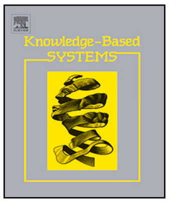
Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Knowledge-based Clustering Federated Learning for fault diagnosis in robotic assembly

Peng Xiao, Chuang Wang, Ze Lin, Ying Hao, Gang Chen [*], Longhan Xie

South China University of Technology, Guang Zhou, China

ARTICLE INFO

Keywords:
Federated Learning
Knowledge
Similarity
Industrial robot
Fault diagnosis

ABSTRACT

Fault diagnosis in industrial robots is a critical aspect of intelligent manufacturing. However, the accuracy of fault diagnosis models can be significantly affected by the few-shot problem, which refers to the limited availability of labeled data for training. Traditional methods often rely on combining data from different sources, which can raise privacy concerns due to the sensitive nature of the data involved. Federated Learning has emerged as a privacy-preserving approach, but it faces challenges in dealing with the Non-IID (Non-Independent and Identically Distributed) data distribution across different robot systems. In this study, we propose a novel approach called Knowledge-Based Clustering Federated Learning (KCFed) to address both the few-shot problem in robot fault diagnosis and the Non-IID problem in Federated Learning. KCFed incorporates Federated Learning principles to ensure privacy protection during the model training process. Additionally, by leveraging the Knowledge-based Clustering Mechanism (KCM) and Knowledge Accumulation Mechanism (KAM), KCFed aims to improve the performance of fault diagnosis models by clustering similar tasks and accumulating useful prior information. To evaluate the effectiveness of KCFed, we conducted experiments using data collected from industrial robots performing 12 different tasks, resulting in a diverse set of 48 states. The experimental results demonstrate the promising performance of KCFed in improving fault diagnosis accuracy while preserving privacy in a federated learning setting.

1. Introduction

In recent years, intelligent manufacturing has gained significant traction across industries worldwide. Industrial robots have emerged as a prominent component of intelligent manufacturing, enabling the automation of repetitive and mundane tasks. Intelligent fault diagnosis for industrial robots is a crucial aspect of ensuring smooth operations in this context. The training of fault diagnosis models plays a vital role in achieving accurate and efficient diagnosis, which requires a substantial amount of labeled data for effective training [1]. However, many factories face constraints in collecting a large volume of labeled data due to cost limitations. This challenge of limited labeled data in training fault diagnosis models is commonly referred to as the few-shot problem.

To address this few-shot problem, few-shot learning (FSL) techniques have been developed, which rely on advancements in data, models, and algorithms to reduce the cost of data collection [2]. While one factory only collects limited data, this may be not enough to cover the whole feature of distribution. Multi-task Learning involves leveraging data from multiple factories that perform similar tasks and may have data with similar distributions [3,4]. By integrating data from multiple sources, more training data can be obtained at a lower cost. However, factories that perform similar tasks are often commercial competitors, making data sharing a significant privacy risk, which is exactly what traditional methods are not good at.

To address the privacy concerns associated with sharing sensitive data, Federated Learning (FL) has emerged as a promising solution. Compared to the traditional method, it allows participants to share only the mediation parameters for collaborative and effective model training, ensuring that individual data remains secure [5]. FL encompasses various techniques [6] for sharing information, with gradient sharing and model parameter sharing being the most commonly used approaches. Recent studies, such as the work by Islam et al. [7], have demonstrated the effectiveness of FL models constructed using ensemble CNN architectures. However, manufacturing tasks can vary significantly in terms of workpieces and operations, leading to diverse distributions of robot data. In the context of FL, this type of

[*] Correspondence to: Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guang Zhou, China. Gang Chen and Longhan Xie have the same contribution.

E-mail addresses: cs_xiaopeng@mail.scut.edu.cn (P. Xiao), wichuangwang@mail.scut.edu.cn (C. Wang), 202320159835@mail.scut.edu.cn (Z. Lin), 202320163190@mail.scut.edu.cn (Y. Hao), gangchen@scut.edu.cn (G. Chen), melhxie@scut.edu.cn (L. Xie).

https://doi.org/10.1016/j.knosys.2024.111792
Received 12 October 2023; Received in revised form 20 February 2024; Accepted 8 April 2024
Available online 12 April 2024

data is commonly referred to as Non-IID (Non-Independent Identically Distribution) data, which arises from the non-identical client distributions [8]. Studies conducted by Li et al. have demonstrated that direct model aggregation under FedAvg can negatively impact the overall performance of the model when dealing with Non-IID client datasets [9]. This implies that the performance of fault diagnosis models, being data-driven models, can be significantly affected by the presence of variant data.

To address the issues mentioned above in robot fault diagnosis, this paper proposes a new learning framework called Knowledge-Based Clustering Federated Learning (KCFed). This enhances the performance of traditional federated methods to handle the few-shot problem in robot fault diagnosis and raises a novel and effective thought for solving the Non-IID problem. As FL model performance is related to data distribution, we use the Base Model to judge the similarity of tasks, which is called Similarity-based Clustering Mechanism(SCM). To make full use of the similarity model for efficient knowledge sharing, we also propose an extensible knowledge accumulation method to include tasks new to the server and to keep the high efficiency of framework, which forms a Knowledge Accumulation Mechanism (KAM).

The contributions and novelties of this paper are listed as follows:

1. The Knowledge-Based Clustering Federated Learning (KCFed) is designed for robot fault diagnosis, which improves the performance of traditional federated methods to handle the few-shot problem and Non-IID problem.

2. A Similarity-based Clustering Mechanism(SCM) is proposed to distinguish and cluster tasks of each client. This mechanism uses a novel way to solve the Non-IID problem in FL.

3. A Knowledge Accumulation Mechanism(KAM) is also proposed for accumulating prior knowledge. This method gives a further performance improvement method for the whole framework.

4. A dataset in many assembly tasks of the industrial robot is collected to simulate a real situation of many factories implementing similar work. Many slight but decisive variants are considered in tasks to express the diversity of assembly tasks in industrial scenario more comprehensively.

5. Experiments validate the KCFed framework is established, which simulates a scenario in which clients are related to various tasks connected to one cloud server.

The remaining contents of this paper are organized as follows. Related work is introduced in the second section. The third section presents the preliminary of this paper. The two main parts of this paper's method will be described in the fourth and fifth sections. The experiments validate this paper's method in the sixth section. Finally, the seventh section concludes this paper.

