Federated Domain Generalization for Condition Monitoring in Ultrasonic Metal Welding

Ahmadreza Eslaminia[a,b], Yuquan Meng[a], Klara Nahrstedt[b], Chenhui Shao[a,c,*]

[a] *Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA*
[b] *Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA*
[c] *Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, USA*

Abstract

Ultrasonic metal welding (UMW) is a key joining technology with widespread industrial applications. Condition monitoring (CM) capabilities are critically needed in UMW applications because process anomalies, such as tool degradation and workpiece surface contamination, significantly deteriorate the joining quality. Recently, machine learning models emerged as a promising tool for CM in many manufacturing applications. Yet, many existing models lack the generalizability or adaptability and cannot be directly applied to new manufacturing process configurations (i.e., domains). Although several domain generalization techniques have been proposed, their successful deployment often requires substantial training data, which can be expensive and time-consuming to collect in a single factory. Such issues may be potentially alleviated by pooling data across factories, but data sharing raises critical data privacy concerns that have prohibited data sharing for collaborative model training in the industry. To address these challenges, this paper presents a Federated Domain Generalization for Condition Monitoring (FDG-CM) framework that provides domain generalization capabilities in distributed learning while ensuring data privacy. By effectively learning a unified representation from the feature space, FDG-CM

*Corresponding author

Email addresses: ae15@illinois.edu (Ahmadreza Eslaminia), yuquanm2@illinois.edu (Yuquan Meng), klara@illinois.edu (Klara Nahrstedt), chshao@umich.edu (Chenhui Shao)

Preprint submitted to Journal of Manufacturing Systems *October 1, 2024*

can adapt CM models for new clients (factories) with different process configurations. To demonstrate the effectiveness of FDG-CM, we investigate two distinct UMW CM tasks, including tool condition monitoring and workpiece surface condition classification. Compared with state-of-the-art federated learning algorithms, FDG-CM achieves a 5.35%–8.08% improvement in CM accuracy. FDG-CM is also shown to achieve excellent performance in challenging scenarios involving unbalanced data distributions and limited participating clients. Furthermore, by implementing the FDG-CM method on an edge-cloud architecture, we show that this method is both viable and efficient in practice. The FDG-CM framework is readily extensible to other manufacturing applications.
Keywords: Federated learning, Condition monitoring, Anomaly detection, Ultrasonic metal welding, Domain generalization, Quality control

1. Introduction

Ultrasonic metal welding (UMW) is an advanced joining technology that utilizes high-frequency oscillation shear forces to create a strong solid-state joint. It has been used in a wide range of industrial applications, including automotive body construction [1, 2] and lithium-ion battery assembly [3, 4]. Compared to conventional fusion welding methods, UMW offers several important advantages, such as reduced energy consumption, lower emissions, higher production rates, and environmental friendliness [5]. Therefore, it is one of the promising dissimilar metal joining techniques for sustainable manufacturing.

Despite numerous advantages, the quality of UMW joint is significantly influenced by process anomalies that may take different forms. Tool degradation and surface contamination of workpieces are two major causes of such anomalies in UMW [6, 7, 8]. As such, developing effective and efficient condition monitoring (CM) methods to ensure the quality and reliability of UMW has attracted in-depth research attention [6, 8]. In recent years, the application of machine learning methods in UMW CM has yielded promising outcomes. This success can be attributed to these models' ability to extract and learn complex patterns from high-dimensional data [9, 10].

Most machine learning-based CM methods are developed under the assumption that the training and test data are generated from the same manufacturing process configuration (i.e., domain), which guarantees same or highly similar data distributions. Nevertheless, this assumption does not always hold in the real world. Process reconfiguration takes place rather frequently in modern manufacturing [11]. In the industrial applications of UMW, manufacturers may adapt an UMW machine for different joining tasks, where different materials, welding parameters, and workpiece conditions are involved. Such process configurations often lead to distinct data distributions (domains). Consequently, the accuracy of a well-trained model substantially drops [12], leading to the phenomenon known as domain shift. To address the domain shift issue, recent works [12, 13] have developed CM models for UMW based on domain generalization and domain adaptation techniques. However, the success of these techniques requires full access to raw data from diverse process configurations (domains). Collecting such data can be both expensive and time-consuming in manufacturing, often leaving a single company without sufficient data to cover multiple configurations [14].

Pooling data from diverse configurations across different factories or companies, can offer a potential solution to the data availability problem [15]. However, collaborative model training through data pooling may raise significant data privacy concerns as the resistance to data integration can occur even within different departments of the same organization [16, 17]. To alleviate privacy constraints, the federated learning (FL) paradigm can be utilized, wherein models are trained locally, and only their weights are transmitted. However, FL's restriction on simultaneous access to raw data from diverse configurations, makes it challenging to accurately learn domain-invariant features and develop well-generalized models [18]. This dilemma can be addressed by federated domain generalization techniques [19], which enable multiple source domains to collaboratively learn a model that generalizes well to unseen domains while maintaining data privacy.

Generalizing federated models under domain shifts is a significant technical challenge that has received limited attention in existing research, particularly in the manufacturing sector [18]. To address this research gap, this paper develops an innovative Federated Domain Generalization for Condition Monitoring (FDG-CM) framework. FDG-CM is designed to operate across two or more distinct UMW CM tasks, such as tool condition monitoring and workpiece surface condition classification. These tasks are explored under various UMW process configurations. FDG-CM is able to effectively transfer knowledge by learning a generalized representation of the feature space shared across different datasets acquired from different companies while preserving privacy. Additionally, we introduce a new loss function to improve both the generalizability and convergence of the model under varying configurations. The main contributions of this research are summarized as follows:

1. The FDG-CM framework successfully achieves domain generalization in the context of FL. Compared with state-of-the-art FL algorithms, our framework exhibits superior performance, with an accuracy improvement of 5.35%–8.08% in new, previously unseen domains.

2. To the best of our knowledge, this research is the first to realize CM for UMW in an FL setting. This addresses the industry-wide concerns of data safety and data privacy, laying a foundation of large-scale collaborations among companies that use UMW in their production.

3. To demonstrate the industrial applicability of FDG-CM, we implement it in a real-world cyber-manufacturing environment. Running on an edge-cloud architecture, FDG-CM proves to be both data-efficient and time-efficient.

The remainder of this paper is organized as follows. In Section 2, the existing literature is reviewed in detail. Section 3 formulates the problem and describes the dataset used in this study. Section 4 introduces the proposed FDG-CM framework and Section 5 presents its implementation details. Several case studies are presented in Section 6 to thoroughly evaluate the effectiveness of the FDG-CM framework. Subsequently, Section 7 provides an in-depth discussion about the learning process, implications of the findings, and directions for future research. Finally, Section 8 concludes the paper. The developed code is publicly available at `https://github.com/AhmadrezaNia/FDG-CM`.

2. Related Work and Background

2.1. Data-driven CM for UMW

Data-driven CM methods are becoming increasingly crucial in many industrial applications [20, 21, 22], such as additive manufacturing, machining, and air brake systems [23, 24, 25, 26]. Similarly, in recent years, machine learning methods for monitoring UMW processes have also yielded promising outcomes [8, 9, 10]. For example, Nazir and Shao [8] developed an online sensing system and investigated several machine learning algorithms, such as logistic regression and K-nearest neighbors, to predict tool conditions in a real-time fashion. Wu et al. [9] utilized residual networks to predict joint quality in UMW with sensing signals. Schwarz et al. [27] improved process monitoring of UMW by using linear regression as well as multi-layer perceptron (MLP) regression to predict tensile shear strength. Most recently, Lu et al. [10] implemented an MLP classifier for identifying mixed welding disturbances, focusing on tool conditions and material surface conditions.

Despite these advances, a common challenge among most of these studies is their limited generalization ability across varying industrial situations due to the domain shift issue [12, 22, 28]. Only a few studies have aimed at addressing this challenge for the CM of UMW processes. Tian et al. [13] introduced a neural network (NN) model that incorporates an augmentation strategy to tackle the issue of domain shift in CM tasks within UMW. Meng et al. [12] developed a few-shot learning method to address the variability of configurations in anomaly detection for UMW. Despite good performance, this model encounters difficulties when there are no data available for new domains. Additionally, these methods presume the availability of data from diverse process configurations, a resource typically not accessible within a single factory. Building a model using training data across different companies, emerges as a viable solution to overcome data constraints. However, implementation of this approach often encounters privacy concerns, posing a significant barrier in many cases [29].

