Mechanical Systems and Signal Processing 210 (2024) 111068

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

Federated learning with uncertainty-based client clustering for fleet-wide fault diagnosis

Hao Lu [a], Adam Thelen [b], Olga Fink [c], Chao Hu [d,*], Simon Laflamme [e]

[a] Department of Electrical Engineering, Iowa State University, Ames, IA 50011, United States of America
[b] Department of Mechanical Engineering, Iowa State University, Ames, IA 50011, United States of America
[c] Intelligent Maintenance and Operations Systems, EPFL, Lausanne, 12309, Switzerland
[d] Department of Mechanical Engineering, University of Connecticut, Storrs, CT 01776, United States of America
[e] Department of Civil, Environmental, and Construction Engineering, Iowa State University, Ames, IA 50011, United States of America

ARTICLE INFO

Communicated by Y. Lei

Keywords:
Federated learning
Fault diagnosis
Deep learning
Uncertainty quantification
Bearings

ABSTRACT

Operators from various industries have been pushing the adoption of wireless sensing nodes for industrial monitoring, and such efforts have produced sizeable condition monitoring datasets that can be used to build diagnosis algorithms capable of warning maintenance engineers of impending failure or identifying current system health conditions. However, single operators may not have sufficiently large fleets of systems or component units to collect sufficient data to develop data-driven algorithms. One potential solution to overcome the challenge of having limited representative datasets is to merge datasets from multiple operators with the same type of assets. However, directly sharing data across the company's borders yields privacy concerns. Federated learning (FL) has emerged as a promising solution to leverage datasets from multiple operators to train a decentralized asset fault diagnosis model while maintaining data confidentiality. However, the performance of traditional FL algorithms degrades when local clients' datasets are heterogeneous. The dataset heterogeneity is particularly prevalent in fault diagnosis applications due to the high diversity of operating conditions and system configurations. To address this challenge, this paper proposes a novel clustering-based FL algorithm where clients are clustered based on their dataset similarity. Estimating dataset similarity between clients without explicitly sharing data is achieved by training probabilistic deep learning models and having each client examine the predictive uncertainty of the other clients' models on its local dataset. Clients are then clustered for FL based on relative prediction accuracy and uncertainty. Experiments on three bearing fault datasets, two publicly available and one newly collected for this work, show that our algorithm significantly outperforms FedAvg and a cosine similarity-based algorithm by 5.1% and 30.7% on average over the three datasets. Further, using a probabilistic classification model has the additional advantage of accurately quantifying its predictive uncertainty, which we show it does exceptionally well.

1. Introduction

The increased availability of sensor data from fleets of cloud-connected assets, such as vehicles and manufacturing facilities, has been driving a transformation in system health monitoring for fault diagnosis. Plant operators and process engineers are interested

* Corresponding author.
E-mail addresses: hlu1@iastate.edu (H. Lu), acthelen@iastate.edu (A. Thelen), olga.fink@epfl.ch (O. Fink), chao.hu@uconn.edu (C. Hu), laflamme@iastate.edu (S. Laflamme).

https://doi.org/10.1016/j.ymssp.2023.111068
Received 25 April 2023; Received in revised form 19 November 2023; Accepted 20 December 2023

in leveraging their data to proactively diagnose faulty equipment and notify maintenance personnel of impending potential failures so they can schedule accordingly, improving reliability and reducing downtime costs in the process [1,2]. To facilitate this effort, numerous data-driven diagnosis algorithms have been developed, enabling real-time detection and accurate classification of system faults [3,4]. Data-driven fault diagnosis methods rely on machine learning and deep learning models to effectively classify condition monitoring signals that indicate the system's health status. Numerous fault diagnosis models have been proposed previously, including artificial neural networks [5–7], random forests [8,9], and support vector machines [10,11]. In general, the performance of data-driven diagnosis models is directly related to the quality and quantity of available training data [1]. However, the fault patterns that data-driven models are trained to recognize are often unique to the specific system they were collected from and are highly influenced by the operating conditions. This presents a significant challenge to model development since collecting training data under all possible working conditions is time-consuming and costly, which impedes the deployment of data-driven diagnosis models to the field [12]. A potential solution to tackle this challenge could be compiling data from multiple clients into a central database for training. Most clients hesitate to do so because of data privacy concerns and legal regulations [13]. In turn, there is a great need to develop decentralized machine learning algorithms that maintain data privacy while providing greater fault classification accuracy than asset operators can achieve individually [14].

In 2017, McMahan et al. [15] developed the first federated learning (FL) strategy for training a centralized deep learning model to serve multiple clients without explicitly requiring them to share their data, thus preserving privacy, referred to as federated averaging (FedAvg). Several researchers have effectively applied FedAvg and similar methods to train fault diagnosis models [16–19]. Notably, Zhang et al. [18] demonstrated a dynamic validation and self-supervised FedAvg algorithm where the server evaluated local models using a validation dataset to determine which models to neglect during federation. A follow-up work by Zhang et al. [19] introduced blockchain technology to improve the security of model parameter sharing. While FedAvg has shown promise in cross-client fault diagnosis, challenges arise when clients' data are heterogeneous or non-independent and identically distributed (non-IID) [20–22].

The performance of FL algorithms can be affected by two types of statistical data heterogeneity. The first, known as *domain shift*, occurs when clients' data do not share a similar distribution due to differences in system operating conditions or system configurations. The second, referred to as *label heterogeneity*, occurs when clients' datasets have different quantities and types of system faults [23]. In most real-world scenarios, clients' data generally do not exhibit similar distributions, fault types, and fault prevalence due to the diverse operating conditions experienced by systems or components in the field (e.g., run-time, load, temperature, and humidity) [14]. Unfortunately, large variations among clients' datasets have been shown to reduce the training speed and overall accuracy of the FedAvg algorithm [21,24]. One potential solution to deal with data heterogeneity is to develop a personalized feature extractor network that extracts fault-related features that are invariant to the different operating conditions and fault types of the clients (see Fig. 1b) [25–27]. This approach, referred to as adaptation-based FL, mitigates the data domain shift using unique model architectures that extract features that are indistinguishable between different clients. Another approach to deal with data heterogeneity is to group clients for training based on their dataset similarity, where each group builds a different global model (see Fig. 1c) [28–31]. The key to this approach lies in the clustering strategy used to group the clients since directly comparing clients' datasets would violate data privacy. For example, Tian et al. [31] utilized the cosine similarity between each pair of client model parameters to infer the similarity of client dataset distributions.

While both adaptation-based FL and client clustering are effective approaches to dealing with client data heterogeneity, numerous distinct challenges specific to system fault diagnosis must be resolved. *First*, due to large differences in operating settings, clients may experience significant differences in the frequencies of faults occurring, and fault types observed by some clients may not have been observed by other clients. This results in high heterogeneity among clients' datasets, denoted as open-set label shifts in this study [32]. Only a few studies have considered such data heterogeneity. For example, Zhang et al. [18] validated the performance of their dynamic validation and self-supervised FedAvg algorithm with three experimental settings: IID, non-IID-class, and non-IID-domain. For the non-IID-class setting, each client's data contains two health classes: healthy and faulty (one faulty class). For the non-IID-domain setting, each client's data contains all the health classes but is collected under different operating conditions. Similarly, Mehta et al. [33] tested the performance of FedAvg for non-IID datasets. However, these two studies did not consider complex scenarios with client data heterogeneity with respect to both the sample label distribution and operating conditions. *Second*, the performance of fault diagnosis models highly depends on the validity of the assumption that the training and test data follow similar distributions. However, in real-world scenarios, this assumption might be invalid [34,35]. For example, changes in operating conditions could make test data significantly different from training data, and deterministic machine learning models that do not consider predictive uncertainty are typically incapable of detecting these differences, which might lead to largely incorrect diagnosis results on distribution-shifted data [36,37]. Making incorrect predictions on out-of-training-distribution (or simply out-of-distribution) samples without conveying model prediction confidence may lead to inappropriate health management decisions and could even affect production safety. Therefore, it is often more desirable to train probabilistic models capable of accurately estimating their predictive uncertainty and confidence on a per-sample basis.

In this work, we propose a novel FL framework to train fault diagnosis models using local datasets of multiple clients with privacy protection in the presence of distribution shifts and open-set label shifts among these local datasets. Unlike existing algorithms that use model parameters or prediction errors to cluster clients, this work adopts a probabilistic deep learning algorithm for client clustering, and each local model's predictive uncertainty towards other clients' datasets is used to cluster clients. During the model evaluation, the ability to quantify the predictive uncertainty on a per-sample basis allows clients to discern out-of-distribution samples. Considering the scenario where a client's model faces out-of-distribution samples, a model selection strategy is developed to enable the client to select the most suitable model from those in the server (i.e., the model that achieves the least predictive uncertainty).

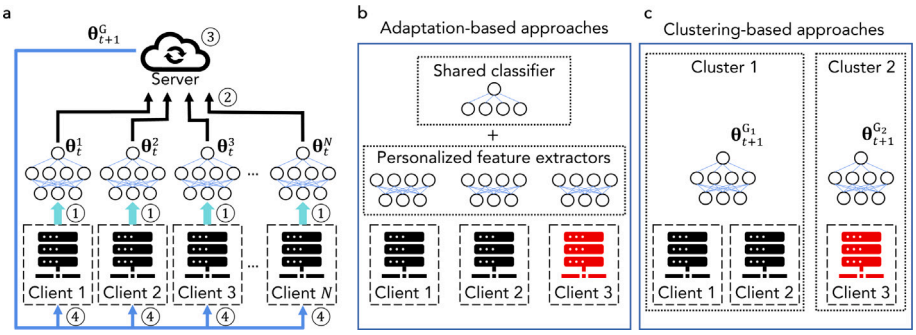

Fig. 1. An illustration of FL (**a**) and two FL strategies (**b**, **c**). **a**, The four-step process of FL is common to nearly all FL approaches. **b**, An overview of an adaptation-based approach to FL that uses a global classifier to accommodate client-specific feature extractors. **c**, An overview of a clustering-based approach to FL that groups clients for training based on non-private comparison criteria.

We benchmark the fault classification performance of our FL algorithm (FedSNGP) against FedAvg and a cosine similarity-based clustering algorithm FedCos, adapted from the algorithm proposed by the authors in Ref. [31] using two publicly-available bearing fault datasets and one new bearing fault dataset collected specifically for this work. In cases where clients' datasets are heterogeneous in fault type and exhibit significant domain shift (Scenario 2), our FedSNGP algorithm outperforms FedAvg and FedCos. Further, in cases where clients' datasets exhibit domain shift and are heterogeneous in fault type as well as the sample size (Scenario 3), our FedSNGP algorithm demonstrates its prowess, achieving significantly higher accuracy than both FedAvg and FedCos. This final scenario highlights the advantage of our self-supervised and uncertainty-aware clustering algorithm for FL, as it achieves excellent fault classification accuracy, even in the most extreme scenarios. Further, using a probabilistic classification model has the added benefit of quantifying its predictive uncertainty to detect out-of-distribution samples, which we show it does exceptionally well.

2. Preliminaries on FL

Data-driven fault diagnosis has emerged recently and shown promising results in many studies. However, when it comes to industrial applications, it is often impractical for a single client to collect large volumes of high-quality data, without which the performance of data-driven models may likely be compromised. FL was developed to address this challenge by allowing multiple clients to train more accurate models without sharing the clients' local datasets. This study targets industrial FL scenarios where each operator has its own dataset and shares the same diagnosis task. To stay consistent with existing literature on FL, we refer to operators as clients. We begin by defining notations used in FL studies. We denote the number of clients as N and the ith client's local training dataset as $D^i = \left\{ \left(\mathbf{x}_j^i, y_j^i \right) \right\}_{j=1}^{n_i}$, where this client has n_i input–output training pairs, with the jth training pair being $\left(\mathbf{x}_j^i, y_j^i \right)$. Each client aims to optimize its model parameters θ^i to minimize the global loss $\sum_{i=1}^{N} \sum_{j=1}^{n_i} Loss\left(\theta^i, \mathbf{x}_j^i, y_j^i \right)$. Here, $Loss\left(\theta^i, \mathbf{x}_j^i, y_j^i \right)$ represents the prediction loss of the machine learning model with parameters θ^i on a given input–output pair $\left(\mathbf{x}_j^i, y_j^i \right)$ [38,39].

One simplistic approach is to allow the server to aggregate all of the client datasets and optimize the model parameters by computing the gradient of the global loss. This implies that the server has complete access to all clients' datasets. However, aggregating sensitive client data on a central server presents privacy and legal concerns and should be avoided. Instead, FL strategies aim to train a global model or models without sharing data between clients. To perform FL, as illustrated in Fig. 1a, each client independently optimizes their local model using their own dataset (Step 1) and transmits their model parameters to the server (Step 2). On the server side, model parameters are aggregated to create one or more global models (Step 3), which are subsequently disseminated to the clients (Step 4) [15]. A single instance of this four-step process is often called a *communication round*, as it involves the clients communicating with the server to facilitate training.

