ASME Journal of Manufacturing Science and Engineering
Online journal at:
https://asmedigitalcollection.asme.org/manufacturingscience

Rong Lei
Department of Industrial and Systems Engineering,
Rutgers University,
New Brunswick,
Piscataway, NJ 08854
e-mail: rl839@scarletmail.rutgers.edu

Y. B. Guo
Fellow ASME
New Jersey Advanced Manufacturing Institute,
Rutgers University,
New Brunswick,
Piscataway, NJ 08854
e-mail: yuebin.guo@rutgers.edu

Jiwang Yan
Department of Mechanical Engineering,
Keio University,
Yokohama 223-8522, Japan
e-mail: yan@mech.keio.ac.jp

Weihong "Grace" Guo[1]
Mem. ASME
Department of Industrial and Systems Engineering,
Rutgers University,
New Brunswick,
Piscataway, NJ 08854
e-mail: wg152@soe.rutgers.edu

Bridging Data Gaps: A Federated Learning Approach to Heat Emission Prediction in Laser Powder Bed Fusion

Deep learning has impacted defect prediction in additive manufacturing (AM), which is important to ensure process stability and part quality. However, its success depends on extensive training, requiring large, homogeneous datasets—remaining a challenge for the AM industry, particularly for small- and medium-sized enterprises (SMEs). The unique and varied characteristics of AM parts, along with the limited resources of SMEs, hamper data collection, posing difficulties in the independent training of deep learning models. Addressing these concerns requires enabling knowledge sharing from the similarities in the physics of the AM process and defect formation mechanisms while carefully handling privacy concerns. Federated learning (FL) offers a solution to allow collaborative model training across multiple entities without sharing local data. This article introduces an FL framework to predict section-wise heat emission during laser powder bed fusion (LPBF), a vital process signature. It incorporates a customized long short-term memory (LSTM) model for each client, capturing the dynamic AM process's time-series properties without sharing sensitive information. Three advanced FL algorithms are integrated—federated averaging (FedAvg), FedProx, and FedAvgM—to aggregate model weights rather than raw datasets. Experiments demonstrate that the FL framework ensures convergence and maintains prediction performance comparable to individually trained models. This work demonstrates the potential of FL-enabled AM modeling and prediction where SMEs can improve their product quality without compromising data privacy.
[DOI: 10.1115/1.4065888]

Keywords: additive manufacturing, small- and medium-sized enterprises, data sharing, federated learning, inspection and quality control, sensing, monitoring and diagnostics

1 Introduction

Metal additive manufacturing (AM) has been widely used in industries, proving its success across various enterprises [1]. Its advantages in producing complex geometries and customized parts have been a game-changer, offering rapid prototyping and cost-effective production of complex parts [2]. Despite these advantages, metal AM faces challenges related to achieving consistent stability in production. This instability can be traced to the intricate nature of material deposition, which is inherently complex and dynamic, leading to the formation of defects during the process. To ensure process stability and print quality, it is crucial to detect anomalies promptly. Research has been directed toward mitigating the impact of defects, focusing on approaches such as process monitoring, anomaly detection, and defect prediction [3–5].

Advanced sensing in AM provides rich, timely, and in situ data, enabling data-driven approaches for process monitoring. Machine learning (ML) and deep learning (DL) have been widely recognized as effective in enhancing predictive accuracy and operational efficiency in these applications [6]. However, ensuring high prediction performance requires meticulous training of these models, which typically involves large, homogeneous datasets. Homogeneous data consist of similar, uniform elements, while heterogeneous data comprise diverse elements from multiple sources. In AM, sensing data are usually homogeneous when the process is identical (e.g., same material and same process setting), whereas heterogeneous data arise when data exhibit variability in these aspects due to customized designs.

The requirement of homogeneous data poses challenges to the training of robust and high-performing ML and DL models for small- and medium-sized enterprises (SMEs) in the AM industry [7]. These SMEs often make custom parts, leading to different process settings each time and resulting in heterogeneous datasets that usually vary by design [6]. This variability makes it difficult, and sometimes even infeasible, for them to gather large, homogeneous datasets necessary for training deep learning models. The customized design objects in AM usually have much smaller data volumes compared to the mass-produced objects [8]. Furthermore,

[1]Corresponding author.

Manuscript received March 6, 2024; final manuscript received June 27, 2024; published online July 23, 2024. Assoc. Editor: Jaydeep Karandikar.

limited by time and budgets, SMEs are often restricted to a small range of designs, further reducing their data collection.