2.2. Federated learning

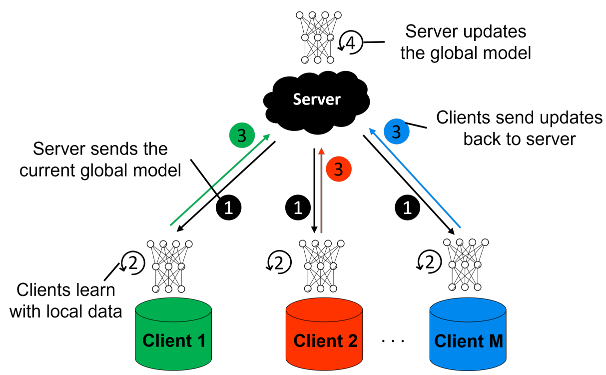

Figure 1: A schematic for training process in FL regime.

In recent years, FL has emerged as a promising solution to the challenges of data availability and privacy in distributed learning environments [30, 31]. The FedAvg method, a widely recognized baseline introduced by McMahan et al. [32], facilitates collaborative training across multiple data resources/clients without sharing raw data. Figure 1 illustrates a typical FL procedure. In each server round, the central server initially sends the model parameters to each participating client. Subsequently, clients independently train the received model on their local data and then transmit their local models' parameter updates back to the central server, which then aggregates these updates. The server round iterates until the model meets convergence criteria or a predetermined number of iterations is achieved.

Not all devices are available for local training at any given time. Thus, the FedAvg algorithm assumes m available clients from a total of M per server round, defining the ratio $\frac{m}{M}$ as the client fraction (C). The total number of data points (N) is obtained by summing n_k, the data points from each of the m clients. FedAvg's objective is to minimize the following optimization function:

$$\arg\min_{\theta} F(\theta) = \frac{1}{N}\sum_{k=1}^{m} n_k \cdot F_k(\theta), \tag{1}$$

$$\text{s.t. } F_k(\theta) = \frac{1}{n_k}\sum_{j=1}^{n_k} f_j(\theta), \tag{2}$$

where θ represents the model parameters, $F_k(\theta)$ is the loss function of the k-th client, and $f_j(\theta)$ represents the loss of each individual data point.

To effectively utilize FedAvg, it is essential to define some key parameters: the local learning rate (η), the number of local training epochs (E) per client per server round, and the local mini-batch size (B).

To enhance FedAvg's convergence, Li et al. (2020) [33] proposed the FedProx method by adding a proximal term to the client loss, thereby limiting the impact of variable local updates. The objective function of each client in FedProx is:

$$\min_{\theta} h_k(\theta; \theta_t) = F_k(\theta) + \frac{\mu}{2}\|\theta - \theta_t\|^2, \tag{3}$$

where μ represents a parameter controlling the proximal term, and $\|\theta - \theta_t\|^2$ denotes the squared norm of the difference between the current model θ and the previous model θ_t. A large μ may lead to slower convergence by keeping updates near the initial point, while a small μ may not make any difference.

The model's ability to generalize to new users is paramount in FL. In this regard, Nguyen et al. (2022) [34] developed the FedL2R algorithm that incorporates an L2-norm regularizer (L2R) into the representation of the network to improve the generalizability across new domains by limiting the representation's information capacity. Thus, the final local objective function of each client k is:

$$\min_{\theta} F_k(\theta) + \alpha^{L2R} \cdot \mathcal{L}_{\mathrm{k}}^{L2R}(\theta), \tag{4}$$

$$\text{s.t.} \quad \mathcal{L}_{\mathrm{k}}^{L2R} = \frac{1}{n_k}\sum_{n=1}^{n_k} \|z_k^{(n)}\|_2^2, \tag{5}$$

where $z_k^{(n)}$ represents the model representation of a specific layer for the client k and data point n, and α^{L2R} is a hyper-parameter. It was demonstrated that in domain generalization tasks, FedL2R surpasses relevant FL baselines, such as the federated adversarial domain generalization method introduced in [35].

The aforementioned works mainly targeted cross-device decentralized learning. Yet, existing research has underscored the value of cross-silo FL, where fewer but resource-rich organizations like manufacturers, hospitals, and banks serve as clients [36]. FL has recently received attention in the manufacturing sector [37]. For instance, Ahn et al. [38] employed FL for anomaly detection and predictive maintenance in pumps. FL methods were developed for defect detection in metal additive manufacturing [15] and fault diagnosis in rotating machinery [16]. Despite these advances, there is a notable lack of research on FL for UMW. CM in UMW poses more significant challenges due to complex process physics and limited understanding of process mechanisms. Moreover, the presence of multiple types of anomalies complicates online monitoring [10].

While FL can address the data availability issue in CM applications by enabling secure collaboration among data sources, its restriction on accessing raw data can impair the effectiveness of traditional domain generalization techniques in coping with the domain shift issue [37]. To address domain shift in FL settings, a few federated domain generalization techniques have been proposed, e.g., [39, 40, 41]. Kevin et al. [40] developed a federated transfer learning method for cross-domain knowledge transfer, achieving better accuracy in target applications with less data and time. However, this framework was not tested on manufacturing datasets and lacked test data from entirely unseen domains. In a separate effort, Chung et al. [41] proposed the FedEntropy framework, a personalized FL method using flat minima, which is known to perform better on unseen data than sharp minima, to enhance generalization. Yet, the effectiveness of their framework in real-world scenarios is unknown, as it was evaluated using a synthetic dataset on aircraft engine degradation.

In summary, although FL has been investigated in manufacturing applications, methods that specifically address the domain shift in these applications, including CM for UMW, have been limitedly studied. Moreover, effective methods using real-world test data from entirely unseen domains are generally lacking among these studies.

3. Problem Formulation and Data Description

3.1. Problem statement

This paper investigates the domain shift problem, particularly federated domain generalization, in CM for UMW, under the following general assumptions.

- Data privacy preservation: The raw data remains localized and is not shared between clients and the central server.
- Similar feature space across datasets: We assume the availability of at least two datasets, each possessing a similar feature space but potentially encompassing distinct domain spaces and classification tasks.
- Data distribution heterogeneity: It is assumed that participating domains have distinct but related data distributions. This diversity motivates the need for domain generalization.
- Common task objective: Each dataset's domains share a common task or objective despite their differences, and the goal is to develop a model for each dataset that can generalize across these domains to make accurate predictions on the common task.
- Evaluation on unseen domains: The model's performance is evaluated based on its ability to generalize to unseen domains during testing.

Consider a scenario where data from several source domains with the same task is available for training. The goal is to develop a model that performs well on a different domain, known as the target domain, for the same task. In CM for UMW, we consider different process configurations, which are characterized by different workpiece materials, welding parameters, etc., as different domains. These domains share a common task, such as classifying tool conditions. The primary goal in domain generalization is to develop a model that utilizes data from the source domains/configurations to effectively perform the common task on never-before-seen data from the target domain/configuration. We assume that each client in the FL network corresponds to a company with access solely to data from a specific configuration.

3.2. Dataset description and preprocessing

Table 1: Design of experiments for domain groups M, S, and T configurations [12].

(a) Domain group M configuration

Domain	Material	Welding time	Data size	Classification goal
AC	Al-Cu	0.5 s	200	TC1, TC2, TC3, TC4
CC	Cu-Cu	0.9 s	200	
CA	Cu-Al	0.5 s	200	
AA	Al-Al	0.9 s	200	

(b) Domain group S configuration

Domain	Data size	Classification goal
Clean	90	New, Worn, DMGD
Polished	90	
Contam	90	

(c) Domain group T configuration

Domain	Data size	Classification goal
DMGD	90	Clean, Polished, Contam
New	90	
Worn	90	

We utilize datasets from a previous work [12] for CM of UMW. In UMW, the joint quality is influenced by the interplay of process configurations, tool conditions, and workpiece properties [10]. The experiments were carried out using the Branson Ultraweld L20 ultrasonic welding machine that is equipped with an online monitoring system. Figure A.1 in Appendix A shows data collected by the monitoring system from four sensors: a built-in linear variable differential transformer (LVDT), a built-in power sensor, an acoustic emission

(AE) sensor, and a microphone. In these datasets, different types of materials, tool conditions, and surface conditions are categorized into the following three distinct groups.

Domain group M: The welding samples are generated by combining different materials from either 0.25 mm thick Copper (Cu) or Aluminum (Al) sheets, each subjected to varying welding times, as detailed in Table 1a. Each combination/domain comprises 200 samples, evenly distributed across four tool conditions, serving as the classes: new horn/new anvil (TC1), new horn/worn anvil (TC2), worn horn/new anvil (TC3), and worn horn/worn anvil (TC4).

Domain group S: In this setup, 0.20 mm Cu sheets are welded for fixed welding time of 1.0 second. As detailed in Table 1b, different domains are characterized by distinct surface conditions, including a "Clean" surface prepared using alcoholic wipes, a "Polished" surface achieved by sandpapering the contact faces, and a "Contam" (Contaminated) surface created by contaminating the surface by cutting fluid. The objective of the S domains is to classify three tool conditions: new horn/new anvil (New), worn horn/worn anvil (Worn), and damaged horn/damaged anvil (DMGD). 30 samples were generated for each combination of workpiece surface condition and tool condition, resulting in a total of 270 samples.

