and an epoch of 80. The simulation of multi-task learning is implemented on a virtual edge device based on Python3.9.

Baseline Methods: We modified the following three baseline federated learning frameworks to some extent for anomaly detection in the IoT environment. FedTL [33] used a multi-task federated learning framework with the Local model as a pre-trained model, modified to FedTL-AD for multi-task anomaly detection; FedAVG [30] used a multi-task federated learning framework with the Local model as the local model, weight aggregation using weight averaging decisions, modified to FedAVG-AD for multi-task anomaly detection; FedKD [32] used the Local model as the Student model and Teacher model, respectively, and updated the model using weight averaging decisions, modified to FedKD-AD for multi-task anomaly detection. The classical methods are PCA [13] and OCSVM [13] used as anomaly detection; the deep learning methods are VAE [15] and DeepSVDD [16] used as anomaly detection. The specific experimental groupings are as follows:

– Single-Task Anomaly Detection: Comparison of Local and Transfer, PCA, OC-SVM, VAE, DeepSVDD anomaly detection performance on SMD, SWaT, and SKAB using accuracy, recall, F1 score, and MCC as evaluation metrics.
– Multi-Task Anomaly Detection: Comparison of MTAD-FL and FedTL-AD, FedKD-AD, and FedAVG-AD methods for anomaly detection performance on SMD, SWaT, and SKAB using accuracy, recall, F1 score, and MCC as evaluation metrics.

We extracted 10,000 samples from SWaT; for SMD, we chose the first four subsets for the experiment; for SKAB, we used all data under the valve1 file. We took 70% of the samples from the three datasets as the training set and 30% as the test set. For multi-task, the simulation is based on virtual nodes implemented in Pytorch 1.13 and Python 3.9. All experiments were conducted on a PC with an NVIDIA 3070 (8G) graphics card, an Intel(R) Core(TM) i5-12600KF CPU @ 3.69 GHz, and 16 GB RAM.

4.4 Results

Table 2. SWaT, SMD, and SKAB results

Dataset	SWaT			SMD			SKAB		
Metric	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
PCA	0.706	0.747	0.726	0.138	0.829	0.19	0.363	0.986	0.531
OCSVM	0.715	0.766	0.74	0.3	0.401	0.264	0.465	0.713	0.558
VAE	0.706	0.742	0.724	0.04	0.91	0.138	0.363	0.985	0.53
DeepSVDD	0.112	0.115	0.113	0.092	0.615	0.133	0.354	0.974	0.519
Local	0.992	0.59	0.74	0.943	0.73	0.815	0.906	0.787	0.842
Transfer	0.967	**0.864**	**0.913**	**0.999**	**0.997**	**0.998**	**0.923**	0.786	**0.849**
FedTL-AD	0.985	0.619	0.761	0.706	0.230	0.324	0.556	0.254	0.345
FedKD-AD	0.994	0.554	0.711	0.509	0.291	0.343	0.626	0.417	0.569
FedAVG-AD	0.994	0.554	0.711	0.491	0.191	0.238	0.738	0.39	0.454
MTAD-FL	0.902	**0.848**	**0.871**	**0.999**	**0.997**	**0.998**	**0.911**	**0.786**	**0.844**

Two experiments were conducted to demonstrate Single-Task and Multi-Task Anomaly Detection, respectively. We split the experiments into a Single-Task and Multi-Task to demonstrate the robustness and generalization of the two models in SE-AE-EDN for deployment in nodes and a Multi-Task to demonstrate that MTAD-FL is more suitable for deployment in IoT cloud environments than other baseline methods. The results of the anomaly detection experiments are shown in Table 2.

Single-Task Anomaly Detection. SWaT: The Transfer model is not as good as the other baseline models regarding the recall, but F1 is 17.3% higher than the best baseline model. Accuracy is slightly lower than the Local model but higher than the other baseline models.

SMD: The Transfer model outperformed the other three methods in all metrics overall, with an accuracy rate 5.6% higher than the best baseline method, a recall rate 16.8% higher than the best baseline task, and an F1 18.3% higher than the best baseline task.

The SKAB: Transfer model was inferior to the other baseline models (except the Local model) regarding the recall but was 1.7% and 0.7% higher than the best-performing baseline model in precision and F1, respectively.

Multi-Task Anomaly Detection. In the Multi-Task case, we all trained the experiments for four federated cycles, and the results were obtained as averages.

SWaT: Our method was inferior to the other three baseline tasks in terms of accuracy but was 22.9% better than the best baseline method in terms of recall and 11% better than the best baseline method in terms of F1.

SMD: Our method outperformed the other three methods in all metrics overall, with a 29.3% higher recall than the best baseline method, a 70.6% higher recall than the best baseline task, and a 65.5% higher F1 than the best baseline task.

SKAB: Our method was more consistent in overall performance than the best overall performance of all the baseline methods.