One possible solution to address the data scarcity issue among SMEs is to leverage knowledge transfer methods. For instance, transfer learning has been developed for deep learning models to use pretrained models to facilitate training, thus mitigating the data scarcity issue while maintaining model performance [9]. Nevertheless, transfer learning relies mainly on centralized training data and powerful hardware. The effectiveness of pretrained models is also critical, and careful measures are needed to avoid negative transfer and ensure good performance. Knowledge distillation is another popular method for knowledge transfer. Originally designed for model compression [10], the method distills knowledge from complex model architectures into simpler ones, thereby improving training efficiency. Recently, knowledge distillation-based methods have been adapted in the manufacturing sector for predictive models on defect detection [11]. Multitask learning also leverages shared knowledge, effectively capitalizing on the commonalities across tasks by using insights gained from one task to boost performance and generalization in others [12].

The abovementioned methods facilitate knowledge transfer, enabling its application in manufacturing. However, even though some aspects of AM processes and defect formation are similar across different scenarios [13], which could help in knowledge sharing, privacy concerns about disclosing information such as part geometries, process parameters, and quality specifications are significant obstacles [14]. These concerns prevent data sharing and limit SMEs from achieving more robust model training. As a result, models based on data from a single design type are often not useful for other designs, making it expensive and impractical for SMEs to develop new models for every different production scenario. Yet, few studies have investigated the development of models that can handle limited data volume and heterogeneous data while ensuring data privacy for SMEs in AM [15]. Therefore, there is a pressing need for a paradigm that addresses data privacy concerns effectively. It would be particularly advantageous if this paradigm could also enhance its knowledge transfer capabilities.

Inspired by the success of federated learning (FL) in various fields such as the Internet of Things [16], banking [17], and health care [18], we propose to leverage FL to tackle SMEs' challenges of data scarcity in AM, especially in concerns about knowledge sharing of heterogeneous data without compromising data privacy. These concerns arise from the lack of sufficient homogeneous datasets, which are essential for training machine-learning models effectively. FL is an innovative machine-learning method that enables several participants to train machine-learning models through heterogeneous data collaboratively without sharing their locally stored data [19]. Figure 1 is the diagram of the proposed FL framework. The framework consists of four components. First, the central server model initializes a machine-learning model and broadcasts the model weights to all local client models. Second, each local client assigns the weights to the local model and trains based on the local storage data. Third, clients update their local model weights to the central server. Fourth, the global model aggregates the local model weights and continues to the next updating round, until the global model reaches its convergence.

FL addresses the data privacy issue through its mechanism of sharing only model weights during training. Although various FL algorithms have shown success in other industries, their application in AM remains underexplored. This highlights the need for effectively selecting and refining the appropriate FL algorithms across diverse AM designs to improve model accuracy [20]. There is limited existing work in AM so far. Notable examples include Mehta and Shao [21], who formulated an FL methodology with U-net for semantic segmentation, and Truong et al. [22], who incorporated FL with time-series data in industrial control systems. However, the general averaging-based FL paradigm encounters significant challenges when applied in AM due to the unique printing configurations of AM. Consequently, exploring the usage of heterogeneous datasets has become a prominent topic. The knowledge transfer techniques reviewed above can be effectively integrated with FL in an orthogonal manner. These techniques are particularly useful in mitigating data heterogeneity issues by leveraging knowledge learned across various SMEs. When combined with FL, these techniques improve the local model's personalized performance. Personalized FL provides significant customization potential throughout the training process, from the architecture of the local model and communication hierarchy to the aggregation method [23,24]. Efforts have been made to balance local client model performance with the efficiency of global training. For example, Putra et al. utilized transfer learning to build a hierarchical FL framework for efficient training [25], and Shi et al. incorporated knowledge distillation with FL for online process monitoring [26]. Another emerging topic involves enriching model input with multimodal datasets as tensors [27,28]. These techniques enhance local model performance but also introduce challenges related to data acquisition and raise concerns about overfitting in increasingly complex models and architectures. Currently, there is no consensus paradigm for personalized FL. Considering the challenges of overfitting and complexity in personalized frameworks, this article focuses on the generic performance of FL models to ensure broader applicability in AM scenarios.