The most important step in any FL strategy is the model aggregation (Step 3 in Fig. 1a). Model aggregation leverages non-sensitive client information (e.g., model parameters and other information the client is willing to share) as part of an algorithm to update the global model. The model aggregation algorithm directly affects the amount of information transferred from each client to the final model and is crucial to the model's overall accuracy. The first model aggregation strategy, proposed by McMahan et al. [15], was the FedAvg strategy. The FedAvg method of model aggregation combines the client's models by performing a weighted average of model parameters, where the weights are determined based on the number of training samples each client has. The weighted average model aggregation is defined as follows:

$$\theta_{t+1} = \frac{\sum_{i=1}^{N} n_i \theta_t^i}{\sum_{i=1}^{N} n_i} \tag{1}$$

The weighted averaging approach to model aggregation used in FedAvg allows clients to employ a collaborative prediction model that contains additional information beyond what each client had separately. This is because each client's model can be regarded as a condensed representation of their data, and constructing a global model by incorporating the weights of all clients enables the

global model to learn each client's information indirectly. As a result, this yields an exceptionally accurate model for the clients. Studies have shown that when each client has a limited amount of training data, adopting the concept of FL could produce more accurate models than models that are trained locally [13,18]

3. Related works

Though FedAvg yields decent performance in most scenarios, a major challenge of the FedAvg approach is that its performance decreases significantly when the clients' data are vastly different from each other, i.e., non-IID and heterogeneous in fault types [12,21]. Large differences in local dataset distributions (often referred to as domain shifts) cause some clients' model parameters to deviate significantly from those of the group. Including information models from clients with significantly different dataset distributions can negatively impact the global model aggregation process, causing the final global model to serve neither the majority of clients with similar data nor the minority clients with outlier data well. Numerous algorithms have been developed to tackle the challenge of data heterogeneity when deploying an FL strategy for a fleet of system or component units. This section introduces two of the most common methods for dealing with non-IID datasets and highlights notable research in this area.

3.1. Adaptation-based approaches to FL

The challenge of domain shift has been widely recognized in the field of deep learning [40]. One common method of dealing with domain shift in traditional deep learning settings is known as adaptation-based learning [23,41]. There are different domain adaptation approaches. One way is to train a feature extractor that extracts domain-invariant features from the input data. Then, feed the domain invariant features to a classifier for prediction.

The same domain adaptation approach can be applied to the FL problem. In adaptation-based FL, shown in Fig. 1b, the ith client's model is divided into a personalized feature extractor and a shared global classifier [23,27]. The personalized feature extractor learns to map the raw input data into a low-dimensional feature space such that the learned mapping is invariant to the system's operating conditions. Then, the shared classifier takes the extracted domain-invariant features as input to diagnose the system's health. Mathematically, this optimization process can be written as follows:

$$\underset{\varphi_1,\varphi_2,\ldots,\varphi_N,h_g}{\operatorname{argmin}} \sum_{i=1}^{N}\sum_{j=1}^{n_i} Loss\left([\varphi_i,h_g],\mathbf{X}_j^i,y_j^i\right)+\alpha\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} Dis\left(z_i,z_j\right) \tag{2}$$

where φ_i and h_g denote the parameters of the feature extractor and those of the global classifier, respectively, $Loss\left([\varphi_i,h_g],\mathbf{X}_j^i,y_j^i\right)$ denotes the ith model's prediction loss towards the local training dataset, z_i denotes the client i's extracted features, and $Dis(\cdot)$ is the metric of discrepancy of two distributions.

Adaptation-based FL can be achieved by introducing a domain classifier into the model architecture [27] or the maximum mean discrepancy distance into the loss function [42]. Additionally, researchers also investigated alternatives to these transfer learning-based FL frameworks. For example, Ran et al. [43] stated that in transfer learning scenarios, some source-domain clients might have mostly low-quality data (e.g., mislabeled data or data that significantly differs from the target data distribution) and developed a dynamic model aggregation algorithm to mitigate the influence of these low-quality clients. Similarly, Chen et al. [42] developed a discrepancy-based weighted averaging algorithm targeting a unique FL setup where a new client (considered the target domain client) with unlabeled data may join the existing clients (considered the source domain clients) in FL.

While domain adaptation algorithms are a popular solution for dealing with data domain shifts, their main limitation is their lack of flexibility to changes in future operating conditions. Adaptation-based approaches assume that the data distribution and operating conditions will not change with time. If any aspect of the system changes, the input data will shift, and the personalized feature extractor cannot be trusted to accurately map the new inputs into domain-invariant features. To alleviate this challenge, test-time adaptation algorithms have been developed. Regardless, domain adaptation algorithms remain promising to deal with domain shifts in FL environments.

3.2. Clustering-based approaches to FL

Research conducted in Refs. [13,18] has highlighted that, in certain non-IID FL scenarios, certain clients should be assigned lower weights or even disregarded during model aggregation to alleviate the negative effect of client data heterogeneity. Clustering-based FL approaches generate client clusters, and within each cluster, the models are aggregated to create a combined model [44]. During each training communication round of clustering-based FL, the server performs client clustering that assigns clients with comparable data distributions to the same cluster, whereby the clients in the same cluster share the same model parameters. The optimization target of the clustering strategy and model parameter optimization can be formulated as:

$$\begin{aligned}&\underset{C_1,C_2,\ldots,C_k}{\text{minimize}}\ \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N} r_{i,k}\sum_{j=1}^{n_i} Loss\left(C_k,\mathbf{X}_j^i,y_j^i\right)\\ &\text{subject to } r_{i,k}=\underset{r_{i,k}}{\operatorname{argmin}}\ \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N} r_{i,k}\left\|\mathbf{D}^i-\Omega_k\right\|\end{aligned} \tag{3}$$

Where C_k denotes the model parameters of the clients in kth cluster, $r_{i,k}$ denotes the resulting client clustering with $r_{i,k} = 1$ if client $i \in$ cluster k, otherwise $r_{i,k} = 0$. The $\mathbf{D}^i$ denotes client dataset distribution, and Ω_k represents the data distribution center for cluster k. It is important to note that in order to maintain data privacy, the server must indirectly deduce dataset similarity ($\|\mathbf{D}_i - \Omega_k\|$).

Studies have shown that for non-IID datasets, clustering-based FL methods outperform the traditional FL methods such as FedAvg and FedProx [18,29,31]. The key to clustering-based FL algorithms is to estimate the dataset similarity between each pair of clients without accessing the client's dataset. Tian et al. [31] developed a similarity-based clustering approach that used the cosine distance between models' parameters to estimate their dataset similarity and determine if they should be clustered together for training. In a similar line of research, Li et al. [29] proposed a soft clustering algorithm that can assign clients to overlapped clusters. In this way, each client's information is fully utilized.

For clustering-based FL algorithms, the clustering strategy plays a crucial role in determining the algorithm's performance. If a client is allocated to an unsuitable cluster, the resulting global model may perform worse than the more basic algorithms, like FedAvg, that incorporate all models in the federation process regardless of data domain shift and label heterogeneity. Altogether, several clustering-based FL approaches have been shown to be effective solutions to the challenge of domain shift and label heterogeneity [31,45,46]. However, little work has been done to apply these methods in the field of system fault diagnosis.

3.3. FL for fault diagnosis

This section summarizes recent FL studies that address data heterogeneity challenges for machinery fault diagnosis. Due to diverse operating conditions and system configurations, datasets for system fault diagnosis are often non-IID and unbalanced [47]. The data heterogeneity poses unique challenges that require developing and validating more advanced FL algorithms.

Adaptation-based approaches are developed by training customized models to accommodate client-specific datasets. Wang et al. [48] proposed an adversarial domain generalization-based FL framework where clients collaborate to train a global classifier, and a personalized feature extractor is trained for each client to extract domain-invariant features. A similar idea was pursued by Lin et al. [49] in their study on hierarchical model parameters aggregation. In their approach, the model layers are divided into initial and higher layers, where initial layers are shared among clients, and the higher layers serve as personalized classifiers for each client. Cong et al. [50] attempted to guide the feature extractor to extract domain-invariant features by adding the maximum mean discrepancy term to the loss function. A notable outcome was an increase in the similarity among the feature sets extracted from different clients. Zhao et al. [51] developed a global feature alignment module based on multiple kernel maximum mean discrepancy. Li et al. [52] developed a two-stage transfer learning-based FL framework. In their proposed method, the source-domain clients first generate fake samples, and then a prediction alignment method is proposed to achieve knowledge transfer without using source data.

All the methods described above leverage transfer learning techniques for domain adaptation and have each client contribute approximately equally to the final model. However, it may not always be optimal to have equal contributions from clients toward model building. Researchers made efforts to tackle this issue by investigating various weighting schemes for averaging over clients. Zhang et al. [18] proposed a dynamic validation and self-supervision FL framework. In their approach, the server selects subsets of uploaded client models to generate multiple updated models, and then the updated model with the highest validation accuracy is selected as the global model for the next FL communication round. Li et al. [47] adopted an improved particle swarm optimization algorithm to optimize the weight of each client model. One limitation of those studies [18,47] is they require a model validation dataset to be split out of the training dataset. Chen et al. [13] developed a similarity-based dynamic weighted averaging FL approach, where the server assigns higher weights to clients' models that are more similar to the global model during model aggregation. Geng et al. [53] included the local models' recall rate and precision values and developed an improved weighted FL algorithm. Those studies leverage a group of clients' information to train a more powerful model for the target client/task. However, in these approaches, client models whose training data distribution is different from the target data distribution are assigned lower weights than other models. In that way, those FL approaches ignore some clients' information. Although the developed model performs well for the target client/task, that model may not be applicable to the clients assigned with lower weights.

Currently, limited research focuses on clustering clients into groups, each with a separate and sometimes independent FL run. Gholizadeh et al. [45] developed a clustering-based algorithm where each client performs a grid Search with cross validation to tune their model hyperparameters. Then, during model aggregation, the clients that share similar hyperparameters are clustered into the same group to get updated models. Li et al. [28] proposed a multi-center clustering-based FL algorithm. They used a k-means-based algorithm to cluster clients. However, the drawback of this approach is that it requires specifying the number of clusters in advance, which can significantly affect performance.

4. Proposed FL method

Our approach to FL differs from existing clustering-based FL approaches in two important ways. *First*, the proposed method incorporates a distance-aware model into the FL framework, which enables us to use the predictive uncertainty to infer the dataset similarity and group clients into clusters. The distance-aware model employed in the proposed FL approach has a unique capability of detecting out-of-distribution input samples, thus preventing the model from making overconfident predictions. *Second*, a clustering algorithm that can determine the number of clusters without requiring user-defined parameters is used for client clustering. The number of clusters is dynamically adjusted by the clustering algorithm during each communication round based on the dataset similarity inferred by the distance-aware model. This ensures that the client grouping is optimized to achieve the maximum

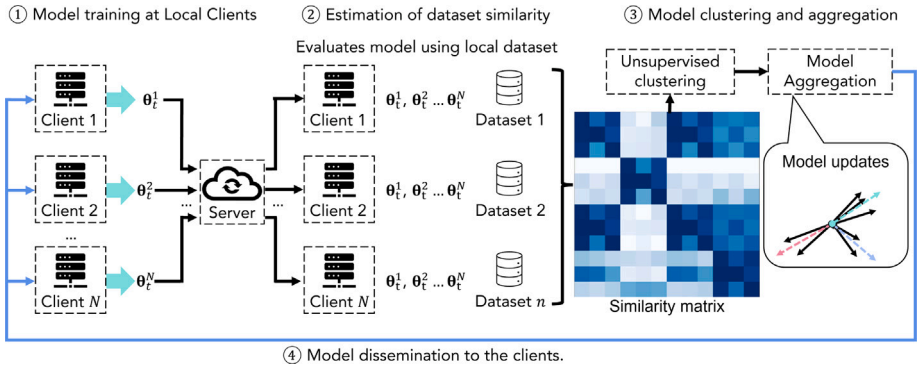

Fig. 2. An illustration of the FedSNGP training process. Each training communication round has four key steps: (1) local model training, (2) dataset similarity estimation, (3) local model clustering for training, and (4) model dissemination to the clients.

information transfer to the global model without the need for predefining the number of clusters. Our algorithm is designed to work well under a wide range of dataset conditions with various levels of domain shift and missing fault types.

Shown in Fig. 2, our FL training strategy consists of four steps:

1. Local model training: each client trains its own model using its dataset and uploads model parameters to the server.
2. Estimation of dataset similarity: each client downloads the models of other clients from the server and estimates the predictive uncertainty of downloaded models towards its dataset to infer distribution similarity between each pair of client datasets.
3. Model clustering and aggregation: the server forms a similarity matrix based on the estimated dataset similarity, uses an unsupervised clustering algorithm to group the clients into clusters, and performs model aggregation within each cluster.
4. Model dissemination to the clients: the server sends the updated global model back to the clients for further local training.

In what follows, we describe our uncertainty-aware deep learning model and our dynamic clustering algorithm for FL.

4.1. Uncertainty-aware model training at local clients

Deploying a deep learning model in the field requires building confidence among operators and maintenance personnel regarding the model's diagnostic accuracy. At a minimum, the model should indicate higher predictive uncertainty to operators when it lacks confidence in a prediction. This is particularly important for safety-critical equipment, as an incorrect diagnosis can result in catastrophic failures, putting people and equipment at risk. For this reason, it is essential to build a fault diagnosis model that can accurately estimate its predictive uncertainty. Moreover, the uncertainty estimates can be leveraged to improve the performance of our FL approach.

Real-world fault diagnosis is prone to challenges such as domain shifts caused by changes in operating conditions or other factors. With a slight shift in input distribution, a vanilla machine learning model may yield overconfident prediction results for these samples, misleading operators and wasting time. To tackle this problem, it is desirable to develop an algorithm that can infer when new data exhibit domain shift or unknown fault type with respect to the training dataset. The larger the discrepancy between the new sample and the model's training data, the higher the model's predictive uncertainty should be. To accomplish this, the proposed method employs a distinctive model architecture that enables the model to identify input samples that differ significantly from the data it was trained on and consequently produce high uncertainty estimates on these input samples.