Domain group T: In this domain group, the data is identical to that in domain group S. However, the primary objective shifts to classifying the surface conditions while maintaining different domains representing distinct tool conditions, as outlined in Table 1c.

To capture high-frequency information, the monitoring system samples data at over 200 kHz, resulting in each raw data instance containing four signals, each exceeding 100,000 data points. To tackle the challenge of high dimensionality, we apply discrete wavelet transformation (DWT) [5] to decompose the original signals into 13 levels of wavelet coefficients, and 12 statistical indexes are computed for each level. These indexes include entropy, zero crossing rate, mean crossing rate, various percentiles, median, mean, standard deviation, variance, and root mean square value. A detailed explanation of each index is avail-

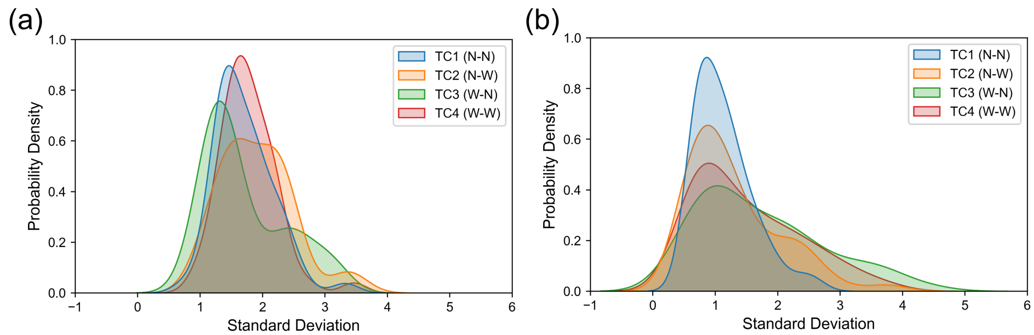

Figure 2: Comparison of the distribution of standard deviation at the first wavelet coefficient level from the AE signal between (a) Domain Cu-Cu and (b) Domain Al-Cu, both from Domain Group M. The classes represent different tool conditions: TC1 (New horn/New anvil), TC2 (New horn/Worn anvil), TC3 (Worn horn/New anvil), and TC4 (Worn horn/Worn anvil). The distribution is estimated by kernel density estimation.

able in [5]. This approach yields a total of 624 features for each data instance. Figure 2 compares the standard deviation at the first wavelet coefficient level for the AE signal between the Cu-Cu and Al-Cu domains across different tool conditions (classes). The results show that the distribution of different tool conditions in the feature space varies significantly across domains, highlighting the domain shift issue. The observed distribution shift is primarily due to the differences in material properties between Cu and Al, such as melting temperature and thermal conductivity. For instance, Al has a lower melting temperature and higher thermal conductivity than Cu, leading to distinct vibration patterns for different material combinations [42]. These unique patterns are reflected in the feature distributions captured by the sensors.

4. Methodology

This section presents the details of the FDG-CM framework. Specifically, Section 4.1 elaborates on the novel approach used for aggregating model parameters. This aggregation method, along with the loss function introduced in Section 4.2, aims to overcome the domain shift issue in FL settings and enable domain generalization capabilities for CM models.

4.1. The FDG-CM framework

In this framework, knowledge is transferred between at least two domain groups that have the same feature space. In a typical NN-based model, different features are extracted in different layers. The lower layers extract low-level features that are very likely to be transferrable between heterogeneous applications [40]. For instance, in the CM of industrial processes, while there are differences between the domains and tasks among different datasets, certain fundamental characteristics of the processes, rooted in the underlying physics, remain consistent. This consistency in the underlying physics and feature space enables the learning of invariant features that can be generalized across different domain groups. Therefore, the goal is to commonly train the initial layers across various domain groups, to facilitate the collaborative extraction of low-level information of the common feature space. This approach not only broadens the data pool for these shared layers but also enables them to see data from different domain spaces.

Although the feature space across domain groups is similar, the significance of specific features may differ between domain groups and tasks. Accordingly, the framework is designed to yield a personalized final model for each domain group, customizing these models to their specific tasks. To achieve this personalization, the framework leverages model parameters from clients within the same domain group to train a set of upper layers that capture domain-specific features. These layers are then integrated with the earlier-mentioned lower layers to construct a comprehensive NN model. This strategy of knowledge transfer and model personalization is anticipated to significantly improve the performance of the final models, especially in tasks requiring domain generalization.

Utilizing a shared base layer among all clients, accompanied by personalized layers for each, has been previously implemented in the FedPer algorithm [43]. However, FedPer restricts collaboration among clients with similar tasks, as it does not allow for the sharing of personalized layers with the server. In contrast,

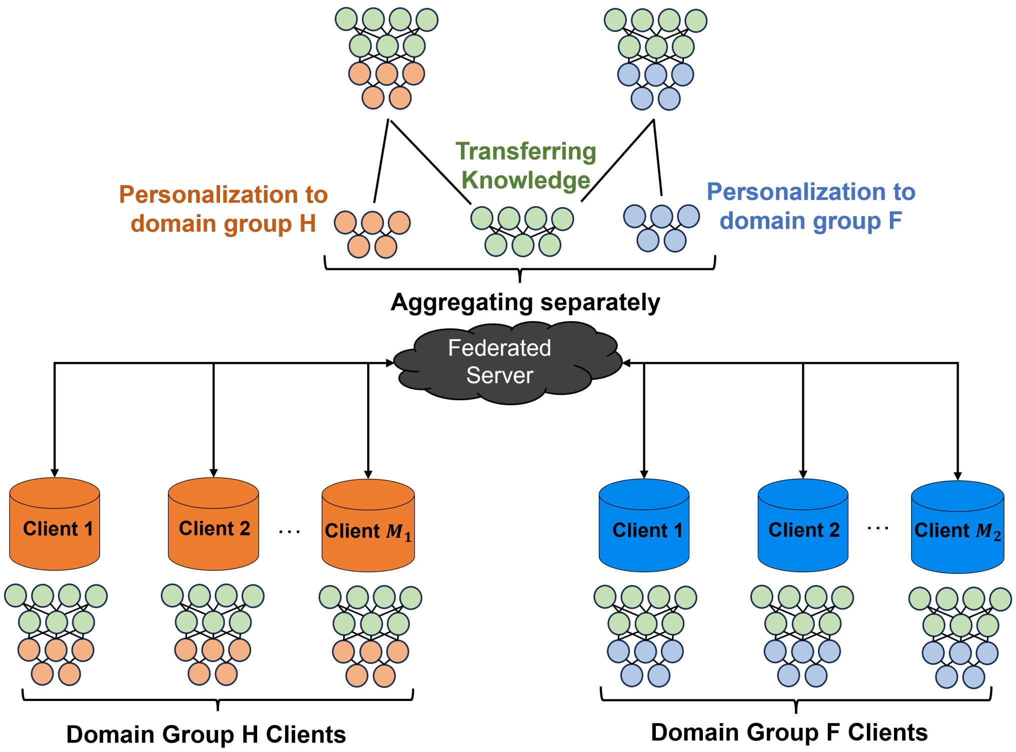

Figure 3: Illustration of the proposed FDG-CM structure. Note that in this framework, the number of classes for each domain group and the neuron count in the blue and orange layers can differ.

our approach promotes collaboration by enabling the aggregation of common personalized layers among clients with identical tasks and domain groups.

For simplicity, we illustrate our framework using two CM classification tasks with domain shift issues, though it is designed to be scalable for more tasks. Figure 3 presents the details of implementing the FDG-CM framework on two hypothetical datasets from domain group H and domain group F, each associated with a specific task and assigned to a separate group of clients. In the first step, the server initializes and distributes two distinct models to the two groups of clients. Each model consists of shared base layers, depicted in green in the accompanying figure, followed by personalized layers tailored specifically to each dataset. For domain group H, these personalized layers are highlighted in orange, while for domain group F, they are shown in blue.

In the next step, each client trains the provided model on its local data and returns the updated model to the server. Then, the server aggregates the base layers from all clients while only aggregating the personalized layers from the clients of the same domain group. The server then sends the two aggregated models back to the clients. These steps iterate until both models converge. This approach ensures that the base layer, learning a general data representation, benefits from both datasets, while the personalized layers, which extract task-specific information, are learned exclusively from their respective datasets.

In this approach, the number of base layers (K_B) is considered as a hyper-parameter, and all preceding parameters are the base parameters (θ_B). Furthermore, K_{P1} and K_{P2} represent the number of personalized layers in each model, with corresponding parameters denoted as θ_{P1} and θ_{P2}. The total number of clients in each domain group is M_1 and M_2. Hyper-parameters, such as client fractions (C_1, C_2), must be specified individually for each client set. Notably, as the same base layer is employed for both sets of clients, a unified learning rate (η), client epoch (E), and batch size (B) are defined for all clients to enhance convergence. Algorithm 1 shows the details of the server and client steps.