Motivated by the need to further explore the benefits of knowledge sharing across SMEs in AM, the objective of this article is to develop an FL approach to improve the accuracy of defect prediction. Our approach predicts heat emission readings captured through pyrometry, during the printing process of the laser powder bed fusion (LPBF). In LPBF, a high-power laser selectively melts and fuses powder materials layer by layer, printing the product. Furthermore, studies have attempted to incorporate the

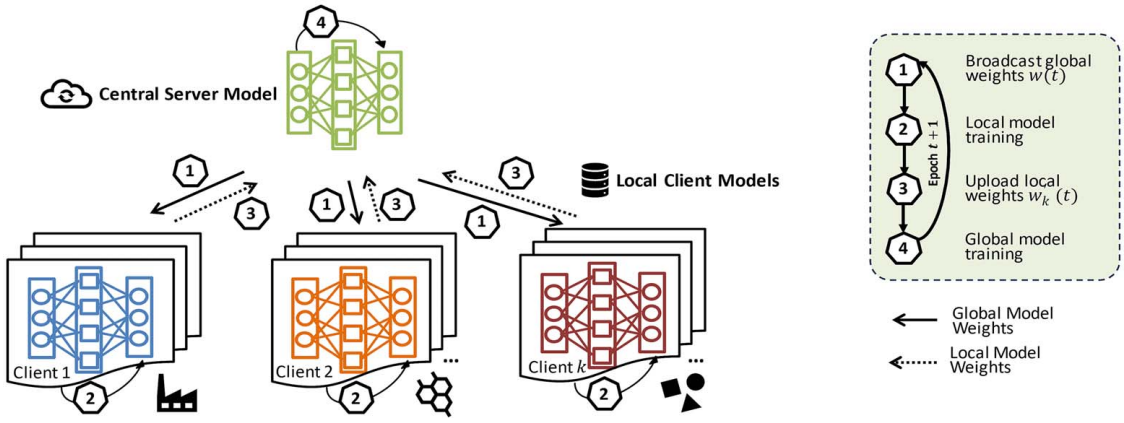

Fig. 1 Overview of the federated learning framework

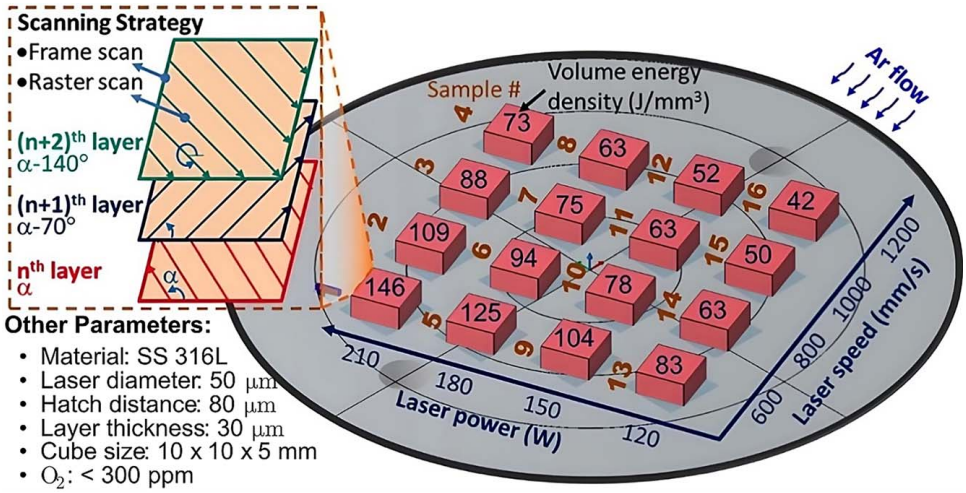

Fig. 2 Printing layout and parameter settings in the experiment [33]

physics of AM processes and domain knowledge to improve the interpretability of machine-learning and deep-learning models [29–31]. Incorporating physics insights with FL can be a highly effective strategy for heterogeneous scenarios in AM. The framework explores the relationship between emission readings and part quality under different process parameter scenarios. Each client, mimicking an SME, utilizes a long short-term memory (LSTM) model with a tailored loss function. The model is used to capture the time-series properties under the dynamic AM process. A global LSTM model [32] is developed and aggregates weights from these local models within the FL framework, enabling knowledge sharing without direct data transfer. Three state-of-the-art FL aggregation algorithms (federated averaging (FedAvg), FedProx, and FedAvgM) are employed, and their effectiveness is assessed through a series of experiments.

The key contributions of our article are as follows: (1) it is empirically demonstrated that custom loss functions enhance FL model performance; (2) a thorough examination of the impact of data heterogeneity on FL model training is provided; and (3) scenario of minimal client participation in FL is explored. The findings have direct implications for AM, demonstrating how FL can be effectively employed in scenarios where SMEs face challenges due to limited data availability or client participation. This has significant practical benefits for the AM industry, where data sensitivity is a common concern among SMEs.