2. Related work

2.1. Few-shot learning

With the increasing versatility of industrial robots, it has become essential to develop specific fault diagnosis models tailored to each task. However, obtaining labeled data for training such models is challenging, as robots typically operate in normal working states. Therefore, it is crucial to efficiently train models with limited datasets. FSL techniques have been proposed to address this issue by enabling the construction of machine-learning models from a few labeled examples. In the literature, Wang et al. [2] have characterized FSL methods based on their utilization of prior knowledge in three key directions: data, model, and algorithm. Zhang et al. [10] have employed composed data augmentation operations to enhance dataset complexity and have adopted a multi-task framework to leverage information from different types of tasks. Li et al. [11] takes event data into account by leveraging event cameras and provides an event-based data augmentation way to address the few-shot problem. Finn et al. [12] have introduced the Model-Agnostic Meta-Learning (MAML) algorithm, which initializes model parameters through fine-tuning with a few datasets, showing promise in model parameter sharing for FSL. Sung et al. [13] have proposed the Relation Network, which compares embeddings and computes relation scores for image classification. These existing methods offer valuable insights into solving the few-shot problem from different perspectives. This paper draws inspiration from these approaches and incorporates data augmentation techniques and parameter sharing to efficiently train fault diagnosis models. By leveraging these strategies, the proposed framework aims to improve the effectiveness of few-shot learning in the context of industrial robotics.

2.2. Federated learning

Considering the growing emphasis on the protection of data privacy in various fields, McMahan et al. [6] proposed a framework for Learning based on personal terminal devices, alongside the widely used Federated Averaging (FedAvg). This framework demonstrated the effectiveness of federated learning models compared to local training methods. Kairouz et al. further highlighted the capability of Federated Learning to share information and knowledge for model training without the need for data exchange [8]. To enhance the convergence speed of multiple clients in parallelism, Zhong et al. introduced P-FedAvg as an improvement over the FedAvg algorithm [14]. Additionally, Bonawitz et al. proposed a communication-efficient and failure-robust protocol for Federated Learning, providing practical guidance for implementing this approach [15]. However, it is worth noting that this study does not consider the communication aspect between clients and servers. In industrial robot scenarios, many fields are data-sensitive [16]. The application of Federated Learning in industrial robot scenarios, where data sensitivity is a significant concern, has garnered increasing attention [16]. Nevertheless, when the disparity between client data in Federated Learning is substantial, it can lead to slower convergence speeds and lower accuracy [17]. In the industrial scenario, the Non-IID problem can affect model accuracy deeply, Zhang et al. [18] introduce a federated transfer learning method for fault diagnosis to address the Non-IID problem. To handle the Non-IID problem, this paper focuses on clustering similar tasks in Federated Learning to improve its overall performance.

3. Preliminary

In this paper, a distributed situation in which many clients connect to the center cloud server is introduced, simulating a common real situation. Each client relates to a factory with industrial robots doing a specific task. However, each client only have limited labeled data that is insufficient to train a well-performed robot fault diagnosis model.

The few-shot problem is introduced to train an effective fault diagnosis model under the condition that labeled data is limited because of the high cost of collecting data. This paper proposes a method based on federated learning, which can integrate labeled data from different clients in a private way.

While applying federated learning, the Non-IID problem is unavoidable. This problem is investigated in situations where clients in a federated framework have none-independent identity data distribution which affects the efficiency of FL.

So the research is based on some assumptions presented as follows:

(1) Datasets of each client are non-visible to each other, and the process of transferring models is private for each one. This secures the privacy security to some degree.

(2) The sample size of datasets in each client is not big, since industrial robots are usually in normal working states and faults are rare.

(3) There are many clients connected to the cloud server and for each client there will be enough clients with similar tasks.

(4) There is only one work for one client, which means tasks implemented by each robot are with specific size and shape and have identity data distribution in one client.

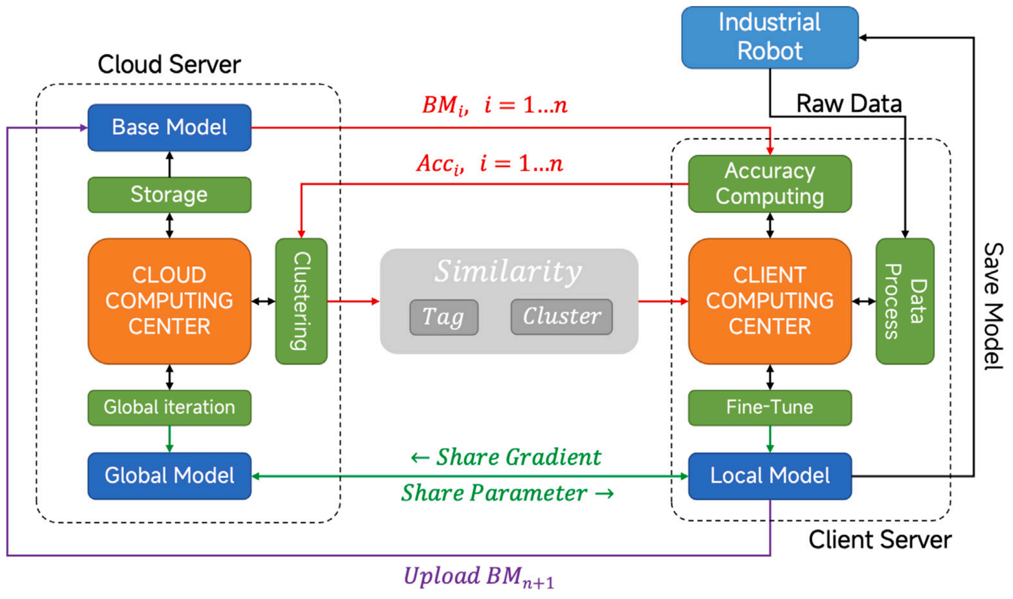


Fig. 1. Framework of Knowledge-Based Clustering Federated Learning.

4. Federated learning framework for robotics fault diagnosis

Fig. 1 shows the framework of the method this paper proposed. In this framework, the "Black Line" represent the data stream between individual robots and client computing center, robot's raw data will send to client and local model will save in each robot; the "Red Line" represent SCM, which computes Similarity for Federated Learning and Knowledge Accumulation; "Green Line" represent Clustering Federated Learning Framework, using Federated algorithm (e.g. Fedavg) to update Global model and Fine-Tune to update Local model; "Purple Line" represent KAM, uploading new Base model base on similarity.