4.5 Matthews Correlation Coefficient(MCC)

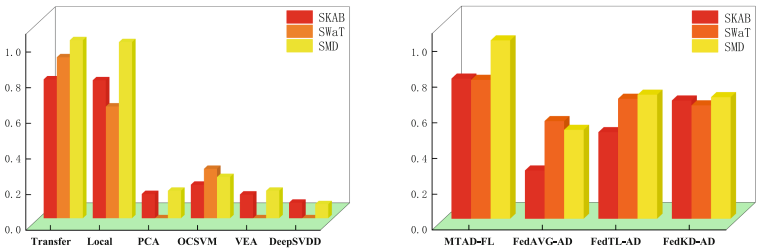

Fig. 3. The right panel shows the multitasking anomaly detection MCC results; the left panel shows the single-tasking anomaly detection MCC results. MTAD-FL and Transfer both outperform all the baseline methods.

When the MCC value is 1, the prediction of the test subject is perfect. When the MCC value is 0, the prediction is worse than the random prediction. -1 indicates that the predicted classification is entirely different from the actual classification.

The MCC values of four Multi-Task Federated Learning methods based on three different datasets are given in Fig. 3. Compared to the other three benchmark methods, our models achieve better performance, further demonstrating the effectiveness of our proposed models.

The Transfer model also performs optimally in the Single-Task Anomaly Detection experiments shown in Fig. 3, with PCA, VAE, and DeepSVDD achieving MCC results of 0 for the SWaT dataset indicating poorer prediction results than random predictions.

5 Conclusion and Future Work

In this article, we propose a multi-task anomaly detection approach based on federated learning (MTAD-FL). According to our experimental results, MTAD-FL outperforms all multi-task federated learning methods in anomaly detection of IoT cloud systems. In addition, we have also found that the model's performance is affected in the distributed case. The overall performance is better in the single-tasking case than in the multi-tasking case. The LGF-PKT framework in the experiments effectively improves specific models that perform poorly in the average weight updating. The average weight is better in an environment with fewer distributed nodes. The data variability is not too significant, yet the performance is still to be improved in the IoT-distributed cloud environment where the dynamics are high and the data variability is significant.

Future work can also be done in the following two areas. First, optimizing the performance of LGF-PKT and finding the commonality between edge nodes can improve the impact on model performance in a distributed environment. Second, our model can be improved to perform root cause localization when anomalies are detected, accurately and quickly locate anomalies in distributed IoT cloud environments, and generate responses immediately. In distributed cloud environments and generate immediate responses.

References

1. Cook, A.A., Mısırlı, G., Fan, Z.: Anomaly detection for IoT time-series data: a survey. IEEE Internet Things J. **7**(7), 6481–6494 (2019)
2. Peng, C., Yunni, X., Shanchen, P., et al.: A probabilistic model for performance analysis of cloud infrastructures. Concurrency Comput. Pract. Experience **27**(17), 4784–4796 (2015)
3. Bonawitz, K., Eichner, H., Grieskamp, W., et al.: Towards federated learning at scale: system design. Proc. Mach. Learn. Syst. **1**, 374–388 (2019)
4. McMahan, B., Moore, E., Ramage, D., et al.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, PMLR, pp. 1273–1282 (2017)
5. Liu, Y., Garg, S., Nie, J., et al.: Deep anomaly detection for time-series data in industrial IoT: a communication-efficient on-device federated learning approach[J]. IEEE Internet Things J. **8**(8), 6348–6358 (2020)