4.2. The FDG-CM loss function

Using a shared base layer in different models that serve clients with various domains and tasks may lead to divergence issues. To counter this, we incorporate the FedProx [33] loss term into the loss function. As mentioned in Section 2.2, this addition effectively reduces the disparity in local updates by penalizing changes in local parameters. Moreover, to address the domain shift issue, we include an L2R term [34] on the output of the shared base layer(s). The L2R term aids in learning a simpler representation from the feature space by constraining this representation, thus improving the model's generalization across various domains. Consequently, the objective function of our framework is defined as:

$$\min_{\theta} F_k(\theta) + \alpha^{L2R} \cdot \mathcal{L}_{\mathrm{k}}^{L2R}(\theta) + \frac{\mu}{2}\|\theta - \theta_t\|^2, \tag{6}$$

Algorithm 1 FDG-CM Method

Server side:

Initialize parameters θ_B^0, θ_{P1}^0, θ_{P2}^0

for each server round $t = 1, 2, \ldots$ **do**

  select random subsets m_1, m_2 with $[C_1 \cdot M_1]$, $[C_2 \cdot M_2]$ number of clients

  Send the model $\theta_B^{t-1} + \theta_{P1}^{t-1}$ and $\theta_B^{t-1} + \theta_{P2}^{t-1}$ to the selected m_1 and m_2 clients respectively.

  for each client $k_1 \in m_1$ in parallel **do**

    $\theta_B^t(k1), \theta_{P1}^t(k1) \leftarrow \text{ClientUpdate}(k_1, \theta_B^{t-1}, \theta_{P1}^{t-1})$

  end for

  for each client $k_2 \in m_2$ in parallel **do**

    $\theta_B^t(k2), \theta_{P2}^t(k2) \leftarrow \text{ClientUpdate}(k_2, \theta_B^{t-1}, \theta_{P2}^{t-1})$

  end for

  $N_{P1} \leftarrow \sum_{k1=1}^{m_1} n_{k1}$

  $N_{P2} \leftarrow \sum_{k2=1}^{m_2} n_{k2}$

  $N \leftarrow N_{P1} + N_{P2}$

  $\theta_B^t \leftarrow \frac{1}{N}(\sum_{k1=1}^{m1} n_{k1} \cdot \theta_B^t(k1) + \sum_{k2=1}^{m2} n_{k2} \cdot \theta_B^t(k2))$

  $\theta_{P1}^t \leftarrow \frac{1}{N_{P1}} \sum_{k1=1}^{m1} n_{k1} \cdot \theta_{P1}^t(k1)$

  $\theta_{P2}^t \leftarrow \frac{1}{N_{P2}} \sum_{k2=1}^{m2} n_{k2} \cdot \theta_{P2}^t(k2)$

end for

Client side:

for each local epoch $i = 1, 2, \ldots, E$ **do**

  for each mini-batch b of size B **do**

    $\theta \leftarrow \theta - \eta \nabla \mathcal{F}(\theta, b)$

  end for

end for

Send the updated local model θ to the server

$$\text{s.t. } \mathcal{L}_{\mathrm{k}}^{L2R} = \frac{1}{n_k}\sum_{n=1}^{n_K}\|z_k^{(n)}\|_2^2, \tag{7}$$

where α^{L2R} and μ are hyper-parameters. Additionally, $z_k^{(n)}$ is the output of the latest base layer for client k with data point n. Combining L2R and FedProx terms in this optimization function aims to achieve superior generalization and enhance model convergence.

5. Implementation Details

5.1. NN architecture

Figure A.1 in Appendix A shows the designed MLP architecture with four layers. The MLP was chosen due to its demonstrated effectiveness in the literature for CM of UMW using DWT features [5, 10, 12]. Additionally, given the limited number of data points available at each client, the MLP architecture is more suitable than deeper neural networks, reducing the risk of overfitting. The model takes 624 DWT features as input, and the output layer provides the classification results for each specific dataset (4 classes for domain group M and 3 classes for domain groups S and T). For the first three hidden layers, the model has 175, 125, and 50 neurons, respectively. The rectified linear unit is chosen as the activation function, and the adaptive moment estimation is selected as the solver.

5.2. Edge-cloud implementation

To emulate a manufacturing environment, we train models on single-board computers, specifically using Raspberry Pis[1] due to their common use as edge devices. Typically, multiple welding machines operate within a single factory; thus, each Raspberry Pi can represent a factory, training models for several welding machines. These devices, emulating companies, connect to a remote central server via the Internet for collaborative model training. The entire

[1]Raspberry Pi is a trademark of Raspberry Pi Ltd

training process is coordinated by the central server, which defines all learning strategies, including the methods used, model hyperparameters, the selection of participating clients, and the number of server rounds. Clients receive the model and hyperparameters from the server, and then conduct their training independently.

The RabbitMQ broker, a publish-subscribe system, connects the server node and client nodes in a bi-directional manner [44]. In RabbitMQ, messages are published to exchanges and delivered to queues based on routing rules defined by bindings and routing keys. This decoupling of producers and consumers enhances scalability, flexibility, and robustness. Secure connections to the RabbitMQ broker require the server and clients to authenticate using a username and password.

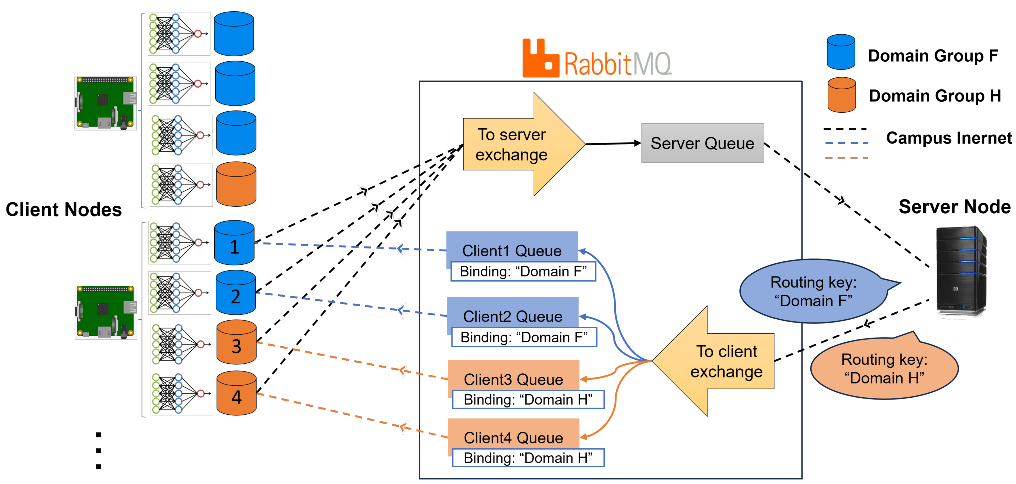

Figure 4: A schematic of edge-cloud implementation for the FDG-CM method.

Figure 4 illustrates the details of message routing in our implementation. For simplicity, we consider clients on only one Raspberry Pi—two from domain group H and the other two from domain group F. Initially, the server publishes two initialized models and corresponding hyperparameters as two JSON files each tagged with a specific routing key, "Domain F" or "Domain H," to the broker. These routing keys are essential for the direct exchange model, labeled as

"To clients exchange," to route messages accurately to the corresponding client queues. Then each client subscribes to its queue to retrieve its parameters for the next round. After the training of the received models on their private data, each client publishes a JSON file with their updated parameters and a topic indicating their domain group to the "To server exchange." This exchange is another direct model that forwards messages to the "Server Queue." The server subscribes to this queue to retrieve all the updated models from all clients. The server identifies the domain groups of retrieved messages using their topics as filters. It then aggregates the weights as per the framework shown in Figure 3, resulting in two updated models. These updated global models are then published to the broker to continue the process. Note that clients, the broker, and the central server are on separate nodes, connected through the Internet.

5.3. Hardware and software

Our setup includes four Raspberry Pi devices, each capable of running four clients, and a server that manages and distributes messages using a RabbitMQ broker. The server has a 12th Gen Intel(R) Core(TM) i9-12900F processor, an NVIDIA GeForce RTX 3070 Ti 8GB GDDR6X, and 16 GB of RAM. Each Raspberry Pi 4B model has a 4-core ARMv8 CPU and 4 GB of RAM. Network connectivity is provided by the University of Illinois Internet. For software, we use TensorFlow (2.10.0), Python (3.9.2), and the Raspberry Pi OS (64-bit) based on Debian Bullseye. We use the RabbitMQ broker with the Pika (1.3.2) library in Python for message routing between the central server and clients.

6. Case Studies

This section compares the FDG-CM method with multiple baseline methods to demonstrate its effectiveness. Additionally, we investigate the performance of FDG-CM under unbalanced data distributions and different client fractions. The training efficiency of FDG-CM in a real cloud manufacturing environment is also evaluated.