The ability of the model to detect the difference between new samples and the training dataset is referred to as 'distance awareness' [54]. Given a deep learning model $logit(\mathbf{x}) = g \circ h(\mathbf{x})$, where g represents the model output layer and $h(\cdot)$ denotes the mapping function that maps the raw input into a hidden representation space, a deep learning model is input distance aware if it follows two conditions:

1. The model's mapping function is distance-preserving. Considering the distance in the data manifold $\|\mathbf{x} - \mathbf{x}'\|_X$, the distance in the hidden space $\left\|h(\mathbf{x}) - h(\mathbf{x}')\right\|_H$ needs to satisfy the following bi-Lipschitz condition:

$$L_1 \times \|\mathbf{x} - \mathbf{x}'\|_X \leq \left\|h(\mathbf{x}) - h(\mathbf{x}')\right\|_H \leq L_2 \times \|\mathbf{x} - \mathbf{x}'\|_X \ , \ 0 < L_1 < L_2 \tag{4}$$

2. The model's output layer is distance aware. The output layer g should output an uncertainty metric that reflects distance in the hidden space $\left\|h(\mathbf{x}) - h(\mathbf{x}')\right\|_H$. The standard Gaussian process machine learning model exhibits this property. However, Gaussian processes are unsuitable for high-dimensional problems and large datasets because the inference is extremely computationally expensive ($\mathcal{O}(n^3)$) [55].

To address the limitations mentioned above, Liu et al. [54] proposed a model architecture known as Spectral-normalized Neural Gaussian Process (SNGP). Fig. 3 shows how SNGP differs from a standard deep residual network. SNGP is formed by residual blocks

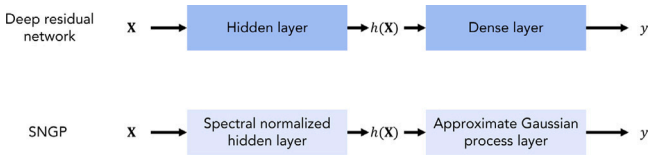

Fig. 3. Architecture comparison between a standard deep residual network and SNGP. SNGP adds the distance-awareness ability to the deep residual network by (1) applying spectral normalization to hidden weights and (2) replacing the last output layer with a Gaussian process.

Algorithm 1 Local training of SNGP

Inputs:
  Local training dataset $D^i = \left\{\left(\mathbf{x}_j^i, y_j^i\right)\right\}_{j=1}^{n_i}$
Initialize:
  $\hat{\Sigma}_k = \mathbf{I}$, $\mathbf{w}_L \sim N(0,1)$, $\mathbf{b}_L \sim U(0, 2\pi)$
for Number training = 1 to Max iteration **do**
  SGD update $\boldsymbol{\beta}, \{\mathbf{w}_l\}_{l=1}^{L-1}, \{\mathbf{b}_l\}_{l=1}^{L-1}$
  if final epoch **then**
    Update precision matrix $\hat{\mathbf{H}}$
  end if
end for
Compute posterior covariance $\hat{\mathbf{H}}^{-1}$

Algorithm 2 SNGP prediction

Inputs:
  Test sample $\mathbf{x}$
Initialize:
  SNGP model $\left\{\boldsymbol{\beta}, \{\mathbf{w}_l, \mathbf{b}_l\}_{l=1}^{L-1}, \hat{\mathbf{H}}^{-1}\right\}$
Compute Features:
  $\Phi_{\mathbf{D}_L \times 1} = \sqrt{\frac{2}{D_L}} * cos\left(-\mathbf{w}_L h(\mathbf{x}_i) + \mathbf{b}_L\right)$
Compute Posterior Mean:
  logit(x)= $\Phi^\intercal \beta$
Compute posterior variance:
  var(x)= $\Phi^\intercal \hat{\mathbf{H}}^{-1} \Phi$
Evaluate predictive posterior distribution:
  Softmax $\left(\frac{\Phi^\intercal \boldsymbol{\beta}}{\sqrt{1+\lambda * \Phi^\intercal \hat{\mathbf{H}}^{-1} \Phi}}\right)$

Fig. 4. Pseudo-code of the training and prediction process of the SNGP model.

equipped with spectral normalization to ensure that the hidden mapping $h(\cdot)$ is distance preserving. For the model's output layer, the dense layer is replaced with an approximate Gaussian process conditioned on the learned hidden representations of the model. adopts random feature expansion to convert the infinite-dimensional Gaussian process into a featured Bayesian linear model and uses Laplace approximation to approximate the posterior. The output layer is designed with fixed hidden weights $\mathbf{w}_L$ and trainable output weights $\boldsymbol{\beta}$, given as:

$$g(\mathbf{h}) = \sqrt{\frac{2}{D_L}} cos\left(-\mathbf{w}_L \mathbf{h} + \mathbf{b}_L\right)^\intercal \boldsymbol{\beta}, \text{with prior } \boldsymbol{\beta}_{D_L \times 1} \sim N\left(0, \mathbf{I}_{D_L \times D_L}\right) \tag{5}$$

where $\mathbf{h}$ denotes the output features of the penultimate layer with dimension D_{L-1}, $\mathbf{w}_L$ is a fixed weight matrix with dimension $D_L \times D_{L-1}$ whose entries are sampled i.i.d. from $\sim N(0,1)$. $\mathbf{b}_L$ is a fixed bias term with dimension $D_L \times 1$ whose entries are sampled i.i.d. from $Uniform(0, \pi)$. Conditional on $\mathbf{h}$, $\boldsymbol{\beta}$ is the only trainable variable in the output layer. For K-class classification, the Laplace method is applied to approximate the posterior for each class as [54]:

$$p(\boldsymbol{\beta}_k \mid D) \approx MVN\left(\hat{\boldsymbol{\beta}}_k, \hat{\Sigma}_k = \hat{\mathbf{H}}_k^{-1}\right), \text{where } \hat{\mathbf{H}}_k = \sum_{i=1}^{N} p_{i,k}\left(1 - p_{i,k}\right) \boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^\intercal + \mathbf{I} \tag{6}$$

and $\hat{\boldsymbol{\beta}}_k$ is the model's maximum a posteriori probability estimate conditioned on the random Fourier hidden representation, $p_{i,k} = 1/(1 + e^{-\boldsymbol{\Phi}_i^\intercal \hat{\boldsymbol{\beta}}_k})$, and $\boldsymbol{\Phi}_i = \sqrt{\frac{2}{D_L}} * cos\left(-\mathbf{w}_L h(\mathbf{x}_i) + \mathbf{b}_L\right)$.

Training the SNGP model can be performed using stochastic gradient descent:

$$-\log p\left(\boldsymbol{\beta}, \{\mathbf{w}_l, \mathbf{b}_l\}_{l=1}^{L-1} \mid \mathcal{D}\right) = -\log p\left(\mathcal{D} \mid \boldsymbol{\beta}, \{\mathbf{w}_l, \mathbf{b}_l\}_{l=1}^{L-1}\right) + \frac{1}{2}\|\boldsymbol{\beta}\|^2 \tag{7}$$

where $-\log p\left(\mathcal{D} \mid \boldsymbol{\beta}, \{\mathbf{w}_l, \mathbf{b}_l\}_{l=1}^{L-1}\right)$ is the cross-entropy loss for the classification task. The pseudo-code of the training and test procedure of the SNGP model is summarized in Fig. 4. A good tutorial of SNGP can be found in Ref. [56]. Each client trains a local model using the local dataset and uploads the model to the server (see Fig. 2).

Algorithm 3 FedSNGP

Initialize:
  Model parameters for all clients $\tilde{\theta}_0$
for Communication round t= 1 to Max number of training round **do**
  Initialize prediction variance matrix $\mathrm{M}_{\mathrm{sim}} = Zeros\,(n, n)$
  Receive local updates: $\tilde{\theta}_t^1, \tilde{\theta}_t^2, \ldots, \tilde{\theta}_t^n$
  for Local client $i \in \{1, 2, \ldots, n\}$ **do**
    for Local dataset $j \in \{1, 2, \ldots, n\}$ **do**
      Model$^j \leftarrow \tilde{\theta}_t^j$
      Calculate $\mathrm{M}_{\mathrm{sim}}(i, j)$ by using Model^j's predictive uncertainty towards $\mathbf{D}^i$
      Clustering List=AffinityPropagation($\mathbf{M}_{\mathrm{sim}}$)
    end for
  end for
  Aggregate mode based on the Clustering List
  Update local model parameters $\tilde{\theta}_{t+1}^1, \tilde{\theta}_{t+1}^2, \ldots, \tilde{\theta}_{t+1}^n$
end for

Fig. 5. Pseudo-code of FedSNGP algorithm.

4.2. Dataset similarity estimation for federated clustering

When dealing with client datasets that are heterogeneous, the typical methods of FL strategies may result in suboptimal performance. One way to alleviate the negative impact caused by data heterogeneity is to cluster clients [57]. Clustering-based FL algorithms group clients into clusters based on their dataset similarity. The SNGP model is capable of inferring the distance between the training dataset and test samples. Therefore, SNGP model can be adopted to infer the similarity between the datasets of two clients. The FedSNGP algorithm involves uploading the locally trained model from each client to the server after training. Following this, the clients work together to estimate the similarities between their datasets.

As the SNGP model is distance aware, the model's prediction result reflects the similarity between the test sample and the training datasets. To infer client dataset similarity, each client downloads other clients' models and evaluates other models' prediction results against the local training dataset. The SNGP model $\left\{\boldsymbol{\beta}, \left\{\mathbf{w}_l, \mathbf{b}_l\right\}_{l=1}^{L-1}, \hat{\mathbf{H}}^{-1}\right\}$ maps the input $\mathbf{x}$ into random Fourier feature $\boldsymbol{\Phi} = \sqrt{\frac{2}{D_L}} cos\left(\mathbf{w}_L h\left(\mathbf{x}\right) + \mathbf{b}_L\right)$, and the output posterior predictive probability for each class follows $MVN\left(\boldsymbol{\Phi}^\intercal \boldsymbol{\beta}_k, \boldsymbol{\Phi}^\intercal \hat{\mathbf{H}}_k^{-1} \boldsymbol{\Phi}\right)$, the expectation of the probability can be approximated by using the mean-field method:

$$E\left(p\left(\mathbf{x}\right)\right) \approx \mathrm{Softmax}\left(\frac{\boldsymbol{\Phi}^\intercal \boldsymbol{\beta}_k}{\sqrt{1 + \lambda * \boldsymbol{\Phi}^\intercal \hat{\mathbf{H}}_k^{-1} \boldsymbol{\Phi}}}\right) \tag{8}$$

where λ is a scaling factor that is commonly set to $\pi/8$ [54,58]. Then, in this study, the predicted variance is used to infer model uncertainty:

$$P_{var} \approx \frac{1}{n_i} \sum_{i=1}^{n_i} var\left(p\left(\mathbf{x_i}\right)\right) = \frac{1}{n_i} \sum_{i=1}^{n_i} E\left(p\left(\mathbf{x_i^2}\right)\right) - \left(E\left(p\left(\mathbf{x}_i\right)\right)\right)^2 \tag{9}$$

The predicted variance reflects the distance between a model's training and test datasets, where higher variance indicates a higher difference between two datasets. In the proposed FedSNGP approach, each client downloads all the local models and evaluates those models using its local training dataset. Then, the predicted variance results are uploaded to the server and are used for clustering. The predicted variance is used to form a 2D matrix, where the entry in ith row, jth column is the predicted variance from evaluating the jth model using the ith client's dataset. After performing column-wise normalization, the results form a predictive uncertainty matrix with the range $[0, 1]$, with dimension $N_{client} \times N_{client}$, denoted as M_U.

In the next step, the affinity propagation algorithm [59] is used to automatically determine the number of clusters. Unlike the k-means clustering algorithm, affinity propagation does not require the user to specify the number of clusters in advance. Instead, it automatically determines a suitable number of clusters. We create a similarity matrix M_{Sim} by calculating $1 - M_U$. The similarity matrix represents the similarity between clients' datasets, where values close to 0 indicate high predictive uncertainty and suggest that the jth client's dataset differs significantly from the ith client's dataset, and the clients would not benefit from federating together. The affinity propagation takes the similarity matrix as input and outputs the clustering strategy for model aggregation. The clustering strategy takes place remotely on the central server. A pseudo-code describing this process is provided in Fig. 5, and a code implementing FedSNGP in Python can be accessed at https://github.com/SalieriLu/FL_fault_diagnosis.

5. Training and test configuration

5.1. Selected datasets

We evaluate the performance of the proposed algorithm on three bearing fault diagnosis case studies to demonstrate the effectiveness of the FedSNGP algorithm. A summary of three case studies is given in Table 1. These case studies are designed

Table 1
Summary of selected bearing datasets.

Name	Bearing health conditions	Cause of bearing faults	Sampling frequency	Other important quantities			
CWRU	Healthy, Inner race fault, Outer race fault	Electric engraver	12 kHz	Operating condition ID	Shaft speed	Bearing fault diameter	
				1	1797 rpm	0.007"	
				2	1797 rpm	0.014"	
				3	1772 rpm	0.007"	
				4	1772 rpm	0.014"	
				5	1730 rpm	0.007"	
				6	1730 rpm	0.014"	
PU	Healthy, Inner race fault, Outer race fault	Accelerated degradation	64 kHz	Operating condition ID	Shaft speed	Load torque	Radial force
				1	1500 rpm	0.7 N m	1000 N
				2	900 rpm	0.7 N m	1000 N
				3	1500 rpm	0.1 N m	1000 N
				4	1500 rpm	0.7 N m	400 N
ISU	Healthy, Inner race fault, Outer race fault, Combination of bearing faults	Electric engraver	25.6 kHz	Operating condition ID	Shaft speed	Radial force	
				1	2100 rpm	0	
				2	2100 rpm	25 N	
				3	1500 rpm	0	
				4	1500 rpm	25 N	

to simulate the data heterogeneity challenges. The dataset for each case study contains vibration signals collected from bearings with distinct health conditions under various operating conditions. To simulate the FL scenario, the dataset is partitioned into 12 subsets, and each subset is allocated to a client. The objective of each client is to train a fault diagnosis model capable of accurately identifying the health condition of bearings. The selected bearing datasets chosen for this task are as follows:

1. Case Western Reserve University (CWRU) dataset [60]: Case study 1 uses the Case Western Reserve University (CWRU) bearing dataset, which is widely used as a standard reference in the bearing diagnosis field. The data was collected with a 12 kHz sampling frequency under four working conditions. The bearing faults were introduced by using electro-discharge machining. Three manual fault sizes were created, with diameters of 0.007, 0.014, and 0.021 inches. Considering that outer race and inner race faults are more frequently appeared in accelerated degradation tests, the healthy, inner race fault, and outer race fault bearings are selected. For each type of fault, the bearings with fault diameters of 0.007 and 0.014 inches are selected.
2. Paderborn University (PU) dataset [61]: Case study 2 uses an experimental dataset collected by the KAT data center in Paderborn University (PU). The PU dataset contains high-resolution vibration data, which are collected from six healthy bearings and 26 damaged bearings under four working conditions. The sampling frequency is 64 kHz. Out of the 26 damaged bearing sets, 12 bearings were artificially damaged, and the other 14 were damaged by accelerated lifetime test. Bearings that undergo accelerated lifetime testing are selected in this case study. In total, bearings with three different health conditions (healthy, inner race fault, and outer race fault) are included, with three bearings for each health condition.
3. Iowa State University (ISU) dataset: An experimental dataset is collected from a machinery fault simulator in our lab at Iowa State University, denoted as the Iowa State University Machinery Fault Simulator (ISU-MFS) dataset, or simply the ISU dataset. The ISU dataset was collected with a 25.6 kHz sampling frequency from four bearings under four working conditions. The bearing faults were introduced by electrical discharge machining, where the size of each fault is approximately 1.5 mm × 1.0 mm × 0.1 mm. Four health conditions are considered: healthy, inner race fault, outer race fault, and a combination of faults.