The remainder of this paper is organized as follows. Section 2 introduces the data, the preprocessing method, and the FL methodology. Section 3 includes the custom loss function, the local LSTM model, and the FL architecture for emission prediction. Section 4 presents three case studies to demonstrate the effectiveness of FL-based architecture with custom LSTM models. Section 5 concludes the article and discusses directions for future research.

2 Data Description

2.1 Data Collection. In this study, the in situ pyrometry sensing data were obtained from an LPBF experiment as shown in Fig. 2, utilizing an AconityMINI machine. The machine is equipped with a laser source of up to 400 W and a spot diameter of 50 μm. This approach builds upon methodologies in prior work [33]. Blocks were fabricated from stainless steel (SS 316L) powder under a controlled argon atmosphere to maintain consistent build conditions. A total of 16 blocks were printed with dimensions of 10 mm × 10 mm × 5 mm, each consisting of 166 layers with a layer thickness of 30 μm. The configurations used for printing included laser powers of 120, 150, 180, and 210 W and laser speeds of 600, 800, 1000, and 1200 mm/s, as shown in Fig. 2. Each block was marked with a sample number ranging from 1 to 16. All experiments were conducted in an argon atmosphere (O_2 < 300 ppm) with continuous recirculation to remove metal vapor and condensate. Each block was subjected to a consistent scanning strategy involving a raster scan for the interior, followed by a frame scan. To capture thermal emissions from the melt pool, two calibrated pyrometers, coaxial with the process laser, were employed at a frequency of 100 kHz. Each layer yielded an estimated 300,000 emission readings.

2.2 Data Preprocessing. Raw emission readings from in situ pyrometry sensing were preprocessed using methods developed in the authors' previous work [34]. Initial steps involved noise attenuation techniques from the readings through methods like radius-based clustering, visual inspection, and automation. The radius-based clustering technique defines a radius to classify isolated points within this boundary as noise, effectively distinguishing actual data points from noise. Visual inspection is conducted manually by trained personnel, relying on human observations to identify and remove any outliers, thereby enhancing the cleanliness of the data. In addition, automated methods are employed to detect and correct inconsistencies in the data, such as unexpected noise paths. The integration of these approaches ensures that the emission data were thoroughly cleaned for the subsequent steps. After noise attenuation, the cleaned data coordinates were transformed to a new alignment by rotating (x, y) coordinates 20 deg clockwise. The emission readings were then systematically organized in a sequential array based on their (x, y) locations and recorded order. To improve computational efficiency, the cleaned emission readings were further segmented. Emission sequences from each printed layer block were segmented into equal-sized sections, with each section represented by its average emissions and coordinates in a new spatial map. This approach, illustrated in Fig. 3, effectively reduced the dataset size while preserving the essential characteristics of the emission data. Each segmented section was represented by its average emission reading, and its location was remapped using the section's average $(\bar{x}, \bar{y})$ coordinates. This segmentation led to a data scale reduction of approximately 1000 times.

As proposed in the previous work, both physical and statistical features were employed as model inputs [32]. Physical features encompassed predefined manufacturing parameters including laser power, laser speed, energy density, and scanning phase. Additionally, sequential information, such as layer number, was derived from the dataset. Furthermore, the section's average coordinates have been incorporated as physical features. These factors capture the cumulative thermal history, vital for understanding the LPBF process outcome. Statistical features (min, max, average emissions,