4.1. Scenario building

The scenario in this paper contains a cloud server and several clients. Cloud server in charge of connecting each client, storing and iterating global model. Cloud server also possesses Base Models(BM) $B_i(i = 1, 2, \dots, N)$. Each BM is trained from a dataset corresponding specific dataset selected by prior knowledge which we select from totally different tasks. Clients are related to factories where industrial robots work with specific tasks and the tasks may be different or similar to each other.

4.1.1. Deep learning model for fault diagnosis

In this paper, both BM and the fault diagnosis method used in KCFed are modified from the model Wen et al. [19] used. It contains two parts. The first part converts the time-domain signal to images. Then, taking the images as input to the CNN model. The CNN model based on LeNet-5 has been proven to be simple and effective in fault diagnosis [20].

For our data composition, specific modifications are made to the original model to improve the performance in the test. The input of the CNN model is a 36*36 image consisting of data. The model used in this study has three convolutional layers and pooling layers. After these layers, there are several fully connected layers with a softmax classifier. The basic structure of the model used in this study is shown in Table 1.

Table 1
Architecture of the CNN model.

Layer	Function	Size
Input	Input Figure	1 × 36 × 36
Conv 1	Convolution	32 × 36 × 36
Pool 1	Pooling	32 × 18 × 18
Conv 2	Convolution	64 × 18 × 18
Pool 2	Pooling	64 × 9 × 9
Conv 3	Convolution	128 × 9 × 9
Pool 3	Pooling	128 × 4 × 4
Re	Reshape	2048 × 1 × 1

4.1.2. Dataset sampling

To demonstrate that our method is effective in real industrial robot diagnosis scenarios, we pick the following three common industrial assembly task families: Peg-in-Hole assembly task family, Gear assembly task family, and Ring assembly task family. All of them are considered challenging and uncertain [21]. Under each assembly task family, there are different tasks with slight but decisive variants. 4 kinds of pegs, gears, and rings with different sizes and shapes are designed as tasks. To express the above tasks more concisely, we use P_i $(i = 1, 2, \dots, 4)$ to represent the ith subtask of Peg-in-Hole assembly task family, use G_j $(j = 1, 2, 3, 4)$ to represent the jth task of Gear assembly task family and use R_k $(k = 1, 2, 3, 4)$ to represent the kth subtask of Ring assembly task family. The dataset that this paper collected is shown as Fig. 2.

In addition to designing the above assembly tasks, this paper has taken the possible failures in real assembly situations. For each task, there are four situations. They correspond to $Normal\ Assembly$, $Workpiece\ Tilt$, $Repeated\ Assembly$, and $Workpiece\ Missing$. The assembly tasks are tested in a simulated environment. The robot model is from UR5 which has 6 DOF. The 3D model of the workpieces required for the task is modeled and exported by 3D software. After that, the model is published to Gazebo through ROS, and the physical properties of the object are set in the corresponding configuration file to achieve high-precision simulation. Fig. 2 shows the assembly tasks in the simulation environment.

In terms of data sampling, the sampling frequency is influenced by various factors, including the nature of the sensor itself, communication delays, and the data acquisition code. After testing, the sampling frequency in the experiment was found to be approximately and stably close to 120 Hz. In this study, the data collected during the assembly tasks contain joint position and joint torque of the six-axis robotic arm. Therefore, the format of data is temporal row vectors with 12 dimensions. Considering the difficulty in collecting fault data and further experiments, we collect data for 30 assembly processes in each case. In these simulated scenarios, the robot learns how to perform the assembly task through reinforcement learning which makes it robust.

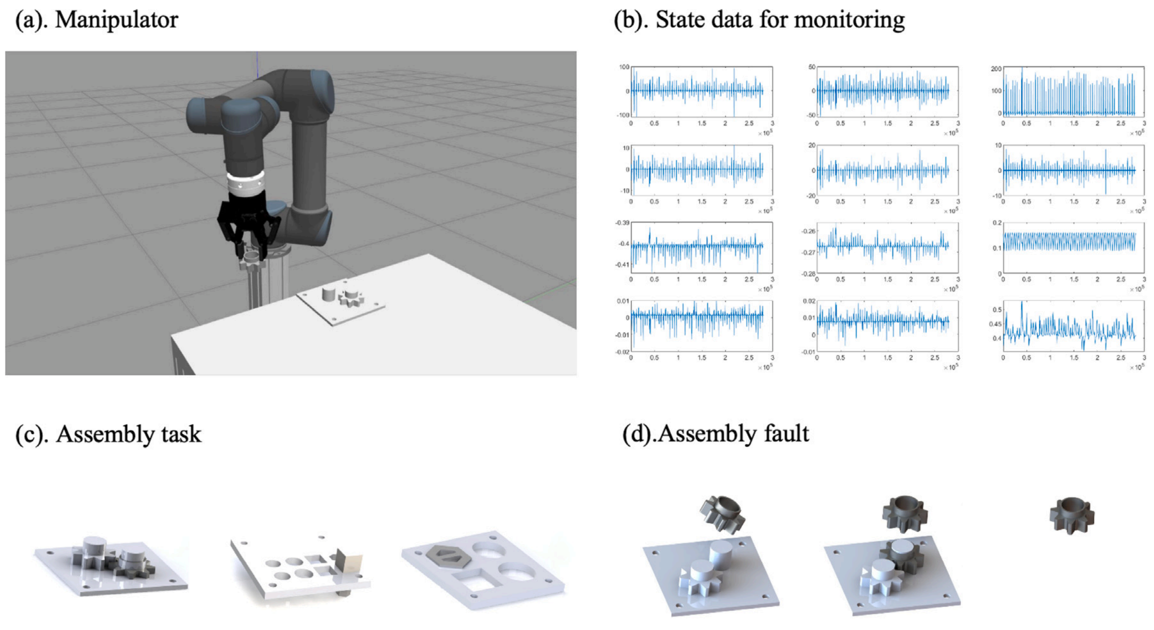


Fig. 2. Introduction to assembly tasks. (a) is the manipulator for the assembly task, (b) shows the six-dimensional pose and torque data of the robotic arm during assembly, (c) is three different types of assembly tasks, including Peg-in-Hole Assembly, Gear Assembly and Ring Assembly, (d) shows three abnormal situations including workpiece tilt, repeated assembly and workpiece missing.