6.1. Experimental setting

We use 5-fold cross-validation and categorical cross-entropy as the base loss function. To assess the domain generalization of all the models, we evaluate them on data from unseen domains. Instead of random initialization, the server uses the "Kaiming normal" [45] to initialize the models' weights. As explained in Section 3, three datasets with domain groups M, S, and T are available. For simplicity, we concentrate on the collaboration between just two datasets for the FDG-CM method. However, we do not combine domain groups S and T, as they represent the same data but with different domains and classes. Therefore, we investigate the combinations of domain groups M vs. S and M vs. T.

The client distribution is as follows: One domain from each domain group is considered the target domain, while the other domains are equally distributed among three clients as source domains. Table A.1 in Appendix A illustrates this for domain group M, with Al-Al as the target domain. Each of the source domains (Al-Cu, Cu-Cu, Cu-Al) is randomly divided among three clients, with each client receiving about 67 data points, resulting in a total of 9 clients for domain group M. Tables A.2 and A.3 in Appendix A show a similar approach for other domain groups (S and T), resulting in 6 training clients per group, each with 30 data points. This distribution mirrors real-world scenarios where each client has access to data from only one configuration. The limited data availability and single-domain access pose significant challenges for the effective domain generalization of the model.

Table 2: Experimental Setup for Domain M and Domain S Configurations

Domain S / Domain M	Cu-Al	Cu-Cu	Al-Cu	Al-Al	Accuracy	Goal
Clean	Cl_CA	Cl_CC	Cl_AC	Cl_AA	Ave(Cl_*)	
Contam	Co_CA	Co_CC	Co_AC	Co_AA	Ave(Co_*)	DMGD, New, Worn
Polished	P_CA	P_CC	P_AC	P_AA	Ave(P_*)	
Accuracy	Ave(*_CA)	Ave(*_CC)	Ave(*_AC)	Ave(*_AA)		
Goal			N-N, W-N, N-W, W-W			

In all experiments, the same NN model presented in Section 5.1 serves as

Table 3: Experimental Setup for Domain M and Domain T Configurations

Domain T / Domain M	Cu-Al	Cu-Cu	Al-Cu	Al-Al	Accuracy	Goal
DMGD	D_CA	D_CC	D_AC	D_AA	Ave(D_*)	Clean, Contam, Polished
New	N_CA	N_CC	N_AC	N_AA	Ave(N_*)	
Worn	W_CA	W_CC	W_AC	W_AA	Ave(W_*)	
Accuracy	Ave(*_CA)	Ave(*_CC)	Ave(*_AC)	Ave(*_AA)		
Goal	N-N, W-N, N-W, W-W					

the basis for all training paradigms. To address the randomness in NN models, all experiments are repeated 5 times. We report the accuracy of the proposed method for each configuration, treating it as the target for one dataset and considering all possible target domain combinations for the other dataset, as depicted in Tables 2 and 3. For instance, in training domain group M vs. domain group T, to report the accuracy for the Al-Al as the target in domain group M, we conduct the experiment three times. Each time involves selecting a different configuration as the target domain in domain group T. This approach ensures a thorough evaluation of the FDG-CM method across a range of domain configurations. Consequently, for the Al-Al target in domain group M, each of the three target domain combinations from domain group T is tested five times, resulting in a total of 15 accuracy measurements. The reported accuracy represents the average and standard deviation of these tests. This is achieved by first calculating the mean and standard deviation for each of the three combinations separately, and then averaging these means and standard deviations across all three combinations. This process ensures a comprehensive and accurate reflection of the model's performance for the never-before-seen target.

We use random search to tune hyper-parameter values, exclusively based on the validation set from source domains, ensuring no data leakage from the target domain. This protocol facilitates a fair comparison among the methods [46]. The hyper-parameters for the FDG-CM method are set identically across both domain group combinations. Following this protocol, the first layer is selected as the base layer, with the next three layers designated as personalized layers. The mini-batch size is set to 8, and the learning rate to 0.0005, with one epoch selected for local training. Additionally, the proximal term and L2 regularization hyperparameters are determined to be 0.01 and 0.001, respectively. 150 server rounds are used for all experiments, as convergence for all clients is achieved within this number. Details on the hyperparameters for other models will be provided in their respective sections.

6.2. Performance comparison with other learning paradigms

This section aims to assess the domain generalization capability of the FDG-CM model in CM tasks for UMW. We compare FDG-CM with centralized learning (CL), individual learning (IL), and centralized transfer learning (CTL). In CL, data from all clients is combined for training, while in IL, each client uses only their own data. Given that FDG-CM involves knowledge transfer between datasets, we also study CTL, where the model is initially trained on one dataset and, after freezing the base layers (first layer in this case), it is fine-tuned with another dataset. In this study, the average performance of all clients trained individually is used to indicate the performance of IL. To ensure consistency, the same target data are employed across all models: IL, CL, CTL, and FDG-CM. These models are configured with a batch size of 8 and a learning rate of 0.0005. Each model underwent training for a total of 150 epochs.

6.2.1. Domain group M vs. domain group S

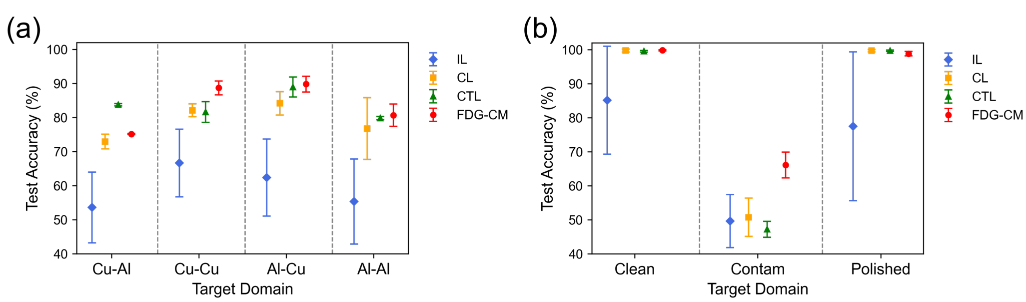

Figure 5: Performance comparison with IL, CL, and CTL for domain groups M vs. S: (a) tool classification accuracy for different material targets and (b) tool classification accuracy for different surface targets.

In this case study, we consider a collaboration between domain group M, with 4 material configurations, and domain group S, with 3 surface configurations. Selecting one configuration from each dataset as the target domain results in 12 different combinations, as depicted in Table 2. The task for both datasets is the tool condition classification, with domain group M having 4 classes and domain group S having 3 classes. Figure 5 shows that the FDG-CM method surpasses both IL and CL in nearly all instances, with an average improvement of 20.82% and 4.72%, respectively. Additionally, the FDG-CM method outperforms the CTL model by an average of 3.08%.

6.2.2. Domain group M vs. domain group T

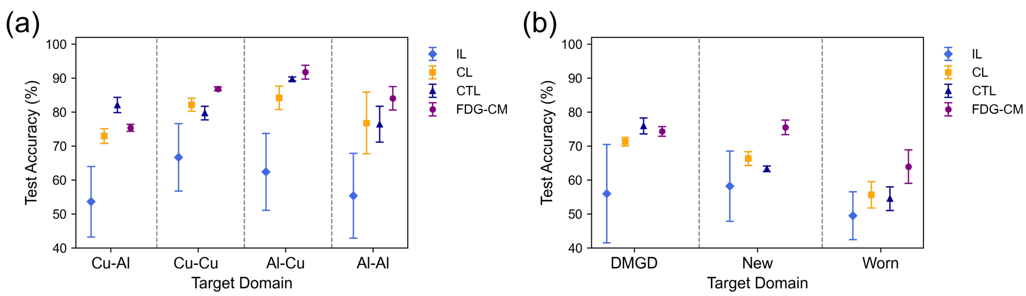

Figure 6: Performance comparison with IL, CL, and CTL for domain groups M vs. T: (a) tool classification accuracy for different material targets and (b) surface classification accuracy for different tool targets.

Table 3 shows the details of the collaboration between domain groups M and T. In this case study, the task in domain group M, which is tool condition classification, is completely different from the task in domain group T, which is surface condition classification. Figure 6 shows the comparative results, demonstrating that the FDG-CM method surpasses both IL and CL in all instances, with average improvements of 20.85% and 6.16%, respectively. Notably, in most cases, our method exhibits less variation in performance, suggesting enhanced robustness compared to these two paradigms. Furthermore, on average, the FDG-CM method outperforms the CTL paradigm with an improvement of 4.60%. This improvement shows the FDG-CM method's effectiveness in transferring knowledge among clients working on two distinct CM tasks.

6.3. Performance comparison with other FL methods

This section compares FDG-CM with state-of-the-art FL models including FedAvg, FedProx, and FedL2R. These FL methods use the same client distribution as mentioned previously for a consistent comparison. A batch size of 8 and a learning rate of 0.0005 are chosen for these three models. Additionally, for FedProx, the proximal term is set to 0.1, and for FedL2R, the regularization term is 0.01. All these models iterate for 150 server rounds.