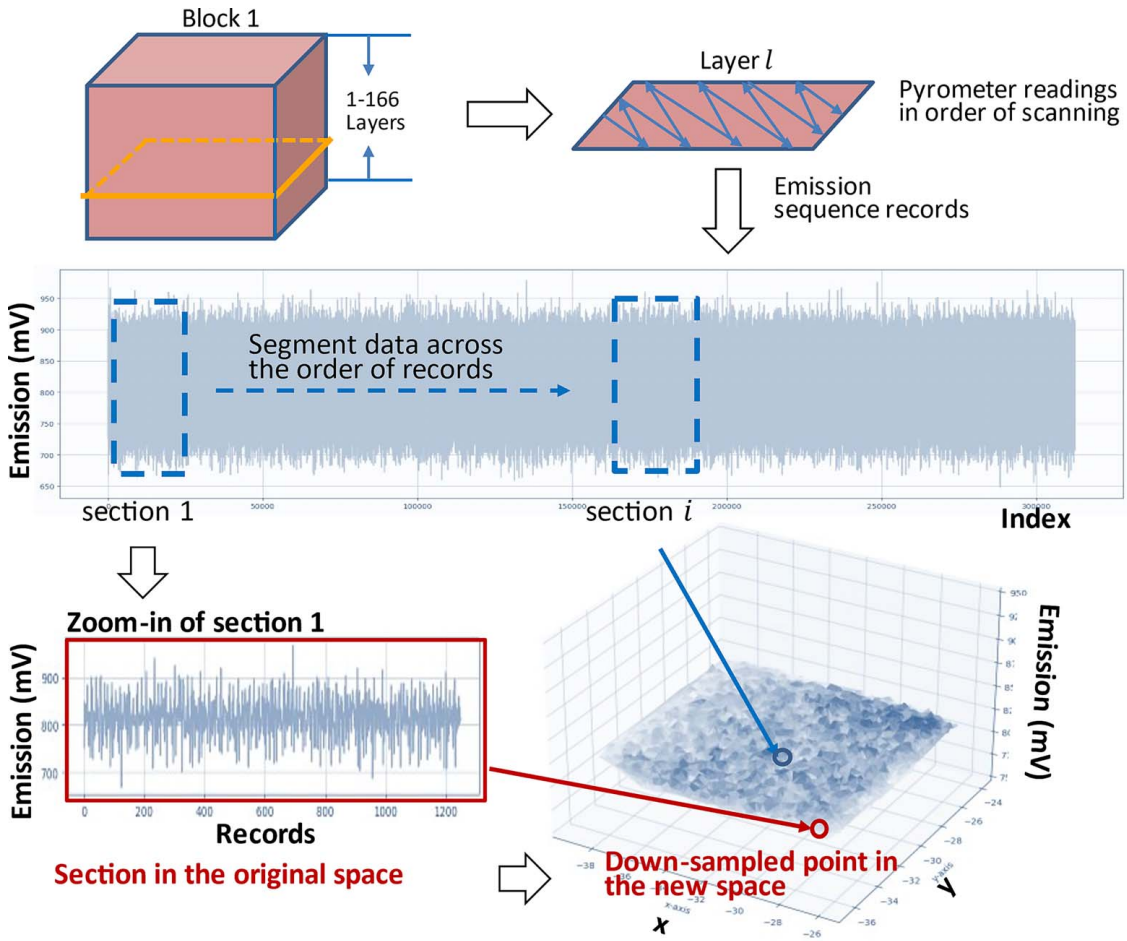

Fig. 3 Data segmentation process [34]: emission sequences from printed layers of each block are segmented into equal-sized sections and represented by their average emissions and coordinates in a new spatial mapping

$25^{th}/75^{th}$ percentiles, and standard deviation), extracted from segmented sections, offer insights into the emission distribution within a segmented section. Selected based on their descriptive potential, these features provide physics insights into section-wise variations and are instrumental in emission prediction.

2.3 Client Data Preparation. In the real world, datasets are often heterogeneous due to varying process settings and customized parts. To mimic this situation, representing different SMEs with their unique configurations and data, we generate multiple clients in preparation for the model training process under the FL paradigm from the preprocessed emission readings. As described in Sec. 2.1, a total of 16 distinct block samples were printed using various combinations of laser power and laser speed. Therefore, this variability allows us to categorize the data by their corresponding printing configurations, thereby simulating the inherent data heterogeneity across clients in an FL setting. Subsequently, the data from 16 blocks are randomly grouped into eight clients, with each training with data from two blocks. Forming clients this way usually leads to nonindependently and identically distributed (i.e., non-IID) data between the clients, while the diverse printing configurations within each client's data are preserved, as each client has two distinct parameter combinations. The data amount between clients is kept comparable, making no client dominant over the others. This can reduce the representativeness of individual clients concerning the entire dataset.

Figure 4(*a*) presents the non-IID data of sectional average emission (mV) across clients. It is evident from the figure that the data for each client appear to be a combination of two normal distributions, each one centered around the mean value of the two blocks of data. This pattern indicates heterogeneity within each client's dataset. The heterogeneity is also quantitatively indicated by the Kullback–Leibler divergence (KLD) value, which is calculated from the Gaussian kernel density estimation for each client's data distribution. The KLD can serve as a simple yet direct index to reflect the divergence between two clients. The KLD values shown in the legend of Fig. 4(*a*) are divergence between each client and client 8, with larger values indicating more divergence, less similar distribution curves, and more differences in the boxplots. Boxplots in Fig. 4(*b*) reveal the variations in the printing parameters of each dataset. The emissions span a range from 820 mV to 840 mV for most data points. Some points reach above 900 mV. This spread of data points indicates the diversity encountered across clients.