So for each task, there is a specific and robust control mechanism, and all four cases of each task use the same mechanism. Random positional errors are added every time an assembly task is performed. It means that each assembly action is unique and makes it reasonable to simulate complete tasks at different conditions.

4.1.3. Data processing and augmentation

To solve the problem of different period lengths, each period is divided into twelve equal parts, and the same numbers of data are selected randomly from each part and then combined sequentially. To overcome the few-shot problem, by selecting data randomly from one part several times, we can get multiple sets of data from one period. Since the input data figure size of our CNN is 36*36, the length of the combined data of each period is set as 1296. The next step is to convert the one-dimensional time series data into two-dimensional data figures through matrix transformation [19]. The converted data figure will be the input of the CNN model. To avoid the augmented data from one period appearing in both the training set and the testing set, this paper divides the training and testing sets before data augmentation.

4.2. Clustering federated learning framework

This paper proposes a Clustering Federated Framework. It is comprised of one cloud server and several clients. Each client represents a factory with data from an industrial robot. In detail, the framework adopts a federated gradient descent approach. In each federated learning cluster, all clients share the same global model. In each training epoch, part of the parameters of the global model will be sent to each client to build its local model, then clients will iterate local models by local data and send the same part of gradients back to the cloud. Cloud server will average gradients from specific clients that Mechanism (introduce in next section) decided, then use that averaging gradient to iterate global model. In the next epoch, the new global model's parameter will be sent to each client repeatedly.

Furthermore, each client will train the received model by using local data and test the accuracy of the trained (fine-tuned) local model. When accuracy is higher than their highest accuracy fault diagnosis model in the previous training epoch, clients can save this fine-tuned model. Each client should continue to participate in federated learning for a specific number of epochs. The final model each client gets from this framework is their highest accuracy fine-tuned model. The framework executes just like Algorithm 1.

Algorithm 1 Clustering Federated Learning framework

```
Input: Initialized Clients $C_i$, corresponding Local Model $M^i_{local}$, the
    number of Clients $N_{clients}$, local data $D^i$, Cloud server $S$, the number
    of training epochs $N_{train}$
1:  $Cluster, Tag \leftarrow SimilarityClustering()$
2:  $Num \leftarrow CountCluster(Cluster)$
3:  for each Cluster Number $j = 0, 1, ..., Num$ do
4:    Initial Global Model $M^j_{global,0}$
5:    Initial Max accuracy of Local Model $A^i_{max} = 0$
6:    for each training epoch t = 1,2,...,$N_{train}$ do
7:      for each Client $C_i$ i = 1,2,...,$N_{clients}$ do
8:        if $Cluster_i == j$ then
9:          $S$ send parameters of $M^j_{global,t}$ to $C_i$
10:         $M^i_{local} \leftarrow LocalTraining(M^i_{local}, D^i_{train})$
11:         Gradient $G^i_{local} \leftarrow GetGradient(M^i_{local})$
12:         Accuracy $A^i \leftarrow Testing(M^i_{local}, D^i_{test})$
13:         if $A^i > A^i_{max}$ then
14:           Save $M^i_{local}$
15:           Set $A^i_{max} = A^i$
16:         end if
17:       end if
18:     end for
19:     $G^j_{global} \leftarrow Average(G^1, ..., G^{N_{clients}}, Tag)$
20:     $M^j_{global,t+1} \leftarrow Iterate(M^j_{global,t}, G^j_{global})$
21:     if all client $C_i$ have saved $M^i_{local}$ then
22:       Stop training in this Cluster
23:     end if
24:   end for
25: end for
```

In Algorithm 1, Function "*SimilarityClustering*" is the SCM which Algorithm 2 described in detail. In step two function "*CountCluster*" counts the number of clusters. Cluster and Tag correspond to each client. all clients will divide into several clusters and implement federated learning with the cluster's global model.

5. Knowledge-based clustering federated learning method

5.1. Similarity-based clustering mechanism

The key point of solving the few-shot problem is to utilize information from others. To mitigate the impact of few-shot, it is common to think of enlarging the information source. But in real situations, as clients connected to FL are incommunicado, the Non-IID problem is a fundamental challenge in improving the effectiveness and efficiency of FL [8]. Also, conflicting gradients among different tasks can be a negative transfer problem [22]. Both of them will be enhanced as the number of clients increases.

To trade off the impact of enlarging and controlling the number of clients in FL, this paper proposes a method called Similarity-Based Clustering Mechanism(SCM). SCM applies an accuracy-based similarity measuring method. The similarity measuring method uses BM as a measuring tool and common knowledge. All BM will sent to every client at the initial time. Each client will use BM and local data to check accuracy. As BM was selected by prior knowledge, each BM was selected from a representative task in each task family initially. The higher the accuracy, the closer the client and task family are. By using this measuring method, the cloud server can only know what clients are similar to the task family but cannot know any technical details of the client's local task since the data are not shared. All accuracy of BM related to a client will sent to the cloud server and the cloud server will group the client following the highest accuracy. The process of SCM is shown in Algorithm 2.

Algorithm 2 Similarity-based Clustering Mechanism

```
Input: Clients $C_i$, Base Model $BM_j$, Accuracy of Base Model at each
    Client $Acc_{i,j}$
1: Accuracycomputing ← Clients
2: for each client $C_i$ do
3:     for each base model $BM_j$ do
4:         test $Acc_{i,j}$
5:     end for
6: end for
7: Clustering ← Cloud server
8: for each client $C_i$ do
9:     $ACC_i = \max_{j\in(1,N)} Acc_{i,j}$
10:    $Cluster_i = \underset{j\in(1,N)}{\text{argmax}}\, Acc_{i,j}$
11: end for
12: for each group $j = 0, 1, 2, ..., N$ do
13:    $Mean_j = \underset{i\in(1,M)}{\text{mean}}\, ACC_i$
14:    $Std_j = \underset{i\in(1,M)}{\text{std}}\, ACC_i$
15:    $Threshold_j = Mean_j - 2 * Std_j$
16: end for
17: for each client $i = 0, 1, 2, ..., M$ do
18:    if $ACC_i > Threshold_{Cluster_i}$ then
19:        $Tag_i = 1$
20:    else
21:        $Tag_i = 0$
22:    end if
23: end for
Output: $Cluster, Tag$
```

The SCM is composed of three parts: accuracy computing, clustering, and training-updating. Cloud server sends base models to each client to test. After receiving the accuracy of each base model from each client, cloud server will cluster and send similar models back. Each client optimizes its model based on local datasets.