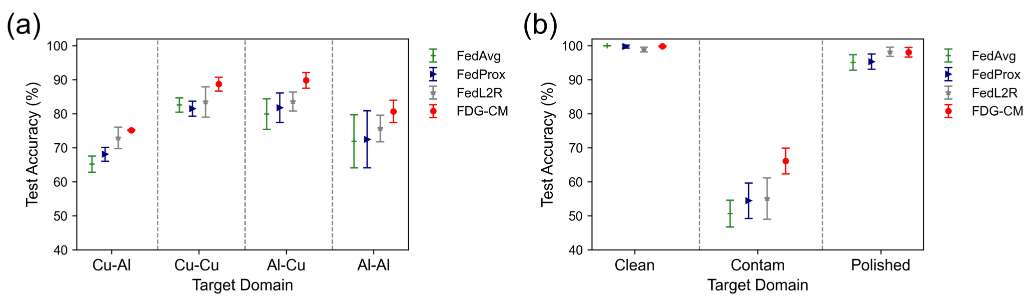

Figure 7: Performance comparison with other FL methods for domain groups M vs. S: (a) tool classification accuracy for different material targets and (b) tool classification accuracy for different surface targets.

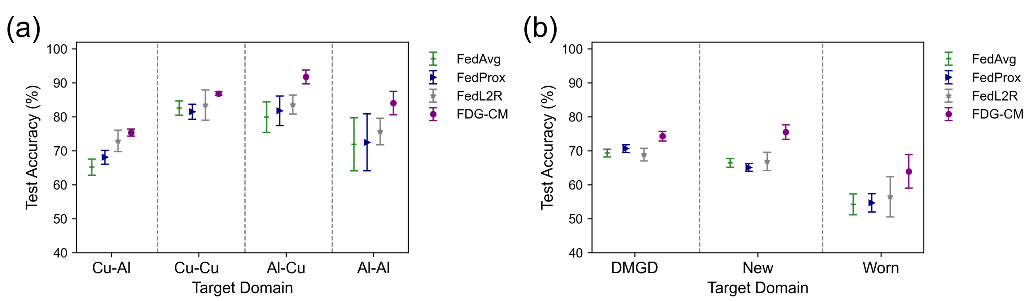

Figure 8: Performance comparison with other FL methods for domain groups M vs. T: (a) tool classification accuracy for different material targets and (b) surface classification accuracy for different tool targets.

Figures 7 and 8 present the comparative results for domain groups M vs. S and M vs. T, respectively. It is seen that FDG-CM outperforms FedAvg, FedProx, and FedL2R with average improvements of 8.08%, 7.19%, and 5.35%, respectively. It is worth noting that FDG-CM outperforms FedL2R, which is considered as a state-of-the-art domain generalization FL method, by a significant margin.

6.4. Performance comparison between balanced and unbalanced data distribution

This section further mimics industrial scenarios by introducing unbalanced data distributions among clients, which is a significant challenge in FL performance. We investigate how the FDG-CM model copes with this challenge using the domain groups M and S combination. To create an unbalanced distribution, we assign 30, 60, and 110 data points to the three clients of each training domain in domain group M, and 15, 25, and 50 data points to the clients in each training domain of domain group S. This aims to replicate a near-worst-case scenario in industrial settings.

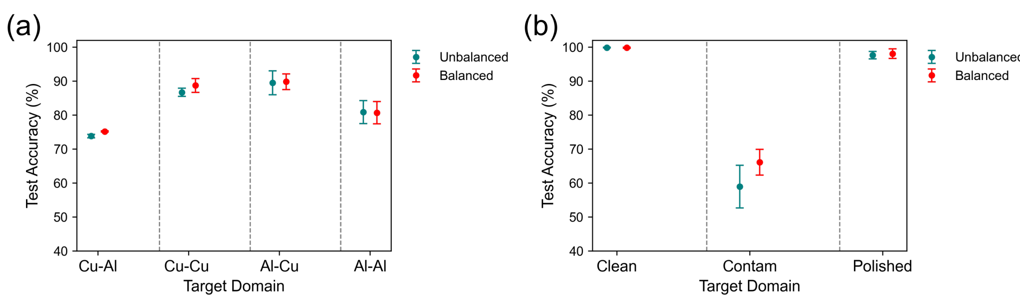

Figure 9: Balanced vs unbalanced performance comparison for the FDG-CM model in domain groups S vs M combinations: (a) tool classification accuracy for different material targets and (b) surface classification accuracy for different tool targets.

Figure 9a shows that the accuracy of the FDG-CM model under unbalanced distribution is similar to that of the balanced distribution, with less than a 2.05% drop, indicating robust performance for domain group M. However, Figure 9b shows that in domain group S, while the accuracy drops for the clean and polished domains are negligible, the Contam domain exhibits a significant drop of 7.17%.

6.5. *Performance comparison for different client fractions*

To evaluate the data efficiency of FDG-CM, the effect of varying client fractions on the training of the model is examined. For this purpose, we select a combination of domain groups M and S to conduct our investigation. In each server round, 1, 2, and all 3 clients in each source domain are randomly selected. This results in collaborations involving 3, 6, and 9 clients from domain group M, as well as 2, 4, and 6 clients from domain group S to represent client fractions of 1/3, 2/3, and 1, respectively.

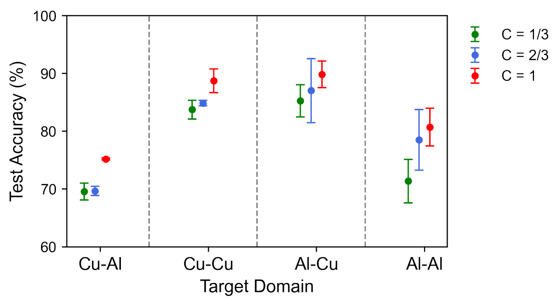

Figure 10: The FDG-CM's accuracy results for various material target domains in domain groups S vs M combination with different client fractions (C).

Figure 10 shows that within domain group M, FDG-CM's performance improves as the number of participating clients increases. This improvement is due to the larger and more diverse dataset available for training the model, consequently boosting model performance. Notably, the performance of the FDG-CM method with a client fraction of 1/3 is comparable to that of other baseline FL methods with a client fraction of 1, and even the CL method.

6.6. *Comparative time analysis of FL methodologies*

Given that the FDG-CM method develops at least two personalized models during the training phase, evaluating its time efficiency is essential. We select the combination of domain groups M and S for this purpose. The aim is to compare the time efficiency of the FDG-CM model with that of single-dataset FL frameworks, including FedAvg, FedProx, and a combination of FedL2R with FedProx, which employs a similar loss function to that of the FDG-CM method.

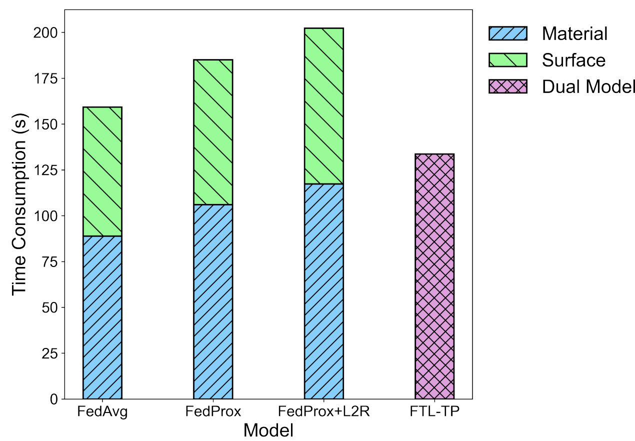

Figure 11: Time consumption for different FL models including FedAvg, FedProx, FedProx with L2R, and the FDG-CM Model for training two models for domain groups S and M.

We implement the edge-cloud architecture presented in Section 5.2. The time measurement is done from the moment the central server begins distributing the initial weights to clients until the completion of 10 server rounds, with a client fraction of $C = 1$. It is important to note that the proposed model outputs two distinct models, one for each of the domain groups M and S, whereas traditional single-dataset FL frameworks would output only one model for either domain groups M or S. Figure 11 compares the time consumption of different FL models. It shows that our proposed framework is less time-consuming compared to other FL approaches for training two separate models.

7. Discussion

7.1. CM performance

The results reported in Sections 6.2 and 6.3 lead to several important findings. First, individual clients do not have sufficient data to train their models effectively. This is shown by the significantly lower performance in CM for IL compared to the collaborative methods (CL, CTL, or any FL) used in this study. Second, the FDG-CM can solve the domain shift issue in CM by learning a shared, simplified representation across various domain spaces and using the proposed new loss function. This is evident by Figures 5–8, which show the FDG-CM's performance generally exceeds the performance of other FL paradigms, IL, and CL. Third, a comparison between CTL and the proposed method, as shown in Figures 5 and 6, shows that the FDG-CM surpasses CTL in most cases. This observation reveals that the enhancement in domain generalization by the proposed method is not solely due to the transfer of knowledge between datasets. The manner of this knowledge transfer and the application of generalization techniques, such as the L2R loss, play crucial roles as well.