The above-mentioned datasets are collected under different sampling rates and different sampling times. All three datasets are used in three separate case studies to evaluate the proposed FL method thoroughly. To allow meaningful comparisons between datasets, this study uses the same settings for data pre-processing, feature extraction, and input size across all three datasets. Keeping the model input size consistent across all datasets ensures all models have the same number of parameters and makes comparisons between datasets meaningful. We first resample vibration signals with sampling frequency = 12.8 kHz and then split each signal into multiple vibration samples. The length of each sample is equal to 1024 (i.e., each sample has a total of 1024 measurement points), equivalent to 0.08 s of vibration measurements. Frequency-domain analysis has been widely applied in vibration-based bearing diagnosis [6,62]. In this study, the fast Fourier transform is used to acquire the single-sided output of a signal's power spectrum. After signal pre-processing, each time-domain signal is converted into a frequency-domain feature vector of length 512. After dataset pre-processing, 80% of the total dataset is used to form training datasets, while the remainder is used for testing.

5.2. Design of the FL scenarios

To demonstrate the impact of dataset heterogeneity, this study considers three FL scenarios in which each client's dataset is collected under different working conditions. Moreover, due to the rareness of faults in real-world industrial equipment, each client

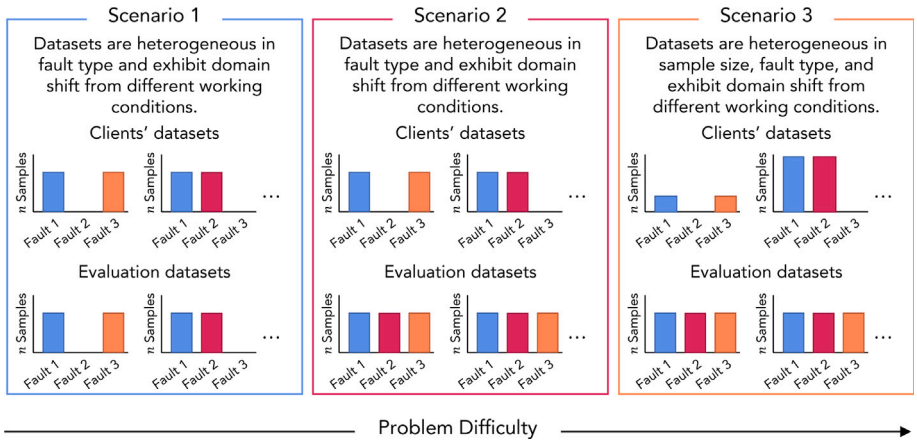

Fig. 6. An illustration of the three test conditions examined in this paper.

may not be able to observe every possible fault under all possible operating conditions. In the domain adaptation problem, the partial domain adaptation scenario is defined by making the source domain have all classes of data and the target domain cover only a subset of those classes [63]. We extend this definition to FL scenarios where certain clients' datasets contain a subset of all classes. For consistency and simplification, such scenarios are also denoted as open-set FL. The training datasets under scenarios 1 and 2 are defined using the partial domain adaptation setting with domain shift. Specifically for Scenario 3, we drastically vary the number of samples in each client's dataset to make the problem more difficult.

Fig. 6 shows the three FL scenarios considered in this work. In scenario 1, client datasets are collected under different working conditions, and differences exist in each client's label space (partial domain adaptation setting). For example, some clients might only have healthy and inner race fault samples, while other clients might have healthy and outer race fault samples. Under Scenario 1, each client's evaluation dataset follows a similar distribution as the training dataset (the types of test data are identical to the training dataset).

The training datasets in scenario 2 are identical to the settings in scenario 1. However, different from scenario 1, the test datasets are label-balanced datasets, which means clients need to collaborate to train a model that is capable of identifying unknown fault types.

Finally, for scenario 3, in addition to the feature distribution shift caused by different working conditions, the clients' datasets follow the partial domain adaptation setting with data quantity heterogeneity. The prediction difficulties are increased from scenario 1 to scenario 3.

Three test scenarios are designed for each case study. We first divide each dataset into several subsets, where each subset contains data collected under the same operating conditions. Then, we further divide each subset of data into clients' local datasets. For example, the CWRU training dataset is divided into six subsets, where each subset is further divided into two training datasets, each produced for one client. Note that the samples within the same subset are collected under the same operating conditions.

For the CWRU dataset, six operating conditions are defined, and $6\times2 = 12$ client datasets are created, each containing data related to two particular health conditions. Under this setting, the clients need to work together to train a model capable of classification for any of three health conditions. For the PU dataset, $4\times3 = 12$ client datasets are created. Clients 1,4,7 and 10 contain three types of training samples, while others only own two types of samples. The ISU dataset is also divided into 12 client datasets.

In scenarios 1 and 2, all the training datasets are utilized to generate client training datasets, and in scenario 3, some clients are assigned a portion of the total data to create the client-wise quantified discrepancy. The design of the training datasets for scenarios 1 and 2 is detailed in Table 5 in the appendix. Table 6 summarizes the design of the training datasets for scenario 3. The design of the test datasets for scenario 1 is detailed in Table 7 in the appendix. And Table 8 summarizes the design of test datasets for scenarios 2 and 3.

5.3. Alternative FL algorithms used for comparison

In this section, we evaluate the performance of the proposed FedSNGP algorithm and compare it with the local training algorithm and FedAvg [15]. The latter two algorithms have been widely used in FL applications. To examine the relative merit of uncertainty-based client clustering, we compare FedSNGP with a cosine similarity-based clustering algorithm (adopted from the work in Ref. [31]).

a **Standard Locally-Trained Fault Diagnosis Models:** To highlight the benefit of performing FL, we include the algorithm where each client updates their respective model parameters without FL. In the local training algorithm, each client trains the model for 250 epochs with a learning rate of 0.005.

Table 2
Summary of average test accuracy (%) for each case study.

Method	Scenario 1			Scenario 2			Scenario 3		
	CWRU	PU	ISU	CWRU	PU	ISU	CWRU	PU	ISU
Local training	99.58	99.93	100.00	75.69	77.72	50.11	70.56	71.33	45.76
FedAvg	99.37	98.31	98.80	99.16	98.14	98.20	89.17	97.67	92.80
FedCos	99.79	99.47	99.92	99.72	80.06	50.00	72.63	80.39	49.98
FedSNGP	99.58	99.90	100.00	99.44	99.92	100.00	95.56	99.81	99.66

b **Federated Averaging:** As a widely recognized FL algorithm, FedAvg performs weighted averaging to aggregate all the client's parameters. The training procedure of FedAvg is given in Fig. 1 a. When the datasets are non-IID, some models may perform worse than standard locally-trained models. FedAvg serves again as a second baseline for comparison where all clients' models are aggregated without considering data heterogeneity.

c **Federated Clustering Using Cosine Similarity:** The cosine similarity has been widely utilized in clustering-based FL [23,31]. This study adopts a clustering-based algorithm presented in Ref. [31] as a comparison algorithm, denoted as FedCos. The FedCos algorithm differs from the FedSNGP in that it determines the clustering strategy by evaluating the cosine similarity between the parameters of local models instead of utilizing predictive uncertainty results. Given two local model parameters θ^i and θ^j, the cosine similarity between the two models is defined as:

$$Sim_{Cos}(\theta^i, \theta^j) = \frac{\theta^i \cdot \theta^j}{\|\theta^i\| \|\theta^j\|} \tag{10}$$

The cosine similarity metric measures the angle between θ^i and θ^j, a smaller angle indicates a higher similarity between θ^i and θ^j. Similar to the FedSNGP algorithm, each communication round of the FedCos algorithm is composed of four steps:

(a) Each client trains a local model individually and uploads the model to the server.
(b) The server evaluates the cosine similarity between the parameters of local models and generates a similarity matrix M_{Sim}.
(c) The affinity propagation algorithm is applied to determine the clustering strategy and generate an updated global model for each cluster by using Eq. (1).
(d) Each client downloads the updated global model from the server.

The key difference between FedAvg, FedCos, and FedSNGP is the server's model aggregation algorithm. Therefore, in this study, all the clients utilize the same SNGP model architecture for their models, and the processed features are a 1D array with a length of 512. The SNGP model is designed with three spectral normalized residual blocks, and each block contains one fully connected layer with 64 units. The models in this study are trained using FL algorithms for a total of 50 communication rounds. In each communication round, each client performs local model updates for a period of five epochs with a fixed learning rate of 0.005.

6. Results and discussion

6.1. Evaluation results

Each of the 12 clients in the FL process undertakes their own diagnostic task, resulting in a total of 12 tasks for each test scenario. To evaluate the performance of the algorithms, we begin by examining the average test accuracy across all 12 clients. The average test accuracies are summarized in Table 2, and the detailed results are listed in Appendix Table 9.

In scenario 1, each client's training and test data follow a similar distribution. The model training and test are conducted using samples that have the same types of bearing faults. The performance of FedAvg is worse than the other three methods, which is caused by the heterogeneity of the dataset. Still, all four algorithms achieved a test accuracy of more than 95% for all three case studies.

Compared to scenario 1, scenarios 2 and 3 present greater challenges. In those scenarios, several clients have unbalanced training datasets, while the test dataset covers all fault types. Clients must acquire diagnosis knowledge from one another. Otherwise, they cannot identify unknown fault types. Consequently, the performance of the local training algorithm experienced a significant decline. For instance, when examining scenario 2 of the PU case study in Table 9 in the appendix, the prediction accuracy for client 2 by the local training algorithm is only 66.67%. This low-accuracy performance is expected since client 2's training dataset contains only healthy and outer race fault samples, rendering the locally trained model unable to classify inner race fault samples correctly. Meanwhile, client 2's model developed by FedAvg achieved a prediction accuracy of over 99%. Although FedAvg demonstrates significant improvement over local training, clients 4, 5, and 6 still achieved only 94.00% accuracy, which is lower than the accuracy number (99.67%) achieved by FedSNGP. Another noteworthy thing is that the models trained by FedCos perform worse than expected. For example, client 2's model under FedCos only achieved 66.67% accuracy, possibly due to incorrect client clustering.

6.1.1. Scenario 2 case studies

Fig. 7 shows the prediction results for scenario 2. The top half of the figure summarizes the average test accuracy. And the bottom half of the figure shows the specific diagnosis accuracy for each client.

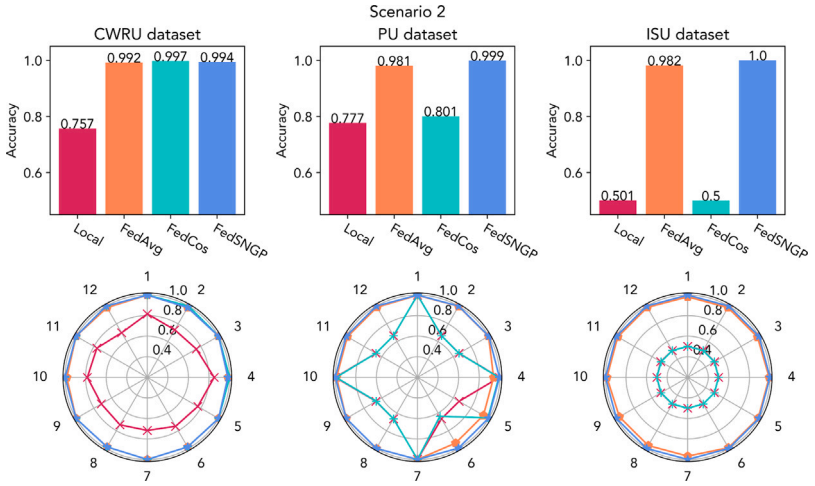

Fig. 7. The prediction results for scenario 2 case studies. The bar charts show the average test accuracy of each algorithm, and the radar chart shows the prediction accuracy of each client model generated by different training algorithms.

As a benchmark bearing dataset, the fault pattern of the CWRU dataset is obvious, the change in working conditions does not lead to a significant domain shift, and the fault patterns can be identified easily. Considering that the feature distribution discrepancy is not severe in scenario 2, the FL algorithms successfully utilize the client's information, and both FedAvg, FedCos, and FedSNGP achieved a test accuracy of more than 95%. As a comparison, the locally trained client models cannot identify unknown fault types. Resulting in an accuracy of around 75%.