5.1.1. Accuracy computing

Cloud server possesses BM $BM_j, j = 1, 2, \ldots, N$. Client $C_i, i = 1, 2, \ldots, M$ test base models according to local datasets $D_i, i = 1, 2, \ldots, M$. The accuracy of each base model for each client is denoted as $Acc_{i,j}, i \in (1, M)\ j \in (1, N)$.

5.1.2. Clustering

For each client C_i, the cloud server calculates the maximum accuracy individually. The highest accuracy determined which cluster client C_i belongs to

$$Cluster_i = \underset{j\in(1,N)}{\text{argmax}}\, Acc_{i,j} \tag{1}$$

$Cluster_i$ represent a cluster index for the C_i client, which corresponding to highest accuracy of this client calculated by all BM. All clients with the same $Cluster_c$ will be assumed to be similar and share a common global model while applying a federated learning framework.

5.1.3. Training-updating

Each client initializes the model with parameters received from cloud server and fine-tunes it with local datasets. According to KAM, client uploads decide whether to upload the gradient of local model to cloud server. Cloud server iterates the global model by averaging the gradient from clients.

5.2. Knowledge accumulation mechanism

As the prior knowledge cannot cover all kinds of industrial robot tasks, there are always clients with new tasks from different task families connected to the cloud server. The self-evolution ability is under this paper's consideration. Because BM works as a similarity measuring tool, each client checks which task family is closer. BM is quite like the knowledge for this method. So this paper considers a Knowledge Accumulation Mechanism(KAM) that accumulates BM when a client with a new task connects to the cloud server.

Knowledge of client's task similar enough to BM's family does not need to be accumulated and only the knowledge of the new task is valuable. But when a client with a new task connects to the cloud, it will affect the effectiveness of global model in a cluster since in this method all clients will be grouped into a cluster and their gradient will be sent to iterate the global model. So the best way is to accumulate new knowledge but not affect the other client's global model. It is need to build a decision paradigm for deciding which knowledge is needed and whether client's gradient should be sent to the cloud server.

In KAM, each base model calculates Relation Threshold Value $Threshold_j$. The first step is calculating maximum accuracy of clients ACC_i's average $Mean$ and standard deviation Std in one cluster. This paper assumes

$$Threshold_j = Mean_j - 2 * Std_j \tag{2}$$

and the client who has accuracy ACC_i higher than $Threshold_j$ is similar to the other client and can fit the base model. Corresponding base model parameters are distributed to each client with the new base model tag which is denoted as Tag_i in Algorithm 2.

This paper assumes that for randomly selecting a client from a cluster, its accuracy ACC_i follows Gaussian distribution. And according to the regular pattern of Gaussian distribution, the possibility for client's ACC_i lower than $Threshold_j$ is about 0.02275, which is a small enough possibility. client with ACC_i lower than $Threshold_j$ will be treated as having a new task, and its gradient will not need to be sent to the cloud server but can gain parameters of global model for better accuracy. As a price, the trained fine-tuned model of this client will be sent to the cloud server for a new base model.

6. Experiment and discussion

6.1. Baseline-impact of few-shots problem

6.1.1. Experiment setting

To figure out that few-shot problem affect the fault diagnosis model's efficiency intensely, this paper build up lots of baseline experiments with different data sizes. The training-testing sets splitting

Table 2
Accuracy of fault diagnosis model that training by each dataset in different size.

	P_1	P_2	P_3	P_4	G_1	G_2	G_3	G_4	Average
Period=25	0.9433	0.9816	0.9283	0.9250	0.9416	0.9583	0.9650	0.9233	0.9424
Period=15	0.8527	0.9944	0.8667	0.9250	0.8167	0.9417	0.9056	0.9222	0.8956
Period=5	0.8333	0.8083	0.8916	0.8000	0.8083	0.8916	0.8333	0.8666	0.8398
Decline	11.66%	17.66%	3.94%	13.51%	15.15%	6.96%	13.64%	6.14%	10.83%

protocol refers to some normal few-shot learning experiments [23]. As mentioned in the previous part of this paper, the dataset concerning each task's state has 30-period record data. In this paper, baselines experiment on training fault diagnosis model using different numbers of period data, which are period=25, period=15, and period=5, which can also be seen as large size, middle size, and small size.

There is usually overfitting when training models in few-shot problems. The baseline experiment used all 4 state data from each task in different period sizes, recording the testing accuracy for each model of subtask under the condition that training each fault diagnosis model until it hardly changes in a few training epochs.

6.1.2. Impact of few-shots problem in industrial robot fault diagnosis

Table 2 shows the result of the baseline experiment. “Period=25”, “Period=15”, and “Period=5” represent all data sizes for this task's model. The element in this table is testing accuracy for specific subtasks and data sizes. Different tasks affected by few-shots are variant, with the decline in the percent of testing accuracy from “Period=25” to “Period=5” up to 17.66% and down to 3.94%. With an average 10.83% decline.

This paper figures out that few-shot problems could affect the efficiency of fault diagnosis models in industry. For all tested tasks, their training accuracy in different period sizes are near 0.98 1. It also means that as the data size goes smaller, the level of overfitting is higher. To avoid the influence of other conditions on the experiment, the following experiments are all using small size of data.

6.2. Ablation study of federated learning framework

6.2.1. Ablation study of parameter sharing method

Hard parameter-sharing method is a classical method for multi-task learning that can back to Caruana [24]. This paper uses several levels of Multi-task Methods to check out whether is the best match for the condition. In detail, this paper uses the method of sharing parameters of all convolutional layers and the different numbers of fully connected layers with many tasks, and each task will relate to specific other fully connected layers. To simulate a practical situation, this experiment uses clients with small data size $Learning\ dataset = \{P_1, P_2, P_3, P_4, G_1, G_2, G_3, G_4\}$ from all tasks in two families to train fault diagnosis model, P_x represent tasks in Peg-in-Hole and G_x represent tasks in Gearing. Each task have two corresponding clients in the experiment, in other words, there are 8 tasks and 16 clients in this experiment.

To verify the impact of the Hard parameter-sharing method, this paper also introduces a controlled experiment. By using the same data in the experiment concerning the Hard parameter sharing method, the controlled experiment used a uniform fault diagnosis model for each task.