Additionally, we evaluate the proposed method under unbalanced data distribution, which is a key challenge in FL settings. Figure 9 shows that FDG-CM yields similar results in both balanced and unbalanced data conditions in most cases. The decrease in accuracy for the Contam domain under unbalanced conditions, as shown in Figure 9b, might be attributed to the specific characteristics of the contaminated surface, such as the presence of oil drops, which introduce a wide range of variability. This agrees with previous studies [12], which found CM tasks under contaminated surface conditions to be more challenging. The model's increased sensitivity to unbalanced conditions could also be a potential reason for this drop in accuracy. Moreover, the limited number of data points available to some clients in domain group S, potentially as few as 15, may further affect the model's performance. However, it is important to note that the unbalanced accuracy for the Contam domain in domain group S still exceeds the performance of IL, CL, CTL, and other FL models under balanced conditions, as evidenced by comparing Figure 9b with Figures 5b and 7b.

7.2. Training efficiency

The efficiency of the proposed model is evaluated in terms of data and time efficiency. For data efficiency, Figure 10 shows that engaging more clients leads to improved performance, as the FL model is trained on a larger dataset. Comparing the performance with one-third of clients in Figure 10 to Figure 5 reveals that even limited collaboration among clients, one client from each source domain, significantly boosts the accuracy of models over IL. Moreover, this minimum of collaboration yields results comparable to those achieved by CL.

Additionally, comparing Figures 7 and 10 shows that the FDG-CM framework with a client fraction of 1/3, yields results that are comparable to or better than other FL approaches with a client fraction of 1. This suggests that FDG-CM, using only two clients from domain group S (60 data points) and three clients from domain group M (200 data points), can outperform state-of-the-art FL methods that use all 600 data points from domain group M. This improvement is likely due to efficient representation learning between clients across both datasets.

Figure 11 shows that our framework is more time-efficient in training models compared to traditional FL frameworks. For instance, the FedAvg framework, considered the simplest FL model, requires over 20% more training time. When using an FL framework with the same loss function as FDG-CM, the extra time needed increases to over 50%. Notably, within the single-dataset frameworks for domain groups M and S, there are 9 and 6 clients running on 3 and 2 edge devices, respectively. By combining these two groups of clients, the FDG-CM method includes a total of 15 clients implemented on 4 edge devices, which can raise the latency in the edge-cloud implementation. Despite this, FDG-CM is more time-efficient overall. Additionally, the training time for 10 server rounds is under two minutes, which is reasonable for industrial implementation.

Figure A.2 in Appendix A illustrates the convergence of the FDG-CM framework with and without the FedProx term. The FedProx term, designed to reduce local parameter disparities, significantly enhances the stability of convergence. As depicted in the figure, the model without the FedProx term exhibits considerable fluctuations and requires more time to converge, if it converges at all. This shows that using the FedProx term improves the overall training efficiency of the FDG-CM framework by stabilizing and accelerating the convergence process.

Based on these observations, the FDG-CM framework is both data-efficient and time-efficient. This combination makes it an effective solution for real-world manufacturing FL applications.

7.3. Limitations and future work

Our study highlights the importance of both data volume and integration approaches in the FL paradigm within industrial settings. It also emphasizes the need for additional model optimization to improve its ability to handle unbalanced data distributions, particularly in complex scenarios like the Contam domain. Future research should focus on adaptive learning strategies or domain-specific adjustments to address these challenges and enhance overall model effectiveness.

Additionally, our approach to privacy is foundational, serving as a solid base for further enhancements. FL clients face various security threats, such as data poisoning, model evasion, and data inference attacks, which could originate from either participating clients or a malicious server. In the field of FL, techniques such as differential privacy and homomorphic encryption are discussed for protecting client data. These methods for enhancing privacy should be examined and incorporated into our framework. Implementing these security measures would not only enhance data protection but also increase the trustworthiness and practicality of the FL paradigm in industrial settings.

8. Conclusion

The ever-changing and varied configurations in modern manufacturing, combined with limited data availability, have posed a significant challenge of domain shift in applying machine learning, especially in CM. To address this challenge, this paper presents a novel FDG-CM framework, enhancing domain generalization capabilities in FL. By learning a unified representation from the feature space, FDG-CM can adapt CM models for clients in distinct domain groups. Implementations on real-world UMW datasets demonstrate that FDG-CM outperforms other learning paradigms and state-of-the-art FL methods. Compared with baseline FL algorithms, FDG-CM shows a 5.35%–8.08% improvement in accuracy for domain generalization tasks. FDG-CM also achieves excellent performance in challenging scenarios involving unbalanced data distributions and limited client fractions. Moreover, the FDG-CM method is implemented on an edge-cloud architecture, showing both viability and efficiency in practice. Notably, the FDG-CM framework is readily extensible to various other industrial applications where challenges of data availability and data privacy are present simultaneously.

Acknowledgments

This research has been supported by the National Science Foundation under Grant Nos. 1944345, 2433484, and 2126246.

CRediT authorship contribution statement

Ahmadreza Eslaminia: Conceptualization, Methodology, Software, Formal analysis, Investigation, Validation, Visualization, Writing – original draft, Writing – review & editing. **Yuquan Meng**: Conceptualization, Formal analysis, Investigation, Data curation, Visualization, Writing – original draft, Writing – review & editing. **Klara Nahrstedt**: Conceptualization, Methodology, Resources, Investigation, Visualization, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Chenhui Shao**: Conceptualization, Methodology, Resources, Investigation, Visualization, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Appendix A.

Table A.1: Domain group M client distribution in case of taking Al-Al as target domain (N (New) - W (Worn))

Domains/Classes	TC1 (N-N)	TC2 (N-W)	TC3 (W-N)	TC4 (W-N)
Al-Al	Target Domains			
Al-Cu	200 data points in 3 training clients (about 67 each)			
Cu-Cu	200 data points in 3 training clients (about 67 each)			
Cu-Al	200 data points in 3 training clients (about 67 each)			

Table A.2: Domain group S client distribution in case of taking clean as target domain

Domain/Classes	DMGD	New	Worn
Clean	Target domain		
Contam	90 data points in 3 training clients (30 each)		
Polished	90 data points in 3 training clients (30 each)		

Table A.3: Domain group T client distribution in case of taking DMGD as target domain

Domain/Classes	Clean	Contam	Polished
DMGD	Target domain		
New	90 data points in 3 training clients (30 each)		
Worn	90 data points in 3 training clients (30 each)		

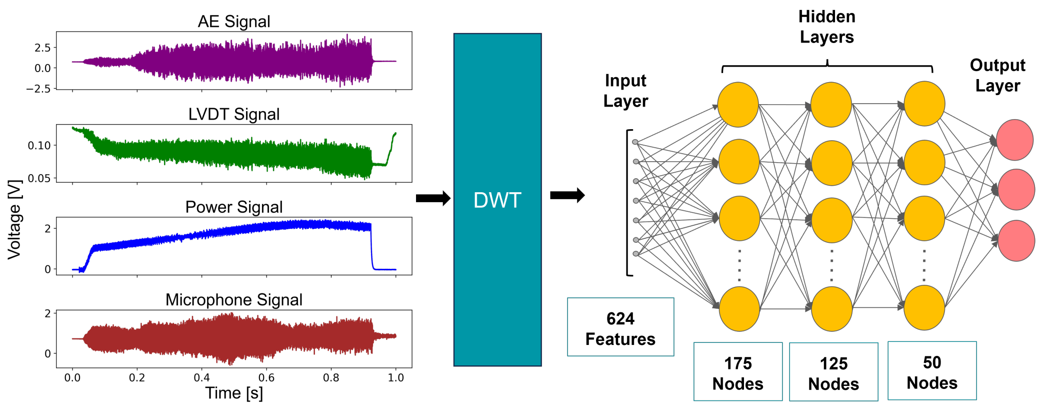

Figure A.1: Data preprocessing and classification procedure.

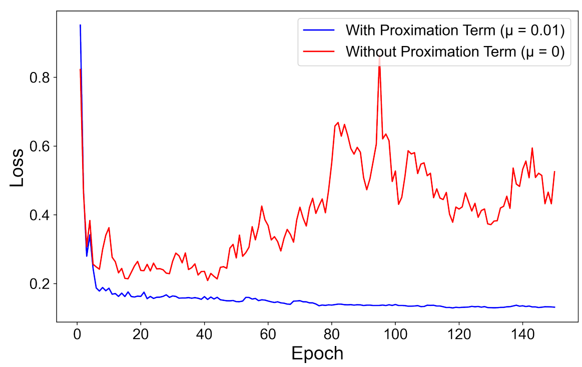

Figure A.2: Convergence of the FDG-CM method with and without the FedProx term, using Al-Cu as the target for domain groups M.