3 Methods

3.1 LSTM Model. For the construction of local client models, as depicted in Fig. 1, our approach employs a long short-term memory architecture. LSTM is a variant of recurrent neural networks that is renowned for its proficiency in processing and predicting sequential data [32]. A one-layer LSTM model is selected as the baseline of this study for its balance between training efficacy and computational cost. This model architecture is consistently used across all clients. This model takes a sequence of $\tau = 10$ section records $z_i \in \mathbb{R}^{\tau \times d}$ as input, sliding from section i to section $i-\tau$, with each section including a total of $d = 13$ features (including 7 physical and 6 statistical features) as described in Sec. 2.2. Specifically, the physical features include laser power, laser speed, energy density, scanning phase, sequential layer number, and x and y coordinates in the new space. The statistical features consist of descriptive features of sections, such as minimum emissions, maximum

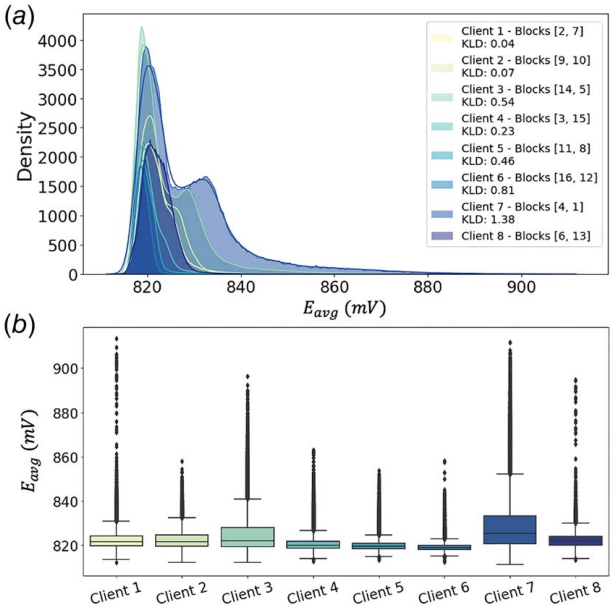

Fig. 4 (*a*) Client emission data density distributions with Kullback–Leibler divergence (KLD) to client 8. (*b*) Client emission data boxplots.

emissions, average emissions, standard deviation of emissions, and the 25th and 75th percentile emissions. This input structure captures the temporal dependencies inherent in the time-series data due to the heat transfer characteristics across layers [35]. Subsequently, the model implements a conventional LSTM layer, whose number of hidden matches the output size. A dropout rate of 0.5 is applied to mitigate the overfitting issue. The model's objective is to predict the $s=6$ statistical features $\boldsymbol{y}_{i+1} \in \mathbb{R}^{1\times s}$ for the subsequent section $i+1$, thereby enabling the prediction of thermal trends. The network is initialized with the "He normal method" [36], and the ReLU function is employed as the activation function. Proposed by Kaiming He, the He normal method is a weight initialization technique designed to alleviate the vanishing or exploding gradient problems in neural networks. It initializes network weights according to a Gaussian distribution with zero mean and a variance calculated based on the number of input and output units. This method ensures more stable and efficient training for neural networks, particularly when paired with the ReLU function. The mean squared error (MSE) is utilized as the primary loss function. Additionally, to refine the training dynamics further, we define a custom loss function tailored to align closely with the learning objectives.

3.2 Custom Loss Function. In the context of the loss function design, let N denote the total number of samples and our primary loss function L_{base} can be represented as

$$L_{\text{base}} = \frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{y}_{\text{pred},i} - \boldsymbol{y}_{\text{true},i}\|^2 \tag{1}$$

where the function computes the MSE between the actual statistics for each section and the predicted statistics by the LSTM model across all samples. This function gives equal weights to all features in $\boldsymbol{y}$. However, this standard MSE method may underemphasize errors in pivotal features such as the sectional average emission value E_{avg}. This feature is critical as it encapsulates the major emission characteristics for subsequent sections, and it is beneficial to illustrate the trends within sequential data. In the custom loss function L_{custom}, we address this limitation in L_{base}, to ensure that the model pays attention to E_{avg} while also considering the prediction accuracy of other statistical features.