The ablation study result is shown in Fig. 3. It shows average testing accuracy varies in 100 epochs concerning different methods. “ALL” represents that all of the parameters of the global model are shared to each client. Since all 4 conditions share convolutional layers in each client. “Full0” denotes that no parameter of the fully connected layer is shared. “Full1” means all clients share the parameter of the first fully connected layer. “Full2” means all clients share the parameters of the first and second fully connected layers. From this result, 3 Hard parameter sharing methods (“Full”) follow the rule that: the more parameter sharing, the slower increase of testing accuracy, and the

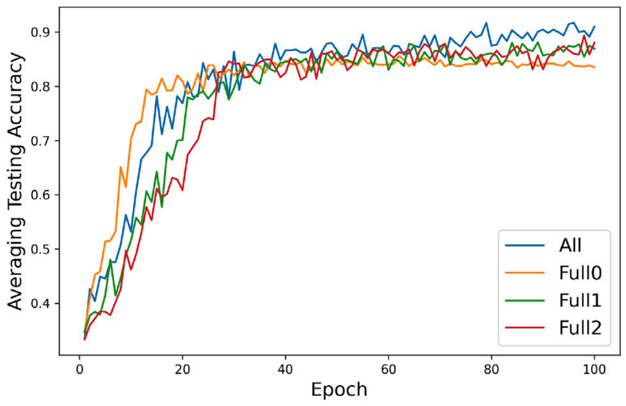


Fig. 3. All Result Collection for Federated learning with different levels of parameter sharing.

higher accuracy in the final phase. But for “All”, it performs the best accuracy in the final phase and the second fastest increasing velocity of accuracy.

Such results indicate a fact that some methods leave parts of the parameter to train from few-shots data may lead to serious overfitting since in this experiment all 4 methods reach a near 100% training accuracy. Even the Hard parameter sharing method is thought to have the ability to reduce the degree of overfitting of the model compared to training all parameters from local data [25]. Furthermore, this fact can be an internal cause for the previous rule of the parameter sharing method, it is that less parameter sharing means more parameter needs to be trained by few-shots data, which is against Occam's razor.

6.2.2. Ablation of training framework

Federated averaging method also called Fedavg [6], is one of the most popular methods in federated learning. This paper simulates that each client is related to robot doing one task. The parameter of global model will sent to clients for building local model. Clients will fine-tune the local models by robot's data with small size. Gradients after client training will be sent back to the cloud. Cloud server will iterate global model using the averaged gradient. This experiment using small size $Learning\ dataset = \{P_1, P_2, P_3, P_4, G_1, G_2, G_3, G_4\}$ from all task in two family like previous experiment.

To verify the universality of the impact of federated learning, this experiment chooses two ways of sharing model and parameter. By only sharing parameters of convolutional layers or all layers' parameters.

In Fig. 4, “Hard-X” represents using data from the different tasks, but only some of the parameters of convolutional layers are shared with each task, the other parameters are individual and relate to different tasks. “Uniform-X” represents using the same data as “Hard-X”, but using all data to train a single fault diagnosis model. “Y-Local” represents the training and testing fault diagnosis model locally; “Y-Federated” represents the training and testing model by using the federated framework with Fedavg method. As in the framework, after fine-tuning the local model, each client will check the testing accuracy Tes_i and training accuracy Tra_i. To show the relative relationship briefly, Fig. 4 using averaging testing accuracy

$$Testing\ accuracy = \frac{\Sigma_0^i Tes_i}{\Sigma_0^i 1} \tag{3}$$

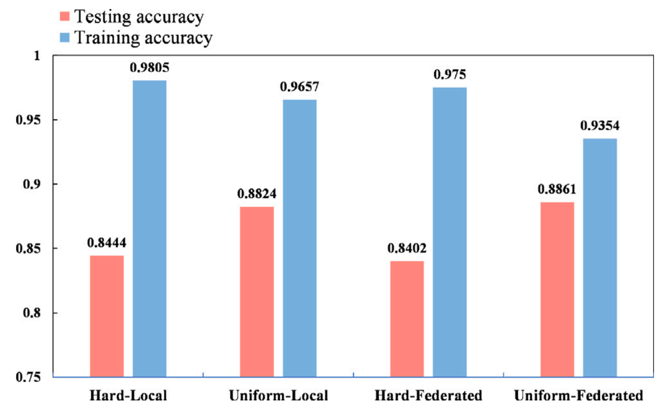


Fig. 4. Training and Testing Accuracy of model training with different method.

and averaging training accuracy

$$Training\ accuracy = \frac{\Sigma_0^i Tra_i}{\Sigma_0^i 1} \tag{4}$$

which computes from accuracy of each task.

With the result in Fig. 4, this paper proves that federated framework makes no negative impact on the final effect of fault diagnosis model comparing the local training. Furthermore, federated framework even has a positive impact on reducing overfitting level. Comparing to Table 2, averaging testing accuracy for all clients using federated framework is higher than result in "Period=5" which the client is trained with only local data. But it still has a distance to "Period=25" which simulates having enough data. This is due to a common problem for federated learning, which is the Non-IID problem. To solve this problem, this paper proposes an SCM and verifies it in the next section.

6.3. Ablation study of similarity-based clustering mechanism

This paper proposes a method combines multi-task learning and federated learning for solving few-shot problem in robot fault diagnosis, also using similarity to achieve cluster federated learning and knowledge accumulation. This experiment testing SCM proposed by simulating a normal situation cloud has connected many clients and some of their robots are doing similar tasks.

For ablation study, this paper uses the same data and clients from the previous experiment, which has 16 clients. This experiment chose 2 BM which from the Peg-in-Hole and Gear Assembly task family, by calculating the accuracy of 2 BM, 16 clients can be divided into two groups.

This paper also randomly divides the entire dataset into training dataset and testing dataset to avoid training data in one client appearing at testing data in the other clients. As in 16 clients, there are some clients using data related to the same subtask. This paper uses random algorithm to make sure that each small size data used in client is randomly selected in the entire training or testing dataset for this task.

Fig. 5 shows that each group reach a higher averaging accuracy compared to unclustering accuracy after divided into 2 groups. The accuracy for two clusters is calculate by the client with "Tag=1" which share the gradient with global model. Cluster0 include clients closer to BM selected from Peg-in-Hole family and Cluster1 include clients closer to BM selected from Gear family.