For the PU dataset case study, the local datasets of clients 1, 4, 7, and 10 contain all the health conditions. That information is sufficient to train highly accurate diagnosis models. Therefore, the local training algorithm achieves a testing accuracy of around 100% for those four clients. On the other hand, the remaining eight client datasets were restricted to only one health and one faulty condition, preventing the client from training an accurate model independently without the knowledge of the missing fault types. As a result, the locally trained models for those eight clients were unable to effectively diagnose fault types that were not present in their respective training datasets. As a benchmark FL algorithm, FedAvg aggregates all the client's information by performing weighted averages to local models' parameters. Most of the models trained by FedAvg achieve an accuracy of more than 95%, and clients 4,5,6 yield a 94.00% accuracy. This decline in performance is attributed to the domain discrepancy among clients, primarily due to the variance in working conditions. The average accuracy of FedCos is 80.1%, which is slightly higher than the one of the local training method (77.7%). Based on the radar chart analysis, the performance of FedCos models was inferior to that of FedAvg models. The models trained by FedCos yield 100% accuracy on clients 1 and 7, and for other clients, the model's accuracy is around 2/3. This is mainly because FedCos is not able to accurately cluster the clients. The FedSNGP models achieved almost 100% accuracy for all the clients. Finally, the ISU dataset case study shows that the proposed FedSNGP outperforms the other three methods.

6.1.2. Scenario 3 case studies

In addition to domain shifts and partial domain adaptation, scenario 3 also considers differences in the number of training samples. In this scenario, some clients may possess a larger number of samples, while others may have only a limited number. Previous studies have indicated that unbalanced training sample quantities can significantly impact the performance of the FedAvg algorithm [23]. The evaluation results for scenario 3 case studies are summarized in Fig. 8. The feature distribution and sample quantity discrepancy make diagnosis tasks more challenging than scenario 2. As a result of the varying numbers of training samples held by each client, the training dataset exhibits more pronounced non-IID properties, resulting in the performance decline of FedAvg and FedCos models. However, the introduction of training sample number discrepancy does not affect the performance of the proposed FedSNGP. In all three case studies within scenario 3, models trained using FedSNGP outperformed those trained using other methods, achieving average prediction accuracies higher than 95%.

6.2. Analysis of the clustering strategy

Unlike FedAvg, which creates a single global model for all the clients, clustering-based FL algorithms assign clients to multiple clusters. Then, the clients in each cluster are federated together to train a global model. Several cosine similarity-based clustering FL algorithms have been developed [29,31]. This study considers both domain shift and partial domain adaptation setup. It is surprising that the FedCos algorithm does not show any improvement compared to FedAvg. In this section, the PU case study (scenario 2) compares the clustering strategy of FedSNGP and FedCos and analyzes why FedCos performs worse than FedSNGP.

The PU dataset was collected under four working conditions, and four data subsets were designed, with each subset containing vibration signals collected under a specific working condition. The kernel density estimation algorithm is a well-known method for

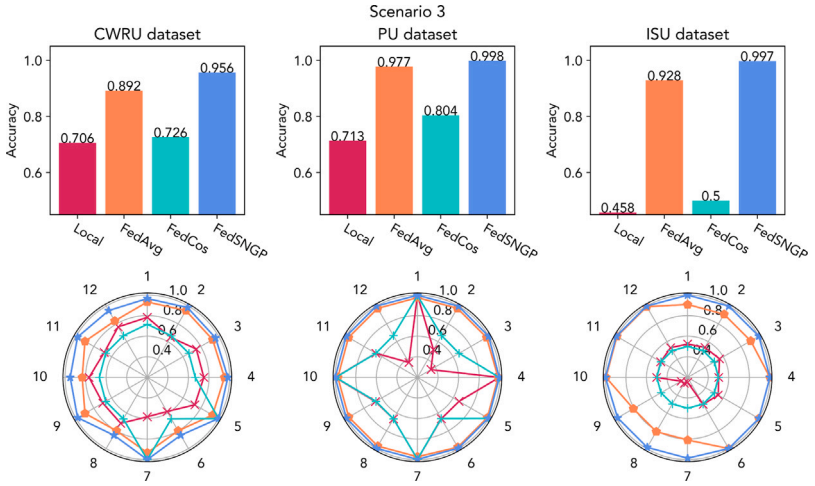

Fig. 8. The prediction results for scenario 3 case studies. The bar charts show the average test accuracy of each algorithm, and the radar chart shows the prediction accuracy of each client model generated by different training algorithms.

	Shaft speed (rpm)	Load Torque (Nm)	Radial force (N)
Subset A	1500	0.7	1000
Subset B	900	0.7	1000
Subset C	1500	0.1	1000
Subset D	1500	0.7	400

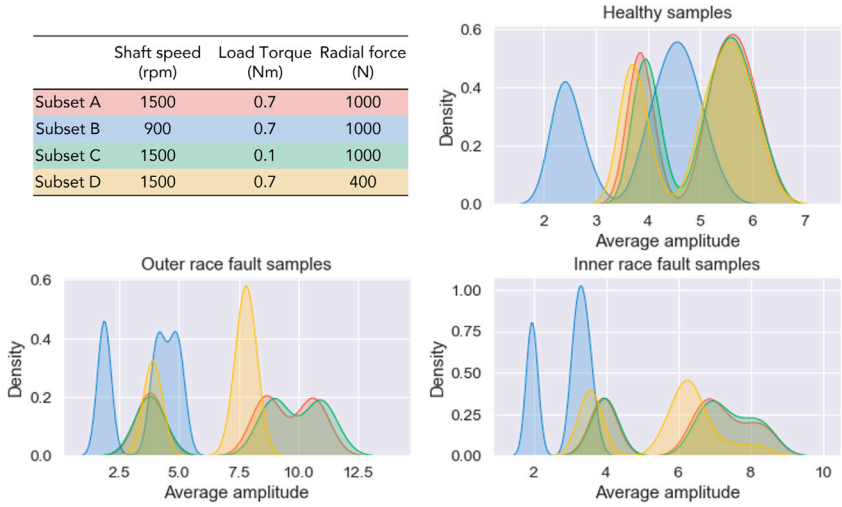

Fig. 9. Kernel density estimations of datasets by calculating the average amplitude of input features (sorted by different labels), the average amplitude distribution of subset A and subset C are similar (the kernel density estimations almost overlap), which indicates the data collected under those two operating conditions can be viewed as following similar data distribution and should be allocated into the same cluster.

estimating a random variable's probability distribution (probability density function) based on a given set of samples. This algorithm was used to analyze the average amplitude distribution of vibration signals [64]. Fig. 9 visualizes the distribution of the average amplitude of the samples. It has been acknowledged that the shaft speed affects the bearing fault characteristic frequencies, and the change of shaft speed affects the vibration amplitude a lot. The subset B data (marked by blue shade) was collected under 900 rpm, which is significantly different from the other three subsets. The kernel density estimation results also infer that the feature distribution of subset B is significantly different from the other three subsets. For the remaining three subsets, their healthy samples seem to follow a similar distribution. The feature distributions of outer race and inner race fault samples suggest that data in subset D differs slightly from the other two subsets. Based on the kernel density estimation analysis, it appears that subsets A and C have similar data distributions, while subset D has some differences from subsets A and C. Subset B, on the other hand, is notably different from the other subsets. A desired clustering strategy needs to make sure that: (a) the clients in the same subset (which means the client datasets were collected under the same working condition) should be grouped together; and (b) if two subset features share similar distributions, then those two subsets should be grouped together. Therefore, a satisfactory clustering strategy should yield the following three clusters for the PU case study: {client 1, client 2, client 3, client 7, client 8, client 9}, {client 4, client 5, client 6}, and {client 10, client 11, client 12}.

Fig. 10 shows the dynamic changes in clustering results by FedCos and FedSNGP in the different communication rounds. In the first communication round, since the models are still in the early training stage, both FedCos and FedSNGP cannot group clients

Clustering results

Communication round	FedCos	FedSNGP
1	[1],[2,3,5,6,8,9,10,11,12],[4],[7]	[1,3,4,5,9],[2,8,12],[6,7,10,11]
2	[1],[2,3,4,5,6,8,9,10,11,12],[7]	[1,2,3,7,8],[4,5,6],[9,10,11,12]
3	[1],[2,8,11],[3,6,7,9,12],[4,5,10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
4	[1,4,5,10],[2,8,11],[3,6,7,9,12]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
5	[1,4,5,6,7,10],[2,8,11],[3,9,12]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
6	[1,4,5,6,7,10],[2,8,11],[3,9,12]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
	...	
50	[1,7],[2,8,11],[3,6,9,12],[4,5,10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]

Fig. 10. Clustering results produced by FedCos and FedSNGP. The color shade for each client ID represents the subset ID (i.e., working condition) under which the data owned by the client was collected.

correctly. As the number of communication rounds increased, each client model started to capture the distribution of its training dataset. FedSNGP assigns clients into three clusters. The subsets A and C form one cluster, while B and subset D clients form two clusters. On the contrary, FedCos fails to group clients from the same subset into the same cluster. The incorrect clustering strategy employed by FedCos resulted in a decrease in the model's performance.

Fig. 11 shows the dynamic changes of the similarity matrix provided by FedCos and FedSNGP in each communication round. Each entry shows the calculated similarities between the two models. The darker blue indicates a higher similarity between the two clients. It should be noted that while FedCos and FedSNGP utilize the same model architecture, they use different metrics to infer data distribution similarity. FedCos estimates the similarity by comparing the model parameters. In the initial communication round, since each client model is initialized with the same parameters and has undergone only five local training epochs, the cosine similarity matrix between each pair of clients exhibits large values. After the second communication round, it is still difficult to determine if any two local models are significantly different. After the communication round 10, the results start to indicate high similarities between certain client models. However, due to the severe distribution discrepancy among clients, some clients are optimized in different directions. More importantly, within each communication round, the server clusters clients based on the model parameters, and the model aggregation process affects the updated model parameters. In case a client is assigned to an inappropriate cluster, it becomes challenging for the algorithm to rectify the mistake and assign the client to the correct cluster. As a result, the M_{Sim} generated by FedCos does not necessarily reflect the dataset similarity, and the algorithm does not assign clients of the same subset to the same cluster. For example, we expect the models of clients 1, 2, 3, 7, 8, and 9 to show a higher similarity because those three clients collect data under the same working condition (subset A). However, according to the entries of the first row, the normalized cosine similarity between clients 1 and 7 is significantly larger than other entries, while the cosine similarity between 1 and 2 is relatively low. Finally, only clients 1 and 7 are clustered together.

Unlike FedCos, which uses cosine similarity to cluster clients, the FedSNGP uses model predictive uncertainty as a driving force for clustering. FedSNGP generates a similarity matrix (M_{Sim}) that is non-symmetric, indicating that the similarity between two datasets is not necessarily the same in both directions. By considering predictive uncertainty, the FedSNGP is better equipped to estimate dataset similarity, making it more effective than FedCos in clustering clients for FL.

While after the first communication, the FedCos indicates that all the models are similar (with cosine similarity being more than 0.8 for each pair of clients), FedSNGP starts to indicate that certain client datasets are significantly different from others. For example, the second row indicates that dataset 2 is significantly different from dataset 4 and 5 (the similarity values are around 0). It is worth noting that a single client's estimation results can be unreliable. For instance, if we consider the perspective of client 3 (the third row), its data distribution is significantly different from that of clients 1 and 2. This is due to the fact that the training dataset of client 3 only covers healthy and outer race fault samples, unlike client 1, which has all the classes of training datasets. Consequently, the similarity estimations from client 3 may not be as dependable as those from client 1. While both the first row and second row indicate that clients 1,2,3,6,7, and 12 have similar data distributions, the third row suggests that the dataset of client 3 differs significantly from those of all the other clients. Affinity propagation clustering considers all clients' information and adaptively creates client clusters. Notably, after round 10 of communication, the FedSNGP algorithm formed three clusters: the first cluster included clients 1, 2, 3, 5, 6, and 7, the second cluster grouped clients 3, 4, and 5, and the third cluster included clients 10, 11, and 12.

Figs. 10 and 11 illustrate how clients are clustered using FedCos and FedSNGP, respectively. Fig. 10 analyzes the training datasets by examining the average amplitude. To validate the accuracy of the clustering strategy, we conducted the following test.

(a) We divide the whole dataset into four subsets based on the working conditions. We then use each subset to train an individual SNGP model. The total number of training epochs is 20. After every five epochs, we record the model parameters. Finally, we use the principal component analysis algorithm to map the parameters of each model onto a 2D plot.

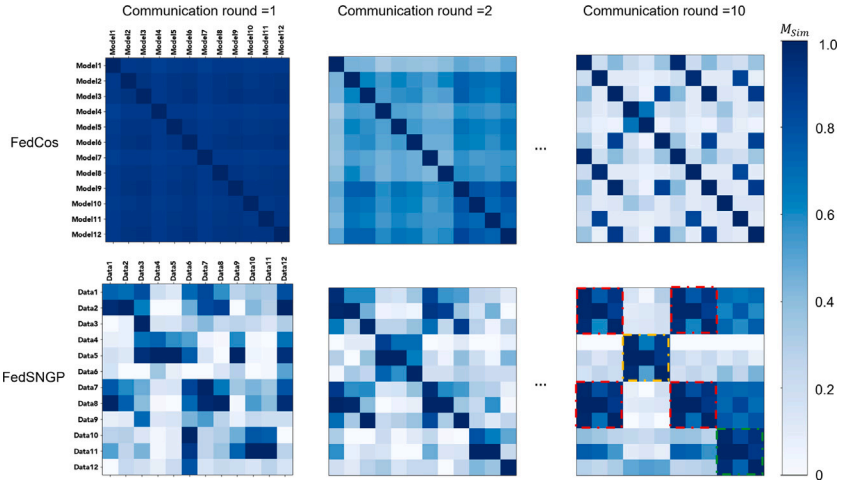

Fig. 11. Compare the similarity matrix generated by FedCos and FedSNGP at communication rounds 1, 2, and 10.