As highlighted by the small variation range on E_{avg} in Fig. 4(*b*), the baseline loss function L_{base} may not adequately address the variability in predictions for this feature, since every $\boldsymbol{y}_{\text{pred}}$ column in the loss calculation is treated with equal weight. Therefore, we emphasize the impact of E_{avg} among other features in the output to better reflect its significance in predictive performance. Specifically, we assign an increased weight for E_{avg} in $\boldsymbol{y}_{\text{pred}}$ as $W_{E_{\text{avg}}}$, and we consider the variance penalty $P_{E_{\text{avg}}}$ to the E_{avg} feature when the training model's predictions converge around the average emission values. This usually reflects the nature of the printing performance of the 3D printers, which tend to maintain a stable temperature, leading to less variation in emission values. L_{custom} is defined as follows:

$$W_{E_{\text{avg}}} = \lambda * \frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{y}_{\text{pred},i,E_{\text{avg}}} - \boldsymbol{y}_{\text{true},i,E_{\text{avg}}}\|^2 \tag{2}$$

$$P_{E_{\text{avg}}} = \begin{cases} \gamma & \text{if } \sigma(\boldsymbol{y}_{\text{pred},E_{\text{avg}}}) < \theta \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$L_{\text{custom}} = L_{\text{base}} + W_{E_{\text{avg}}} + P_{E_{\text{avg}}} \tag{4}$$

In Eq. (2), $\boldsymbol{y}_{\text{pred},i,E_{\text{avg}}}$ refers to the predicted value of the E_{avg} column for the ith data point in the output matrix $\boldsymbol{y}_{\text{pred}}$. Here, i denotes the index of the data point within the dataset. λ, θ, and γ are hyperparameters that control the impact of MSE and penalty level for a shortfall in prediction variance of E_{avg}, respectively. A larger λ gives more emphasis on the accuracy of the specific feature, and a smaller value reduces its influence within the overall loss calculation. The threshold θ is defined for the term $\sigma(\boldsymbol{y}_{\text{pred},E_{\text{avg}}})$, which stands for the standard deviation of the E_{avg} column in the output $\boldsymbol{y}_{\text{pred}}$. The variance penalty is applied when the standard deviation falls below this threshold. It prevents the

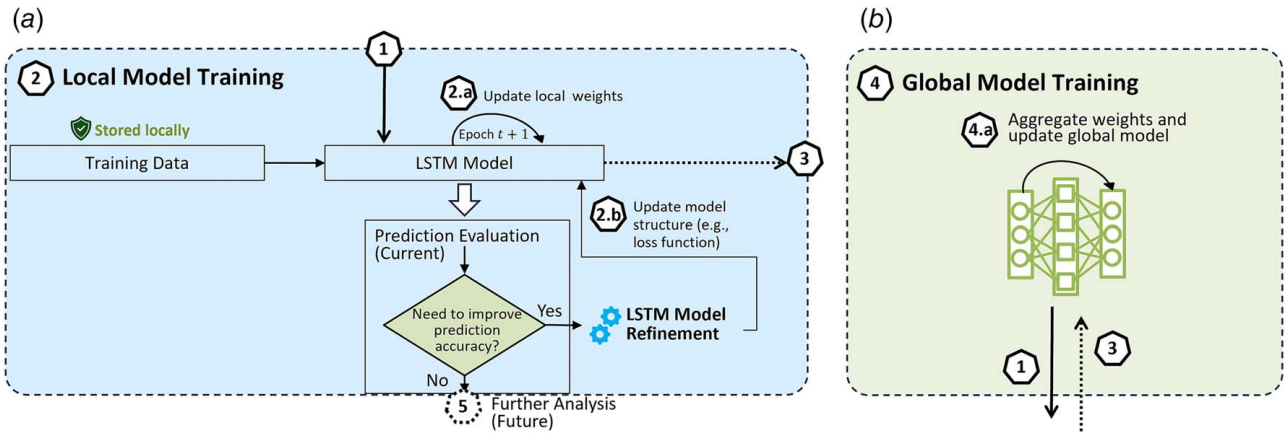

Fig. 5 (*a*) Flowchart of the local model training and (*b*) flowchart of the global model training

model from predicting constant emission values, ensuring that predictions can reflect the underlying data variability. The value of θ is chosen based on the variation of local data predictions. If the predicted data distribution has a smaller variation, a smaller θ is needed, and more tolerance can be given if data are more heterogeneous. γ serves as the regularization factor to specifically quantify the predicted E_{avg} values, where a larger γ value assigns more penalty on uniform predictions, discouraging overly consistent predictions across all data points. This custom loss function aims to ensure that the model not only reduces the overall prediction error but also captures the critical emission characteristics and maintains a sufficient level of prediction variability, which is essential for detecting anomalies in the AM process.