Above all show that clustering federated learning can unify knowledge from different clients with similar knowledge and transfer industrial common knowledge to other clients. Compared to "Period=25" in Table 2, the accuracy of Cluster0 and Cluster1 are higher or closer to the accuracy from large data size.

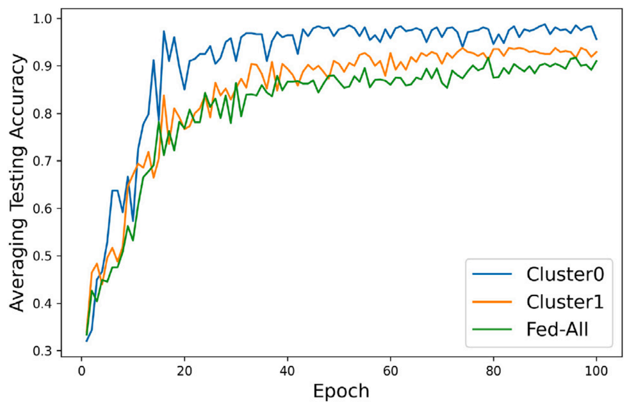


Fig. 5. All Result Collection for Clustering Federated Learning in two Cluster and comparing with normal Federated Learning.

Table 3
Epoch of training fault diagnosis model when new task connect to server.

	R_1	R_2	R_3
Before KAM	63	49	117
After KAM	25	38	65
Decreasing level	60.3%	22.4%	44.4%

Table 4
Accuracy of fault diagnosis model when new task connect to server.

	R_1	R_2	R_3
Before KAM	0.9625	1	0.9375
After KAM	0.9718	1	0.9406
Increasing level	0.95%	0%	0.33%

6.4. Ablation study of knowledge accumulation mechanism

To testify the effect of KAM, this experiment also simulate a real-world condition and divide it into two situations.

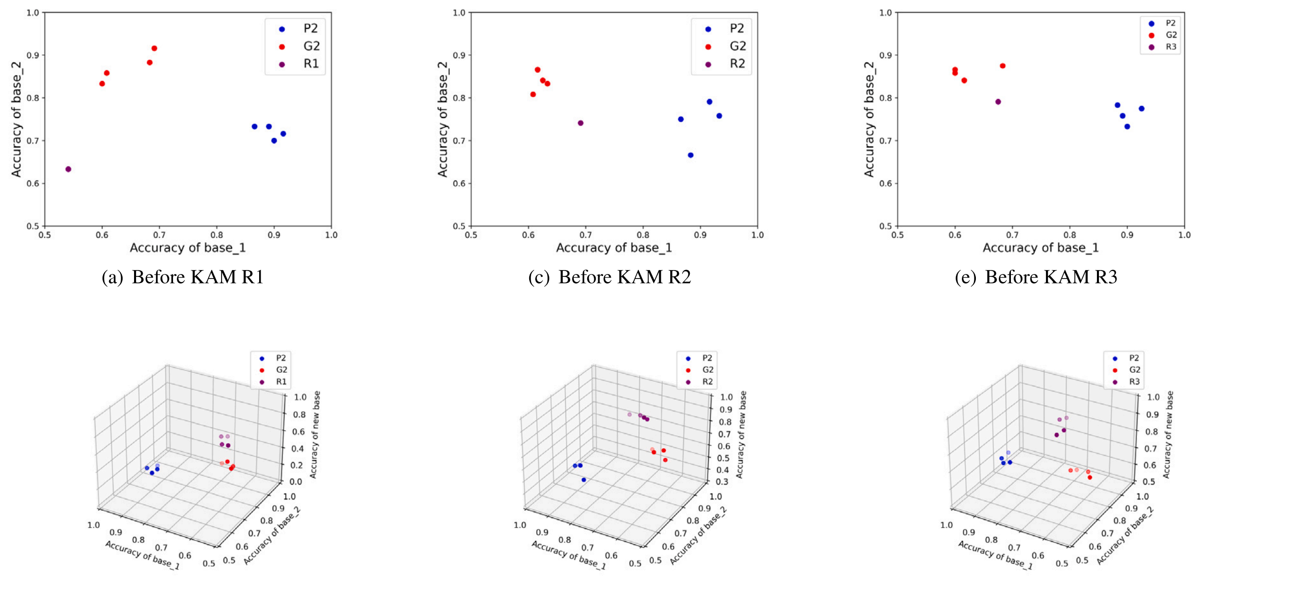
$$Dataset = \begin{cases} P_2, \dots, P_2, G_2, \dots, G_2 + R_i, & (a) \\ P_2, \dots, P_2, G_2, \dots, G_2 + R_i, \dots, R_i, & (b) \end{cases} \tag{5}$$

$which\ i = 1 \dots 4$

The first situation "a" is set to simulate a client with a new task connected to cloud server. The second situation "b" is set to simulate many clients with the new task mentioned in the first group connect to the cloud server after KAM. All clients related to the Ring task family are tested to verify the robustness of KAM.

"Tag" shown in Algorithm 2 represents whether the similarity of clients higher than the Relation Threshold Value $Threshold_j$. When "Tag" = 0 means similarity is not higher than $Threshold_j$ and this client's gradient will not be shared to iterate the global model but the client can receive the parameter from global model. After the end of federated learning, the model of client's "Tag"=0 will sent to the cloud to be a new base model. As some tasks are similar to each other, by such a KAM mechanism the SCM will not think of the fault diagnosis model of a very similar task to be a new base model in the cloud, which can maximize the utilization of industrial common knowledge.

Fig. 6 shows the impact of knowledge accumulation. A new base model will be stored and used for checking similarity in more detail. Initially, base models for P_2 and G_2 are used. R_1, R_2, R_3 are divided to Cluster with G_2 as in (a) (c) (e) of Fig. 6, R_i are closer to G_2 point than P_2 point. After KAM, several R_1, R_2, R_3 are divided into one cluster as in (b) (d) (f) of Fig. 6, which achieves a better clustering result. Considering the efficiency and accuracy of training fault diagnosis model, KAM can also improve the performance of training as shown in Tables 3 and 4. And we can find that training to a specific accuracy



(a) Before KAM R1 (c) Before KAM R2 (e) Before KAM R3

(b) After KAM R1 (d) After KAM R2 (f) After KAM R3

Fig. 6. The figure concerning KAM.

(e.g. 0.92) needs averaging epoch is extremely decreased 42.4% and the averaging final accuracy is also increased by 0.43%.

Previous experiments verify the effect of KAM. KAM realizes selective accumulation of knowledge and improves the efficiency of the SCM.