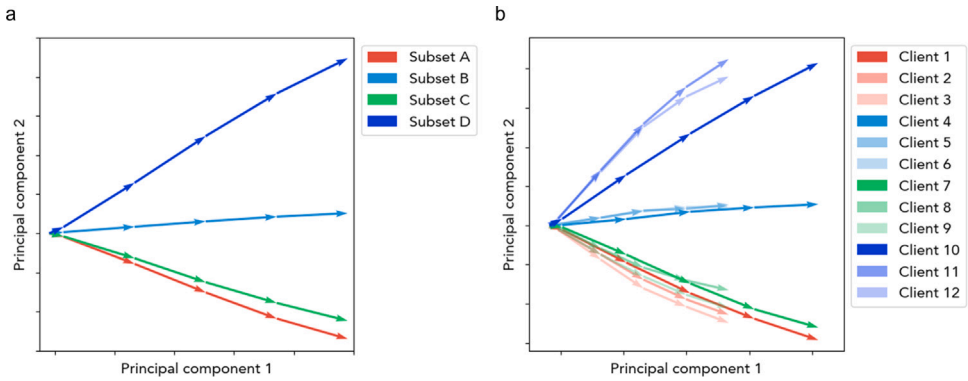

Fig. 12. Change of model parameters. The model parameters are mapped into 2D plots, and the arrow connects each adjacent model point to show the optimization direction of each model.

(b) In the context of the PU dataset (scenario 2 setting), we utilized 12 client datasets to train 12 models. The number of training epochs is 20. Again, model parameters are recorded at every five epochs. Finally, we use principal component analysis to map the parameters of the 12 models onto a 2D plot. As shown in Fig. 12a, the shift of model parameters indicates that models trained on subset A or C are being optimized in a comparable direction, suggesting that the data distribution of subset A is similar to subset C. According to our kernel density feature distribution analysis, subsets B and D should be trained independently without utilizing information from other subsets. This is further supported by Fig. 12b, which demonstrates that the client models generally form three clusters. However, for clients with imbalanced training datasets, such as clients 11 and 12, their optimization directions exhibit slight divergence from the desired direction, especially during the initial stages. This divergence may potentially mislead the cosine-similarity-based FL approach. Therefore, these findings suggest that at the early stages of local model training, the similarity of model parameters may not necessarily reflect the similarity or discrepancy of the data.

6.3. Handling out-of-cluster-distribution samples

One limitation of clustering-based FL is that the trained model's performance may degrade significantly toward out-of-distribution test samples. In the context of bearing diagnosis, if the testing data falls outside the distribution of the client's training dataset, the client may generate inaccurate predictions which can have a substantial impact on availability and maintenance costs. This limitation poses a challenge to the practical application of the diagnosis algorithm in industrial settings. However, unlike conventional clustering algorithms, the FedSNGP incorporates predictive uncertainty into the FL framework, which can quantify the predictive uncertainty toward the model's prediction result. This property of uncertainty quantification could potentially enhance the reliability and practicality of the FedSNGP algorithm for industrial applications.

In this section, we show how FedSNGP deals with out-of-distribution samples. If an SNGP model is tested with out-of-distribution samples, the predictive uncertainty increases significantly higher compared to the uncertainty towards the training dataset. To

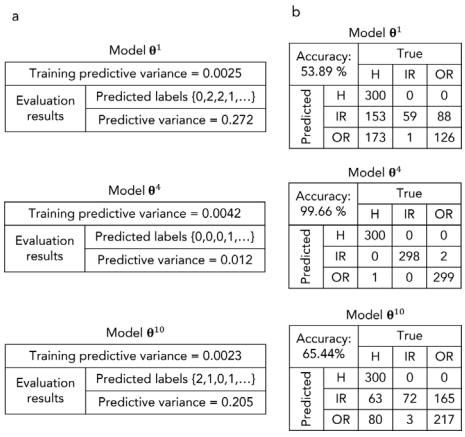

Fig. 13. An example of predicted variance-guided model selection: a. the model's predicted variance results. b. the confusion matrix of prediction results.

identify out-of-distribution samples, a predictive uncertainty threshold is adaptively defined based on a model's predicted variance on its training dataset. In this study, we set the threshold for each client's model to ten times the predicted variance on its own training dataset. During model evaluation, if a client model's predicted variance on a test dataset exceeds this threshold, it suggests that the test dataset falls outside of the client's training data distribution. Considering these two data distributions may differ substantially, the client will likely seek help from other clients.

To do this, the client downloads other client clusters' models from the server and tests them with the test dataset. The client selects the model that yields the least predicted variance. Fig. 13 provides a numerical example using client 1 in the PU dataset case study (scenario 2). The client first evaluates the test dataset using model θ^1, with a test predicted variance of 0.272, which is significantly larger than its predicted variance towards its training dataset. The client then seeks help from the server.

On the server side, it is known that clients form three clusters. The server shares the other two clusters' models (θ^4 and θ^{10}) with client 1. The model θ^4 yields the least predicted variance, and that value is within the threshold. Therefore, model θ^4's result is more reliable. The confusion matrix listed in Fig. 13b demonstrates that θ^4 yields the highest accuracy. The predicted variance-guided model selection strategy can fully utilize the information provided by all the clients.

6.4. Ablation study

The proposed FedSNGP method has two key steps: data similarity estimation and model clustering. The data similarity estimation step generates a similarity matrix to guide model clustering. In this section, we use the PU case study (scenario 2) as an example to analyze how these two steps influence the client models' diagnostic performance. The ablation study is performed by comparing FedSNGP with the following two baselines.

- The first baseline is designed by removing the "data similarity estimation" step, denoted as FedAP. Instead of having the affinity propagation algorithm take the similarity matrix as the input for client clustering, FedAP works by having the affinity propagation algorithm cluster clients according to the local models' parameters.
- The second baseline is designed by leaving out the "model clustering" step. Note that when the "model clustering" step is removed, all the models collaborate during the model aggregation to generate a single updated model. This single model will be allocated to every client for the next FL communication round, effectively making the resulting algorithm identical to FedAvg. Therefore, FedAvg is selected as the second baseline method.

The ablation study results are summarized in Table 3. It is clear that the client models developed by FedAP (the first baseline, no data similarity estimation) perform worse than those by the proposed FedSNGP method. Removing the data similarity estimation step leads to incorrect client clustering, making FedAP perform even worse than FedAvg(the second baseline, no model clustering). Table 4 summarizes the clustering results by FedAP and FedSNGP. The FedAP method fails to assign clients with similar data distributions to the same cluster. Similar to the FedCos method (see Section 6.2), FedAp uses the client models' parameters for clustering and parameter optimization. At the early stage of FL, the similarity between models' parameters may not necessarily reflect the similarity between models' training datasets. If the clustering results are incorrect at one communication round, it is challenging for FedAP to rectify the mistake in the following communication rounds.

The above comparisons between FedSNGP and its two baseline methods provide empirical evidence that the "model similarity estimation" step could allow the server to correctly cluster clients according to their similarity in training data distribution. Also, correctly clustering clients and then performing model aggregation within each cluster could produce more accurate models than models trained by the traditional FedAvg method.

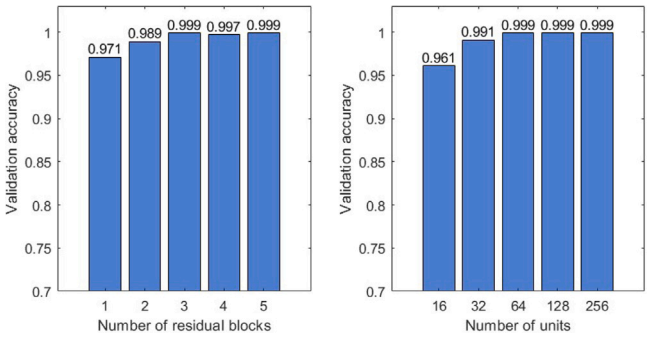

Fig. 14. Effects of hyper-parameters on fault diagnosis performance of the PU dataset.

Table 3
Comparison of prediction accuracy (%) of three different FL methods in the ablation study.

Client ID	FedAP	FedAvg	FedSNGP
1	100.00	99.78	100.00
2	98.44	99.78	100.00
3	66.56	99.78	100.00
4	99.67	94.00	99.67
5	78.33	94.00	99.67
6	64.00	94.00	99.67
7	99.56	100.00	100.00
8	99.56	100.00	100.00
9	66.44	100.00	100.00
10	100.00	98.78	100.00
11	79.56	98.78	100.00
12	66.67	98.78	100.00
Average	84.90	98.14	99.92

Table 4
Clustering results provided by FedAP and FedSNGP in the ablation study.

Communication round	FedAP	FedSNGP
1	[1],[2,3,4,5,6,8,9,11,12],[7],[10]	[1,3,4,5,9],[2,8,12],[6,7,10,11]
2	[1],[2,3,5,6,8,9,11,12],[4],[7],[10]	[1,2,3,7,8],[4,5,6],[9,10,11,12]
3	[1],[2,5,8,11],[3,6,9,12],[4],[7],[10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
4	[1],[2,5,8,11],[3,6,9,12],[4],[7],[10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
5	[1],[2,5,7,8,11],[3,6,9,12],[4],[10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
6	[1],[2,5,7,8,11],[3,6,9,12],[4],[10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]
	...	...
50	[1],[2,5,7,8,11],[3,6,9,12],[4],[10]	[1,2,3,7,8,9],[4,5,6],[10,11,12]

6.5. Hyper-parameter analysis

This section investigates the effects of the model hyper-parameters using the PU dataset. Two hyper-parameters are analyzed: (1) the number of spectral normalized residual blocks and (2) the number of units within each fully connected layer. The default setting is three residual blocks, and each block contains one fully connected layer with 64 units. Each client selects 20% of the training data, and after mixing, this subset is uniformly distributed among 12 clients, serving as their respective validation datasets. The average validation accuracies are summarized in Fig. 14.

The results show that the FedSNGP algorithm is minimally affected by the number of units in each fully connected layer and the number of residual blocks. The algorithm achieved 97.10% accuracy when the client model had a single block with 64 units. The setting of three blocks with 16 units achieved comparable results with 96.12% accuracy. With the increase of residual blocks or units, the validation accuracy shows slight improvement and then becomes stable (around 100%). In FL, a lightweight model generally leads to less computational burden and faster communication. With this in mind, we used a model with three residual blocks and 64 units is suitable for the results presented in this study. Note that in all the tested validation scenarios, the FedSNGP algorithm successfully groups clients to maximize accuracy, as expected. The model with less trainable parameters can be used based on the communication requirement in the practical implementations.

7. Conclusion

This study proposes an FL algorithm with uncertainty-based dynamic client clustering, called FedSNGP, to address the data heterogeneity challenges in fault diagnosis on fleets of system/component units. The benefits of using FedSNGP are two-fold.

First, estimating the data similarity during model training allows the server to effectively cluster clients to develop models better customized to each client's training dataset distribution. *Second*, uncertainty quantification during model evaluation enables each client to discern out-of-distribution samples and be informed of model confidence in making predictions, enhancing the reliability and practicality of the proposed method for industrial applications.

Three case studies are designed by defining client datasets with both domain shifts and open-set label shifts. Experimental results using three datasets and three FL scenarios demonstrate that our FedSNGP outperforms FedAvg and FedCos, achieving over 99% accuracy for all clients. The models trained by FedSNGP achieve high accuracy on data similar to their training data distribution and demonstrate the ability to identify out-of-distribution samples, which makes the proposed model applicable to open-set FL scenarios.

Despite the advantages mentioned above, the proposed FedSNGP method has some limitations and challenges that need future research. For example, estimating data similarity requires each client to download models from the server, resulting in more computational resources and longer training time compared to FedAvg. A potential strategy to mitigate this limitation is to reduce data transmission volumes between the server and clients by encoding model parameters. Another limitation of FedSNGP is that it requires each client to have access to the other clients' models, which may lead to privacy concerns. One way to mitigate this issue is to shuffle the order of client models and hide the models' client IDs when a client downloads other models. This makes it more challenging for each client to reconstruct other clients' training datasets. Besides the aforementioned improvement strategies, exploring transfer learning-based FL across datasets is also a topic of interest for future research.

CRediT authorship contribution statement

Hao Lu: Conceptualization, Data curation, Methodology, Software, Validation, Writing – original draft. **Adam Thelen:** Methodology, Validation, Writing – review & editing. **Olga Fink:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Chao Hu:** Conceptualization, Methodology, Project administration, Supervision, Writing – review & editing. **Simon Laflamme:** Project administration, Supervision, Writing – review & editing.

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

The ISU bearing fault dataset used in this paper will be shared openly in the arXiv version of this manuscript.

Acknowledgments

This work was supported in part by the U.S. National Science Foundation under Grant IIP-1919265. Any opinions, findings, or conclusions in this paper are those of the authors and do not necessarily reflect the sponsor's views.

Code availability

A code implementing FedSNGP in Python on the ISU dataset is provided at https://github.com/SalieriLu/FL_fault_diagnosis

Appendix

Design of the training datasets for scenarios 1 and 2

The training datasets of 12 clients in scenarios 1 and 2 follow the same design, summarized as Table 5.

Design of the training datasets for scenarios 3

To create data sample heterogeneity, in scenario 3, some clients only use a portion of the training dataset to train the model. The design of the training dataset for scenario 3 is listed as Table 6.

Design of the test datasets for scenarios 1

The design of test datasets for scenario 1 is summarized in Table 7.

Design of the test datasets for scenarios 2 and 3

The design of test datasets for scenarios 2 and 3 is summarized in Table 8.

Table 5
Design of training datasets for scenarios 1 and 2.