6. Ying, Z., Junjun, C., Di, W., et al.: Multi-task network anomaly detection using federated learning. In: Proceedings of the 10th International Symposium on Information and Communication Technology, pp. 273–279 (2019)
7. Hongyun, L., Peng, C., Zhiming, Z., Towards a robust meta-reinforcement learning-based scheduling framework for time critical tasks in cloud environments. In: IEEE 14th CLOUD, vol. 2021, pp. 637–647. IEEE (2021)
8. Crawshaw M. Multi-task learning with deep neural networks: a survey. arXiv preprint arXiv:2009.09796 (2020)
9. Hongyun, L., Peng, C., Xue, O., et al.: Robustness challenges in reinforcement learning based time-critical cloud resource scheduling: a meta-learning based solution. Futur. Gener. Comput. Syst. **146**, 18–33 (2023)
10. Juan, C., Peng, C., Xianhua, N., et al.: Task offloading in hybrid-decision-based multi-cloud computing network: a cooperative multi-agent deep reinforcement learning. J. Cloud Comput. **11**, 90 (2022)
11. Box, G.E.P., Pierce, D.A.: Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. J. Am. Stat. Assoc. **65**(332), 1509–1526 (1970)
12. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., et al.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
13. Daffertshofer, A., Lamoth, C.J.C., Meijer, O.G., et al.: PCA in studying coordination and variability: a tutorial. Clin. Biomech. **19**(4), 415–428 (2004)
14. Vallis, O., Hochenbaum, J., Kejariwal, A.: A novel technique for long-term anomaly detection in the cloud. In: 6th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 14) (2014)
15. Haowen, X., Wenxiao, C., Nengwen, Z., et al.: Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In: Proceedings of the 2018 World Wide Web Conference, pp. 187–196 (2018)
16. Ruff, L., Vandermeulen, R., Goernitz, N., et al.: Deep one-class classification. In: International Conference on Machine Learning, PMLR, pp. 4393–4402 (2018)
17. Yujia, S., Ruyue, X., Peng, C., et al.: Identifying performance anomalies in fluctuating cloud environments: a robust correlative-GNN-based explainable approach. Futur. Gener. Comput. Syst. **145**, 77–86 (2023)
18. Peng, C., Hongyun, L., Ruyue, X., et al.: Effectively detecting operational anomalies in large-scale IoT data infrastructures by using a GAN-based predictive model. Comput. J. **65**(11), 2909–2925 (2022)
19. Audibert, J., Michiardi, P., Guyard, F., et al.: Usad: unsupervised anomaly detection on multivariate time series. In: 26th ACM SIGKDD, pp. 3395–3404 (2020)
20. Ruyue, X., Peng, C., Zhiming, Z.: Causalrca: causal inference based precise fine-grained root cause localization for microservice applications. J. Syst. Softw. **203**, 111724 (2023)
21. Tuli, S., Casale, G., Jennings, N.R.: Tranad: deep transformer networks for anomaly detection in multivariate time series data. arXiv preprint arXiv:2201.07284 (2022)
22. Deng, A., Hooi, B.: Graph neural network-based anomaly detection in multivariate time series. AAAI-21 **35**(5), 4027–4035 (2021)
23. Chuxu, Z., Dongjin, S., Yuncong, C., et al.: A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. AAAI-19. **33**(01), 1409–1416 (2019)
24. Ruyue, X., Peng, C., Zhiming, Z.: Robust and accurate performance anomaly detection and prediction for cloud applications: a novel ensemble learning-based framework. J. Cloud Comput. **12**(1), 1–16 (2023)

25. Sater, R.A., Hamza, A.B.: A federated learning approach to anomaly detection in smart buildings. ACM Trans. Internet Things **2**(4), 1–23 (2021)
26. Nguyen, T.D., Marchal, S., Miettinen, M., et al.: DÏoT: a federated self-learning anomaly detection system for IoT. In: 2019 39th ICDCS, pp. 756–767. IEEE (2019)
27. Suyi, L., Yong, C., Yang, L., et al.: Abnormal client behavior detection in federated learning. arXiv preprint arXiv:1910.09933 (2019)
28. Yurochkin, M., Agarwal, M., Ghosh, S., et al.: Bayesian nonparametric federated learning of neural networks. In: International Conference on Machine Learning, pp. 7252–7261. PMLR (2019)
29. Yujing, C., Yue, N., Zheng, C., et al.: Federated multi-task hierarchical attention model for sensor analytics. arXiv preprint arXiv:1905.05142 (2019)
30. Qu, Z., Lin, K., Li, Z., et al.: Federated learning's blessing: Fedavg has linear speedup. In: ICLR 2021 (2021)
31. Xiang, L., Kaixuan, H., Wenhao, Y., et al.: On the convergence of fedavg on non-iid data. arXiv preprint arXiv:1907.02189 (2019)
32. Yang, H., He, H., Zhang, W., et al.: FedSteg: a federated transfer learning framework for secure image steganalysis. IEEE Trans. Netw. Sci. Eng. **8**(2), 1084–1094 (2020)
33. Liu, Y., Kang, Y., Xing, C., et al.: A secure federated transfer learning framework. IEEE Intell. Syst. **35**(4), 70–82 (2020)
34. Seo, H., Park, J., Oh, S., et al.: 16 federated knowledge distillation. Mach. Learn. Wirel. Commun. **457** (2022)
35. He, C., Annavaram, M., Avestimehr, S.: Group knowledge transfer: Federated learning of large CNNs at the edge. Adv. Neural. Inf. Process. Syst. **33**, 14068–14080 (2020)
36. Guodong, L., Ming, X., Ming, X., et al.: Multi-center federated learning: clients clustering for better personalization. World Wide Web **26**(1), 481–500 (2023)
37. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018)
38. Guo, M.H., Liu, Z.N., Mu, T.J., et al.: Beyond self-attention: external attention using two linear layers for visual tasks. IEEE Trans. Pattern Anal. Mach. Intell. **45**(5), 5436–5447 (2022)
39. Chicco, D., Jurman, G.: The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics **21**(1), 1–13 (2020)