7. Conclusion

This paper introduces a novel approach called Knowledge-Based Clustering Federated Learning for fault diagnosis in industrial robots. The method utilizes a selected Base model as a similarity measuring tool. Initially, clients are clustered based on similarity calculations, and a "Tag" is assigned to determine whether to share gradients. The fault diagnosis model is then trained using the Federated Learning framework. In the case of a client with a new task connecting to the server, the Base Model judges whether to send the client's model to the server for Knowledge Accumulation and become the new Base Model. The results show that this method, using the Federated Learning framework, achieves high accuracy comparable to training with large data size, effectively addressing the few-shot problem. Additionally, the paper addresses the Non-IID problem by using an SCM and enhances the overall efficiency of the method by using a KAM.

Funding

This document is the results of the research project funded by the National Natural Science Foundation of China (Grant No. 52075177 and 52305105) the National Key Research and Development Program of China (Grant No. 2021YFB3301400), Research Foundation of Guangdong Province (Grant No. 2019A050505001 and 2018KZDXM002), Guangzhou Research Foundation (Grant No. 202002030324 and 201903 010028), Zhongshan Research Foundation (Grant No. 2020B2020).

CRediT authorship contribution statement

Peng Xiao: Writing – original draft, Software, Methodology, Conceptualization. **Chuang Wang:** Writing – review & editing, Supervision, Methodology. **Ze Lin:** Writing – original draft, Formal analysis, Data curation. **Ying Hao:** Writing – original draft, Software. **Gang Chen:** Supervision. **Longhan Xie:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] J. Wang, W. Qiao, L. Qu, Wind turbine bearing fault diagnosis based on sparse representation of condition monitoring signals, IEEE Trans. Ind. Appl. 55 (2) (2018) 1844–1852.

[2] Y. Wang, Q. Yao, J.T. Kwok, L.M. Ni, Generalizing from a few examples: A survey on few-shot learning, ACM Comput. Surv. (CSUR) 53 (3) (2020) 1–34.

[3] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, C. Finn, Efficiently identifying task groupings for multi-task learning, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J.W. Vaughan (Eds.), in: Advances in Neural Information Processing Systems, vol. 34, Curran Associates, Inc., 2021, pp. 27503–27516, URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/e77910ebb93b5115885578063l0f78f1-Paper.pdf.

[4] E. Yang, J. Pan, X. Wang, H. Yu, L. Shen, X. Chen, L. Xiao, J. Jiang, G. Guo, AdaTask: A task-aware adaptive learning rate approach to multi-task learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, (no. 9) 2023, pp. 10745–10753, http://dx.doi.org/10.1609/aaai.v37i9.26275, URL: https://ojs.aaai.org/index.php/AAAI/article/view/26275.

[5] Y. Lu, X. Huang, K. Zhang, S. Maharjan, Y. Zhang, Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles, IEEE Trans. Veh. Technol. 69 (4) (2020) 4298–4311.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273–1282.

[7] M. Islam, M. Reza, M. Kaosar, M.Z. Parvez, et al., Effectiveness of federated learning and CNN ensemble architectures for identifying brain tumors using MRI images, Neural Process. Lett. (2022) 1–31.

[8] P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., Advances and open problems in federated learning, Found. Trends® Mach. Learn. 14 (1–2) (2021) 1–210.

[9] T. Li, A.K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, in: Proceedings of Machine Learning and Systems, vol. 2, 2020, pp. 429–450.

[10] R. Zhang, Y. Yang, Y. Li, J. Wang, H. Li, Z. Miao, Multi-task few-shot learning with composed data augmentation for image classification, IET Comput. Vis. (2022).

[11] X. Li, S. Yu, Y. Lei, N. Li, B. Yang, Intelligent machinery fault diagnosis with event-based camera, IEEE Transactions on Industrial Informatics (2023).
[12] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: International Conference on Machine Learning, PMLR, 2017, pp. 1126–1135.
[13] F. Sung, Y. Yang, L. Zhang, T. Xiang, P.H. Torr, T.M. Hospedales, Learning to compare: Relation network for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1199–1208.
[14] Z. Zhong, Y. Zhou, D. Wu, X. Chen, M. Chen, C. Li, Q.Z. Sheng, P-FedAvg: Parallelizing federated learning with theoretical guarantees, in: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, IEEE, 2021, pp. 1–10.
[15] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H.B. McMahan, S. Patel, D. Ramage, A. Segal, K. Seth, Practical secure aggregation for privacy-preserving machine learning, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 1175–1191.
[16] L. Li, Y. Fan, M. Tse, K.-Y. Lin, A review of applications in federated learning, Comput. Ind. Eng. 149 (2020) 106854.
[17] X. Peng, Z. Huang, Y. Zhu, K. Saenko, Federated adversarial domain adaptation, 2019, arXiv preprint arXiv:1911.02054.
[18] W. Zhang, X. Li, Data privacy preserving federated transfer learning in machinery fault diagnostics using prior distributions, Structural Health Monitoring 21 (4) (2022) 1329–1344.
[19] L. Wen, X. Li, L. Gao, Y. Zhang, A new convolutional neural network-based data-driven fault diagnosis method, IEEE Trans. Ind. Electron. 65 (7) (2017) 5990–5998.
[20] U.-P. Chong, et al., Signal model-based fault detection and diagnosis for induction motors using features of vibration signal in two-dimension domain, Strojniški Vestnik 57 (9) (2011) 655–666.
[21] M. Nigro, M. Sileo, F. Pierri, K. Genovese, D.D. Bloisi, F. Caccavale, Peg-in-hole using 3D workpiece reconstruction and CNN-based hole detection, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2020, pp. 4235–4240.
[22] W. Zhang, L. Deng, L. Zhang, D. Wu, A survey on negative transfer, 2020, arXiv preprint arXiv:2009.00909.
[23] J. Wu, Z. Zhao, C. Sun, R. Yan, X. Chen, Few-shot transfer learning for intelligent fault diagnosis of machine, Measurement 166 (2020) 108202.
[24] R. Caruana, Multitask learning: A knowledge-based source of inductive bias1, in: Proceedings of the Tenth International Conference on Machine Learning, Citeseer, 1993, pp. 41–48.
[25] J. Baxter, A Bayesian/information theoretic model of learning to learn via multiple task sampling, Mach. Learn. 28 (1) (1997) 7–39.