CWRU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1797 rpm	80	80	0	
2	Fault diameter = 0.007″	80	0	80	
3	Shaft speed = 1797 rpm	80	80	0	
4	Fault diameter = 0.014″	80	0	80	
5	Shaft speed = 1772 rpm	80	80	0	
6	Fault diameter = 0.007″	80	0	80	
7	Shaft speed = 1772 rpm	80	80	0	
8	Fault diameter = 0.014″	80	0	80	
9	Shaft speed = 1730 rpm	80	80	0	
10	Fault diameter = 0.007″	80	0	80	
11	Shaft speed = 1730 rpm	80	80	0	
12	Fault diameter = 0.014″	80	0	80	
PU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1500 rpm	1200	1200	1200	
2	Load torque = 0.7 N m	1200	1200	0	
3	Radial force = 1000 N	1200	0	1200	
4	Shaft speed = 900 rpm	1200	1200	1200	
5	Load torque = 0.7 N m	1200	1200	0	
6	Radial force = 1000 N	1200	0	1200	
7	Shaft speed = 1500 rpm	1200	1200	1200	
8	Load torque = 0.1 N m	1200	1200	0	
9	Radial force = 1000 N	1200	0	1200	
10	Shaft speed = 1500 rpm	1200	1200	1200	
11	Load torque = 0.7 N m	1200	1200	0	
12	Radial force = 400 N	1200	0	1200	
ISU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	Combination of faults
1	Shaft speed = 2100 rpm Radial force = 0	2000	2000	0	0
2		2000	0	2000	0
3		2000	0	0	2000
4	Shaft speed = 2100 rpm Radial force = 25 N	2000	2000	0	0
5		2000	0	2000	0
6		2000	0	0	2000
7	Shaft speed = 1500 rpm Radial force = 0	2000	2000	0	0
8		2000	0	2000	0
9		2000	0	0	2000
10	Shaft speed = 1500 rpm Radial force = 25 N	2000	2000	0	0
11		2000	0	2000	0
12		2000	0	0	2000

Table 6
The design of training datasets for scenario 3.

CWRU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1797 rpm	64	64	0	
2	Fault diameter = 0.007″	64	0	64	
3	Shaft speed = 1797 rpm	64	64	0	
4	Fault diameter = 0.014″	56	0	56	
5	Shaft speed = 1772 rpm	56	56	0	
6	Fault diameter = 0.007″	64	0	64	
7	Shaft speed = 1772 rpm	64	64	0	
8	Fault diameter = 0.014″	64	0	64	
9	Shaft speed = 1730 rpm	64	64	0	
10	Fault diameter = 0.007″	72	0	72	
11	Shaft speed = 1730 rpm	72	72	0	
12	Fault diameter = 0.014″	72	0	72	
PU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1500 rpm	480	480	480	
2	Load torque = 0.7 N m	480	480	0	
3	Radial force = 1000 N	480	0	480	
4	Shaft speed = 900 rpm	1200	1200	1200	
5	Load torque = 0.7 N m	1200	1200	0	
6	Radial force = 1000 N	1200	0	1200	
7	Shaft speed = 1500 rpm	600	600	600	
8	Load torque = 0.1 N m	600	600	0	
9	Radial force = 1000 N	600	0	600	
10	Shaft speed = 1500 rpm	720	720	720	
11	Load torque = 0.7 N m	720	720	0	
12	Radial force = 400 N	720	0	720	
ISU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	Combination of faults
1	Shaft speed = 2100 rpm Radial force = 0	1200	1200	0	0
2		1200	0	1200	0
3		1200	0	0	1200
4	Shaft speed = 2100 rpm Radial force = 25 N	1800	1800	0	0
5		1800	0	1800	0
6		1800	0	0	1800
7	Shaft speed = 1500 rpm Radial force = 0	800	800	0	0
8		800	0	800	0
9		800	0	0	800
10	Shaft speed = 1500 rpm Radial force = 25 N	1600	1600	0	0
11		1600	0	1600	0
12		1600	0	0	1600

Table 7
The design of test datasets for scenario 1.

CWRU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1797 rpm	20	20	0	
2	Fault diameter = 0.007″	20	0	20	
3	Shaft speed = 1797 rpm	20	20	0	
4	Fault diameter = 0.014″	20	0	20	
5	Shaft speed = 1772 rpm	20	20	0	
6	Fault diameter = 0.007″	20	0	20	
7	Shaft speed = 1772 rpm	20	20	0	
8	Fault diameter = 0.014″	20	0	20	
9	Shaft speed = 1730 rpm	20	20	0	
10	Fault diameter = 0.007″	20	0	20	
11	Shaft speed = 1730 rpm	20	20	0	
12	Fault diameter = 0.014″	20	0	20	
PU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1500 rpm	300	300	300	
2	Load torque = 0.7 N m	300	300	0	
3	Radial force = 1000 N	300	0	300	
4	Shaft speed = 900 rpm	300	300	300	
5	Load torque = 0.7 N m	300	300	0	
6	Radial force = 1000 N	300	0	300	
7	Shaft speed = 1500 rpm	300	300	300	
8	Load torque = 0.1 N m	300	300	0	
9	Radial force = 1000 N	300	0	300	
10	Shaft speed = 1500 rpm	300	300	300	
11	Load torque = 0.7 N m	300	300	0	
12	Radial force = 400 N	300	0	300	
ISU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	Combination of faults
1	Shaft speed = 2100 rpm Radial force = 0	500	500	0	0
2		500	0	500	0
3		500	0	0	500
4	Shaft speed = 2100 rpm Radial force = 25 N	500	500	0	0
5		500	0	500	0
6		500	0	0	500
7	Shaft speed = 1500 rpm Radial force = 0	500	500	0	0
8		500	0	500	0
9		500	0	0	500
10	Shaft speed = 1500 rpm Radial force = 25 N	500	500	0	0
11		500	0	500	0
12		500	0	0	500

Table 8
The design of test datasets for scenarios 2 and 3.

CWRU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1797 rpm	20	20	20	
2	Fault diameter = 0.007″	20	20	20	
3	Shaft speed = 1797 rpm	20	20	20	
4	Fault diameter = 0.014″	20	20	20	
5	Shaft speed = 1772 rpm	20	20	20	
6	Fault diameter = 0.007″	20	20	20	
7	Shaft speed = 1772 rpm	20	20	20	
8	Fault diameter = 0.014″	20	20	20	
9	Shaft speed = 1730 rpm	20	20	20	
10	Fault diameter = 0.007″	20	20	20	
11	Shaft speed = 1730 rpm	20	20	20	
12	Fault diameter = 0.014″	20	20	20	
PU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	
1	Shaft speed = 1500 rpm	300	300	300	
2	Load torque = 0.7 N m	300	300	300	
3	Radial force = 1000 N	300	300	300	
4	Shaft speed = 900 rpm	300	300	300	
5	Load torque = 0.7 N m	300	300	300	
6	Radial force = 1000 N	300	300	300	
7	Shaft speed = 1500 rpm	300	300	300	
8	Load torque = 0.1 N m	300	300	300	
9	Radial force = 1000 N	300	300	300	
10	Shaft speed = 1500 rpm	300	300	300	
11	Load torque = 0.7 N m	300	300	300	
12	Radial force = 400 N	300	300	300	
ISU					
Client ID	Operating conditions	Number of samples			
		Healthy	Inner race fault	Outer race fault	Combination of faults
1	Shaft speed = 2100 rpm Radial force = 0	500	500	500	500
2		500	500	500	500
3		500	0	500	500
4	Shaft speed = 2100 rpm Radial force = 25 N	500	500	500	500
5		500	500	500	500
6		500	500	500	500
7	Shaft speed = 1500 rpm Radial force = 0	500	500	500	500
8		500	500	500	500
9		500	0	500	500
10	Shaft speed = 1500 rpm Radial force = 25 N	500	500	500	500
11		500	500	500	500
12		500	500	500	500

Table 9
Summary of prediction accuracy (%) for three case studies.

CWRU												
	Scenario 1				Scenario 2				Scenario 3			
	Local training	FedAvg	FedCos	FedSNGP	Local training	FedAvg	FedCos	FedSNGP	Local training	FedAvg	FedCos	FedSNGP
1	100.00	100.00	100.00	100.00	81.67	100.00	100.00	100.00	78.33	93.33	71.67	96.67
2	95.00	97.50	100.00	97.50	73.33	98.33	100.00	98.33	65.00	95.00	66.67	98.33
3	100.00	100.00	100.00	100.00	75.00	100.00	100.00	100.00	75.00	93.33	66.67	96.67
4	100.00	100.00	100.00	100.00	85.00	98.33	100.00	98.33	75.00	95.00	66.67	98.33
5	100.00	100.00	100.00	100.00	76.67	100.00	100.00	100.00	73.33	93.33	100.00	100.00
6	100.00	100.00	100.00	97.50	75.00	98.33	98.33	98.33	58.33	80.00	66.67	85.00
7	100.00	100.00	100.00	100.00	71.67	100.00	100.00	100.00	58.33	93.33	100.00	100.00
8	100.00	97.50	97.50	100.00	73.33	98.33	98.33	98.33	71.67	80.00	66.67	85.00
9	100.00	100.00	100.00	100.00	71.67	100.00	100.00	100.00	70.00	90.00	66.67	98.33
10	100.00	100.00	100.00	100.00	78.33	98.33	100.00	100.00	76.67	83.33	66.67	95.00
11	100.00	100.00	100.00	100.00	76.67	100.00	100.00	100.00	68.33	90.00	66.67	98.33
12	100.00	97.50	100.00	100.00	70.00	98.33	100.00	100.00	76.67	83.33	66.67	95.00
PU												
	Scenario 1				Scenario 2				Scenario 3			
	Local training	FedAvg	FedCos	FedSNGP	Local training	FedAvg	FedCos	FedSNGP	Local training	FedAvg	FedCos	FedSNGP
1	99.89	99.78	100.00	100.00	99.89	99.78	100.00	100.00	100.00	97.56	99.56	99.89
2	100.00	99.65	100.00	100.00	66.67	99.78	66.67	100.00	51.00	97.56	66.89	99.89
3	100.00	100.00	100.00	100.00	66.67	99.78	66.56	100.00	35.11	97.56	66.67	99.89
4	99.56	94.00	99.11	99.67	99.56	94.00	99.11	99.67	99.56	98.44	99.67	99.56
5	100.00	97.21	100.00	100.00	66.67	94.00	99.11	99.67	66.67	98.44	99.67	99.56
6	99.66	92.33	95.40	99.15	66.56	94.00	63.67	99.67	66.44	98.44	66.22	99.56
7	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	99.89	97.22	99.67	99.78
8	100.00	99.83	100.00	100.00	66.67	100.00	66.67	100.00	66.67	97.22	66.67	99.78
9	100.00	100.00	100.00	100.00	66.67	100.00	66.44	100.00	67.22	97.22	66.67	99.78
10	100.00	98.78	99.22	100.00	100.00	98.78	99.22	100.00	100.00	97.44	99.56	100.00
11	100.00	100.00	100.00	100.00	66.67	98.78	66.67	100.00	66.78	97.44	66.67	100.00
12	100.00	98.26	100.00	100.00	66.67	98.78	66.67	100.00	36.67	97.44	66.89	100.00
ISU												
	Scenario 1				Scenario 2				Scenario 3			
	Local training	FedAvg	FedCos	FedSNGP	Local training	FedAvg	FedCos	FedSNGP	Local training	FedAvg	FedCos	FedSNGP
1	100.00	98.00	100.00	100.00	50.00	98.30	50.00	100.00	52.35	90.85	50.00	100.00
2	100.00	99.50	100.00	100.00	50.35	98.30	50.00	100.00	53.60	90.85	50.00	100.00
3	100.00	99.10	100.00	100.00	50.20	98.30	50.00	100.00	55.85	90.85	50.00	100.00
4	100.00	99.70	100.00	100.00	50.00	99.20	50.00	100.00	50.15	100.00	50.00	100.00
5	100.00	100.00	100.00	100.00	50.00	99.20	50.00	100.00	54.40	100.00	50.00	100.00
6	100.00	98.70	100.00	100.00	50.00	99.20	50.00	100.00	50.25	100.00	50.00	100.00
7	100.00	98.10	100.00	100.00	50.00	96.75	50.00	100.00	24.75	80.95	50.00	98.65
8	100.00	95.80	100.00	100.00	50.05	96.75	50.00	100.00	26.60	80.95	49.85	98.65
9	100.00	99.60	99.90	100.00	50.25	96.75	49.95	100.00	27.45	80.95	49.90	98.65
10	100.00	99.60	100.00	100.00	50.00	98.55	50.00	100.00	50.20	99.40	50.00	100.00
11	100.00	99.70	100.00	100.00	50.40	98.55	50.00	100.00	50.25	99.40	50.00	100.00
12	100.00	97.80	100.00	100.00	50.05	98.55	50.00	100.00	53.25	99.40	50.00	100.00

Results for each of the three test conditions

The results of the three case studies are summarized in Table 9.

References

[1] Mariela Cerrada, René-Vinicio Sánchez, Chuan Li, Fannia Pacheco, Diego Cabrera, José Valente de Oliveira, Rafael E. Vásquez, A review on data-driven fault severity assessment in rolling bearings, Mech. Syst. Signal Process. (ISSN: 0888-3270) 99 (2018) 169–196.

[2] Jay Lee, Fangji Wu, Wenyu Zhao, Masoud Ghaffari, Linxia Liao, David Siegel, Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications, Mech. Syst. Signal Process. 42 (1–2) (2014) 314–334.

[3] Yuekai Liu, Liang Guo, Hongli Gao, Zhichao You, Yunguang Ye, Bin Zhang, Machine vision based condition monitoring and fault diagnosis of machine tools using information from machined surface texture: A review, Mech. Syst. Signal Process. 164 (2022) 108068.

[4] Xiang Rao, Chenxing Sheng, Zhiwei Guo, Chengqing Yuan, A review of online condition monitoring and maintenance strategy for cylinder liner-piston rings of diesel engines, Mech. Syst. Signal Process. 165 (2022) 108385.

[5] Ying Lin, Maohua Xiao, Huijia Liu, Zhuolong Li, Shuang Zhou, Xiaomei Xu, Dicheng Wang, Gear fault diagnosis based on CS-improved variational mode decomposition and probabilistic neural network, Measurement 192 (2022) 110913.