References

[1] Z. Ni, F. Ye, Ultrasonic spot welding of aluminum alloys: A review, Journal of Manufacturing Processes 35 (2018) 580–594.

[2] M. de Leon, H.-S. Shin, Review of the advancements in aluminum and copper ultrasonic welding in electric vehicles and superconductor applications, Journal of Materials Processing Technology 307 (2022) 117691.

[3] W. W. Cai, B. Kang, S. J. Hu, Ultrasonic Welding of Lithium-Ion Batteries, ASME Press, 2017.

[4] I. Balz, E. Abi Raad, E. Rosenthal, R. Lohoff, A. Schiebahn, U. Reisgen, M. Vorländer, Process monitoring of ultrasonic metal welding of battery tabs using external sensor data, Journal of Advanced Joining Processes 1 (2020) 100005.

[5] Y. Meng, C. Shao, Physics-informed ensemble learning for online joint strength prediction in ultrasonic metal welding, Mechanical Systems and Signal Processing 181 (2022) 109473.

[6] C. Shao, T. Hyung Kim, S. Jack Hu, J. Jin, J. A. Abell, J. Patrick Spicer, Tool wear monitoring for ultrasonic metal welding of lithium-ion batteries, Journal of Manufacturing Science and Engineering 138 (5) (2016) 051005.

[7] R. Nunes, K. Faes, S. De Meester, W. De Waele, A. Kubit, Influence of welding parameters and surface preparation on thin copper–copper sheets welded by ultrasonic welding process, The International Journal of Advanced Manufacturing Technology 123 (1-2) (2022) 373–388.

[8] Q. Nazir, C. Shao, Online tool condition monitoring for ultrasonic metal welding via sensor fusion and machine learning, Journal of Manufacturing Processes 62 (2021) 806–816.

[9] Y. Wu, Y. Meng, C. Shao, End-to-end online quality prediction for ultrasonic metal welding using sensor fusion and deep learning, Journal of Manufacturing Processes 83 (2022) 685–694.

[10] K.-C. Lu, Y. Meng, Z. Dong, C. Shao, Online cost-effective classification of mixed tool and material conditions in ultrasonic metal welding: Towards integrated monitoring and control, in: International Manufacturing Science and Engineering Conference, Vol. 87240, American Society of Mechanical Engineers, 2023, p. V002T07A007.

[11] S. J. Hu, Evolving paradigms of manufacturing: From mass production to mass customization and personalization, Procedia Cirp 7 (2013) 3–8.

[12] Y. Meng, K.-C. Lu, Z. Dong, S. Li, C. Shao, Explainable few-shot learning for online anomaly detection in ultrasonic metal welding with varying configurations, Journal of Manufacturing Processes 107 (2023) 345–355.

[13] B. Tian, A. Eslaminia, K.-C. Lu, Y. Wang, C. Shao, K. Nahrstedt, Weldmon: A cost-effective ultrasonic welding machine condition monitoring system, in: 2023 IEEE 14th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), IEEE, 2023, pp. 0310–0319.

[14] M. Mehta, C. Shao, Adaptive sampling design for multi-task learning of gaussian processes in manufacturing, Journal of Manufacturing Systems 61 (2021) 326–337.

[15] M. Mehta, C. Shao, Federated learning-based semantic segmentation for pixel-wise defect detection in additive manufacturing, Journal of Manufacturing Systems 64 (2022) 197–210.

[16] M. Mehta, S. Chen, H. Tang, C. Shao, A federated learning approach to mixed fault diagnosis in rotating machinery, Journal of Manufacturing Systems 68 (2023) 687–694.

[17] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: Concept and applications, ACM Transactions on Intelligent Systems and Technology (TIST) 10 (2) (2019) 1–19.

[18] Y. Li, X. Wang, R. Zeng, P. K. Donta, I. Murturi, M. Huang, S. Dustdar, Federated domain generalization: A survey, arXiv preprint arXiv:2306.01334.

[19] G. Wu, S. Gong, Collaborative optimization and aggregation for decentralized domain generalization and adaptation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 6484–6493.

[20] R. Y. Zhong, L. Wang, X. Xu, An iot-enabled real-time machine status monitoring approach for cloud manufacturing, Procedia Cirp 63 (2017) 709–714.

[21] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, R. X. Gao, Deep learning and its applications to machine health monitoring, Mechanical Systems and Signal Processing 115 (2019) 213–237.

[22] Z. Huang, W. Li, J. Zhu, L. Wang, Cross-domain tool wear condition monitoring via residual attention hybrid adaptation network, Journal of Manufacturing Systems 72 (2024) 406–423.

[23] C. Domínguez-Monferrer, J. Fernández-Pérez, R. De Santos, M. Miguélez, J. Cantero, Machine learning approach in non-intrusive monitoring of tool wear evolution in massive cfrp automatic drilling processes in the aircraft industry, Journal of Manufacturing Systems 65 (2022) 622–639.

[24] C. Ferreira, G. Gonçalves, Remaining useful life prediction and challenges: A literature review on the use of machine learning methods, Journal of Manufacturing Systems 63 (2022) 550–562.

[25] Z. Hou, C. Lee, Y. Lv, K. Keung, Fault detection and diagnosis of air brake system: A systematic review, Journal of Manufacturing Systems 71 (2023) 34–58.

[26] M. Yin, S. Zhuo, L. Xie, L. Chen, M. Wang, G. Liu, Online monitoring of local defects in robotic laser additive manufacturing process based on

a dynamic mapping strategy and multibranch fusion convolutional neural network, Journal of Manufacturing Systems 71 (2023) 494–503.

[27] E. B. Schwarz, F. Bleier, F. Guenter, R. Mikut, J. P. Bergmann, Improving process monitoring of ultrasonic metal welding using classical machine learning methods and process-informed time series evaluation, Journal of Manufacturing Processes 77 (2022) 54–62.

[28] M. Russell, P. Wang, Maximizing model generalization for machine condition monitoring with self-supervised learning and federated learning, Journal of Manufacturing Systems 71 (2023) 274–285.

[29] T. Müller, N. Gärtner, N. Verzano, F. Matthes, Barriers to the practical adoption of federated machine learning in cross-company collaborations., in: ICAART (3), 2022, pp. 581–588.

[30] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, S. Liu, Efficient and privacy-enhanced federated learning for industrial artificial intelligence, IEEE Transactions on Industrial Informatics 16 (10) (2019) 6532–6542.

[31] L. Li, Y. Fan, M. Tse, K.-Y. Lin, A review of applications in federated learning, Computers & Industrial Engineering 149 (2020) 106854.

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

[33] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, Proceedings of Machine learning and systems 2 (2020) 429–450.

[34] A. T. Nguyen, P. Torr, S. N. Lim, Fedsr: A simple and effective domain generalization method for federated learning, Advances in Neural Information Processing Systems 35 (2022) 38831–38843.

[35] L. Zhang, X. Lei, Y. Shi, H. Huang, C. Chen, Federated learning with domain generalization, arXiv preprint arXiv:2111.10487.

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

[37] T. Berghout, M. Benbouzid, T. Bentrcia, W. H. Lim, Y. Amirat, Federated learning for condition monitoring of industrial processes: A review on fault diagnosis methods, challenges, and prospects, Electronics 12 (1) (2022) 158.

[38] J. Ahn, Y. Lee, N. Kim, C. Park, J. Jeong, Federated learning for predictive maintenance and anomaly detection using time series data distribution shifts in manufacturing processes, Sensors 23 (17) (2023) 7331.

[39] Y. Mansour, M. Mohri, J. Ro, A. T. Suresh, Three approaches for personalization with applications to federated learning, arXiv preprint arXiv:2002.10619.

[40] I. Kevin, K. Wang, X. Zhou, W. Liang, Z. Yan, J. She, Federated transfer learning based cross-domain prediction for smart manufacturing, IEEE Transactions on Industrial Informatics 18 (6) (2021) 4088–4096.

[41] S. Chung, R. Al Kontar, Federated condition monitoring signal prediction with improved generalization, IEEE Transactions on Reliability.

[42] J.-S. Seo, S. Y. Beck, Ultrasonic deposit junction characteristic evaluation of metal sheets al/al and al/cu, Journal of the Korean Institute of Metals and Materials 49.

[43] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, S. Choudhary, Federated learning with personalization layers, arXiv preprint arXiv:1912.00818.

[44] D. Dossot, RabbitMQ essentials, Packt Publishing Ltd, 2014.

[45] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.

[46] I. Gulrajani, D. Lopez-Paz, In search of lost domain generalization, arXiv preprint arXiv:2007.01434.