[6] Sheng Shen, Hao Lu, Mohammadkazem Sadoughi, Chao Hu, Venkat Nemani, Adam Thelen, Keith Webster, Matthew Darr, Jeff Sidon, Shawn Kenny, A physics-informed deep learning approach for bearing fault detection, Eng. Appl. Artif. Intell. (ISSN: 0952-1976) 103 (2021) 104295.
[7] Qin Wang, Gabriel Michau, Olga Fink, Missing-class-robust domain adaptation by unilateral alignment, IEEE Trans. Ind. Electron. (ISSN: 0278-0046) 68 (1) (2020) 663–671.
[8] Qin Hu, Xiao-Sheng Si, Qing-Hua Zhang, Ai-Song Qin, A rotating machinery fault diagnosis method based on multi-scale dimensionless indicators and random forests, Mech. Syst. Signal Process. (ISSN: 0888-3270) 139 (2020) 106609.
[9] Suliang Ma, Mingxuan Chen, Jianwen Wu, Yuhao Wang, Bowen Jia, Yuan Jiang, High-voltage circuit breaker fault diagnosis using a hybrid feature transformation approach based on random forest and stacked autoencoder, IEEE Trans. Ind. Electron. 66 (12) (2018) 9777–9788.
[10] XiaoLi Zhang, BaoJian Wang, XueFeng Chen, Intelligent fault diagnosis of roller bearings with multivariable ensemble-based incremental support vector machine, Knowl.-Based Syst. (ISSN: 0950-7051) 89 (2015) 56–85.
[11] Zhenya Wang, Ligang Yao, Yongwu Cai, Rolling bearing fault diagnosis using generalized refined composite multiscale sample entropy and optimized support vector machine, Measurement 156 (2020) 107574.
[12] Olga Fink, Qin Wang, Markus Svensen, Pierre Dersin, Wan-Jui Lee, Melanie Ducoffe, Potential, challenges and future directions for deep learning in prognostics and health management applications, Eng. Appl. Artif. Intell. 92 (2020) 103678.
[13] Junbin Chen, Jipu Li, Ruyi Huang, Ke Yue, Zhuyun Chen, Weihua Li, Federated learning for bearing fault diagnosis with dynamic weighted averaging, in: 2021 International Conference on Sensing, Measurement & Data Analytics in the Era of Artificial Intelligence, ICSMD, IEEE, ISBN: 1665427477, 2021, pp. 1–6.
[14] Olga Fink, Torbjørn Netland, Stefan Feuerriegelc, Artificial intelligence across company borders, Commun. ACM (ISSN: 0001-0782) 65 (1) (2021) 34–36.
[15] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, Blaise Aguera y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial intelligence and statistics, PMLR, 2017, pp. 1273–1282.
[16] Zehui Zhang, Cong Guan, Hui Chen, Xiangguo Yang, Wenfeng Gong, Ansheng Yang, Adaptive privacy-preserving federated learning for fault diagnosis in internet of ships, IEEE Internet Things J. 9 (9) (2021) 6844–6854.
[17] Xue Ma, Chenglin Wen, Tao Wen, An asynchronous and real-time update paradigm of federated learning for fault diagnosis, IEEE Trans. Ind. Inform. (ISSN: 1551-3203) 17 (12) (2021) 8531–8540.
[18] Wei Zhang, Xiang Li, Hui Ma, Zhong Luo, Xu Li, Federated learning for machinery fault diagnosis with dynamic validation and self-supervision, Knowl.-Based Syst. (ISSN: 0950-7051) 213 (2021) 106679.
[19] Wei Zhang, Ziwei Wang, Xiang Li, Blockchain-based decentralized federated transfer learning methodology for collaborative machinery fault diagnosis, Reliab. Eng. Syst. Saf. (ISSN: 0951-8320) (2022) 108885.
[20] Solmaz Niknam, Harpreet S. Dhillon, Jeffrey H. Reed, Federated learning for wireless communications: Motivation, opportunities, and challenges, IEEE Commun. Mag. (ISSN: 0163-6804) 58 (6) (2020) 46–51.
[21] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, Zhihua Zhang, On the convergence of fedavg on non-iid data, 2019, arXiv preprint arXiv:1907.02189.
[22] Hangyu Zhu, Jinjin Xu, Shiqing Liu, Yaochu Jin, Federated learning on non-IID data: A survey, Neurocomputing (ISSN: 0925-2312) 465 (2021) 371–390.
[23] Guogang Zhu, Xuefeng Liu, Shaojie Tang, Jianwei Niu, Aligning before aggregating: Enabling cross-domain federated learning via consistent feature extraction, in: 2022 IEEE 42nd International Conference on Distributed Computing Systems, ICDCS, IEEE, 2022, pp. 809–819.
[24] Hongda Wu, Ping Wang, Node selection toward faster convergence for federated learning on non-iid data, IEEE Trans. Netw. Sci. Eng. (ISSN: 2327-4697) 9 (5) (2022) 3099–3111.
[25] I. Kevin, Kai Wang, Xiaokang Zhou, Wei Liang, Zheng Yan, Jinhua She, Federated transfer learning based cross-domain prediction for smart manufacturing, IEEE Trans. Ind. Inform. (ISSN: 1551-3203) 18 (6) (2021) 4088–4096.
[26] Yaxin Luopan, Rui Han, Qinglong Zhang, Chi Harold Liu, Guoren Wang, FedKNOW: Federated continual learning with signature task knowledge integration at edge, 2022, arXiv preprint arXiv:2212.01738.
[27] Yanxin Wang, Jing Yan, Zhou Yang, Yuannan Dai, Jianhua Wang, Yingsan Geng, A novel federated transfer learning framework for intelligent diagnosis of insulation defects in gas-insulated switchgear, IEEE Trans. Instrum. Meas. (ISSN: 0018-9456) 71 (2022) 1–11.
[28] Weihua Li, Wansheng Yang, Gang Jin, Junbin Chen, Jipu Li, Ruyi Huang, Zhuyun Chen, Clustering federated learning for bearing fault diagnosis in aerospace applications with a self-attention mechanism, Aerospace (ISSN: 2226-4310) 9 (9) (2022) 516.
[29] Chengxi Li, Gang Li, Pramod K. Varshney, Federated learning with soft clustering, IEEE Internet Things J. (ISSN: 2327-4662) 9 (10) (2021) 7773–7782.
[30] Dong Wang, Naifu Zhang, Meixia Tao, Adaptive clustering-based model aggregation for federated learning with imbalanced data, in: 2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications, SPAWC, IEEE, 2021, pp. 591–595.
[31] Pu Tian, Weixian Liao, Wei Yu, Erik Blasch, WSCC: A weight similarity based client clustering approach for non-IID federated learning, IEEE Internet Things J. (ISSN: 2327-4662) (2022).
[32] Saurabh Garg, Sivaraman Balakrishnan, Zachary Lipton, Domain adaptation under open set label shift, Adv. Neural Inf. Process. Syst. 35 (2022) 22531–22546.
[33] Manan Mehta, Siyuan Chen, Haichuan Tang, Chenhui Shao, A federated learning approach to mixed fault diagnosis in rotating machinery, J. Manuf. Syst. (2023).
[34] Pengfei Chen, Rongzhen Zhao, Tianjing He, Kongyuan Wei, Qidong Yang, Unsupervised domain adaptation of bearing fault diagnosis based on join sliced wasserstein distance, ISA Trans. 129 (2022) 504–519.
[35] Xiaohan Chen, Rui Yang, Yihao Xue, Mengjie Huang, Roberto Ferrero, Zidong Wang, Deep transfer learning for bearing fault diagnosis: A systematic review since 2016, IEEE Trans. Instrum. Meas. (2023).
[36] Jian Lin, Haidong Shao, Zhishan Min, Jingjie Luo, Yiming Xiao, Shen Yan, Jian Zhou, Cross-domain fault diagnosis of bearing using improved semi-supervised meta-learning towards interference of out-of-distribution samples, Knowl.-Based Syst. 252 (2022) 109493.
[37] Hanting Zhou, Wenhe Chen, Longsheng Cheng, Jing Liu, Min Xia, Trustworthy fault diagnosis with uncertainty estimation through evidential convolutional neural networks, IEEE Trans. Ind. Inform. (2023).
[38] Li Li, Yuxi Fan, Mike Tse, Kuo-Yi Lin, A review of applications in federated learning, Comput. Ind. Eng. 149 (2020) 106854.
[39] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, Yuan Gao, A survey on federated learning, Knowl.-Based Syst. 216 (2021) 106775.
[40] Weihua Li, Ruyi Huang, Jipu Li, Yixiao Liao, Zhuyun Chen, Guolin He, Ruqiang Yan, Konstantinos Gryllias, A perspective survey on deep transfer learning for fault diagnosis in industrial scenarios: Theories, applications and challenges, Mech. Syst. Signal Process. 167 (2022) 108487.
[41] Alysa Ziying Tan, Han Yu, Lizhen Cui, Qiang Yang, Towards personalized federated learning, IEEE Trans. Neural Netw. Learn. Syst. (2022).
[42] Junbin Chen, Jipu Li, Ruyi Huang, Ke Yue, Zhuyun Chen, Weihua Li, Federated transfer learning for bearing fault diagnosis with discrepancy-based weighted federated averaging, IEEE Trans. Instrum. Meas. (ISSN: 0018-9456) (2022).
[43] Ran Wang, Fucheng Yan, Liang Yu, Changqing Shen, Xiong Hu, Jin Chen, A federated transfer learning method with low-quality knowledge filtering and dynamic model aggregation for rolling bearing fault diagnosis, Mech. Syst. Signal Process. 198 (2023) 110413.
[44] Bin Jia, Xiaosong Zhang, Jiewen Liu, Yang Zhang, Ke Huang, Yongquan Liang, Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in IIoT, IEEE Trans. Ind. Inform. 18 (6) (2021) 4049–4058.
[45] Nastaran Gholizadeh, Petr Musilek, Federated learning with hyperparameter-based clustering for electrical load forecasting, Internet Things (ISSN: 2542-6605) 17 (2022) 100470.

[46] Christopher Briggs, Zhong Fan, Peter Andras, Federated learning with hierarchical clustering of local updates to improve training on non-IID data, in: 2020 International Joint Conference on Neural Networks, IJCNN, IEEE, 2020, pp. 1–9.
[47] Yuanjiang Li, Yunfeng Chen, Kai Zhu, Cong Bai, Jinglin Zhang, An effective federated learning verification strategy and its applications for fault diagnosis in industrial IOT systems, IEEE Internet Things J. (ISSN: 2327-4662) (2022).
[48] Rui Wang, Weiguo Huang, Mingkuan Shi, Jun Wang, Changqing Shen, Zhongkui Zhu, Federated adversarial domain generalization network: A novel machinery fault diagnosis method with data privacy, Knowl.-Based Syst. 256 (2022) 109880.
[49] Jun Lin, Jin Ma, Jianguo Zhu, Hierarchical federated learning for power transformer fault diagnosis, IEEE Trans. Instrum. Meas. 71 (2022) 1–11.
[50] Xiao Cong, Yan Song, Yibin Li, Lei Jia, Federated domain generalization with global robust model aggregation strategy for bearing fault diagnosis, Meas. Sci. Technol. 34 (11) (2023) 115116.
[51] Ke Zhao, Junchen Hu, Haidong Shao, Jiabei Hu, Federated multi-source domain adversarial adaptation framework for machinery fault diagnosis with data privacy, Reliab. Eng. Syst. Saf. 236 (2023) 109246.
[52] Xu Li, Chi Zhang, Xiang Li, Wei Zhang, Federated transfer learning in fault diagnosis under data privacy with target self-adaptation, J. Manuf. Syst. 68 (2023) 523–535.
[53] DaoQu Geng, HanWen He, XingChuan Lan, Chang Liu, Bearing fault diagnosis based on improved federated learning algorithm, Computing (ISSN: 0010-485X) (2022) 1–19.
[54] Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss, Balaji Lakshminarayanan, Simple and principled uncertainty estimation with deterministic deep learning via distance awareness, Adv. Neural Inf. Process. Syst. 33 (2020) 7498–7512.
[55] Jacob Gardner, Geoff Pleiss, Kilian Q. Weinberger, David Bindel, Andrew G. Wilson, Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration, Adv. Neural Inf. Process. Syst. 31 (2018).
[56] Uncertainty-aware deep learning with SNGP | TensorFlow core, URL https://www.tensorflow.org/tutorials/understanding/sngp.
[57] Xuming Han, Minghan Gao, Limin Wang, Zaobo He, Yanze Wang, A survey of federated learning on non-IID data IID data, ZTE Commun. 20 (3) (2022).
[58] Zhiyun Lu, Eugene Ie, Fei Sha, Mean-field approximation to Gaussian-softmax integral with application to uncertainty estimation, 2020, arXiv preprint arXiv:2006.07584.
[59] Brendan J. Frey, Delbert Dueck, Clustering by passing messages between data points, Science (ISSN: 0036-8075) 315 (5814) (2007) 972–976.
[60] Case western reserve university bearing dataset, Case Sch. Eng. (2021) URL https://engineering.case.edu/bearingdatacenter/download-data-file.
[61] Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, Walter Sextro, Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification, in: PHM Society European Conference, Vol. 3, No. 1, 2016.
[62] Gabriel Michau, Olga Fink, Unsupervised transfer learning for anomaly detection: Application to complementary operating condition transfer, Knowl.-Based Syst. 216 (2021) 106816.
[63] Katharina Rombach, Gabriel Michau, Olga Fink, Controlled generation of unseen faults for partial and open-partial domain adaptation, Reliab. Eng. Syst. Saf. 230 (2023) 108857.
[64] Firuz Kamalov, Kernel density estimation based sampling for imbalanced class distribution, Inform. Sci. 512 (2020) 1192–1201.