3.3 Federated Learning. FL enables collaborative model training across multiple clients, with a total of K clients engaging in T rounds of communications to train a global model. The custom loss function is incorporated into the FL approach the same way as standard loss functions, with the objective to minimize the loss functions (1) and (4) as $\min_{w\in\mathbb{R}^d} L(w)$, where $L(w)=\frac{1}{n}\sum_{k=1}^{K} L_k(w)$. In the FL paradigm, client k trains a local model separately on dataset $\mathcal{D}_k=\{\boldsymbol{x}_k,\boldsymbol{y}_k\}$, consisting of n_k data points. Typically, $\min_{w\in\mathbb{R}^d} L_k(w)$ is formulated, where $L_k(w)=\frac{1}{n_k}\sum_{i=1}^{n_k} l_k(\boldsymbol{x}_{k,i},\boldsymbol{y}_{k,i};w)$. The loss $l_k(\boldsymbol{x}_{k,i},\boldsymbol{y}_{k,i};w)$ is calculated based on the discrepancy between the prediction on actual data point $\boldsymbol{y}_{k,i}$ and the predicted value with input $\boldsymbol{x}_{k,i}$ and model parameters w. The client loss functions are then aggregated into the global loss function.

The aggregation algorithm used in the global model training step is an iterative method that merges each client model's $g_k=\nabla L_k(w^k)$ and the gradients of the loss over $\mathcal{D}_k$ for every round. Specifically, the well-known FedAvg algorithm is one such approach, as it selects C-fraction of clients from total K clients to train E epochs in local updates for each communication round, using B local mini-batch size [19]. The local update $w_{t+1}^k \leftarrow w_t^k-\eta\nabla L_k(w_t^k)$ at training epoch t is iterated with a fixed learning rate η in each epoch before the client updates weights to the global server.

While FedAvg has demonstrated empirical success in collaborative training, it does not fully address heterogeneity. Statistically, it has been shown that this vanilla algorithm results in divergence in non-IID data across devices. To address this challenge, we also incorporate FedProx [37] and FedAvgM [38] into the framework. The pseudo codes for the three aggregation algorithms are merged in Algorithm 1. FedProx adds a proximal term $\frac{\mu}{2}\|w-w^t\|^2$ in the update of local weights in addition to the calculated gradients for a more general case of FedAvg but benefits the local objective. The usage of the proximal term is a well-acknowledged method in the optimization settings and FedProx has proved convergence guarantees. FedAvgM incorporates momentum to accelerate training by the accumulation of the gradient history to dampen oscillations, changing the original gradient update into $w\leftarrow w-v, v\leftarrow\beta v+\Delta w$. This modification is beneficial for FL settings with non-IID data. In addition to the classical aggregation formula in the algorithm, an α_k factor is added to the term w_{t+1}^k, representing the weight assigned to each client k based on various criteria such as data quality, correlation of client data distributions, and historical performance. By adjusting this coefficient, clients with more reliable or relevant data will have a greater influence on the aggregated model, thereby improving the overall effectiveness and robustness of the federated process.

Algorithm 1 FedAvg, FedProx, and FedAvgM

Begin
Server side:
  initialize w_0
  for each round $t=1$ to T **do**
    select a C fraction of clients m
    for each client k in m **in parallel do**
      $w_{t+1}^k \leftarrow$ ClientUpdate(k, w_t)
    update global weights
    $w_{t+1}\leftarrow\sum_{k=1}^{K}\alpha_k\frac{n_k}{n}w_{t+1}^k$
ClientUpdate(k, w_t):
  initialize w_t^k
  for each epoch $e=1$ to E **do**
    split $\mathcal{D}_k$ into $\mathcal{B}$ with batches of size B
    for batch $b\in\mathcal{B}$ **do**
      $w\leftarrow w-\eta\nabla l(w;b)$ (FedAvg)
      $w\leftarrow w-\eta\nabla l(w;b)+\frac{\mu}{2}\|w-w^t\|^2$ (FedProx)
      $w\leftarrow w-v, v\leftarrow\beta v+\eta\nabla l(w_t;b)$ (FedAvgM)

Aggregation algorithms play a crucial role in the model convergence speed, particularly when training with heterogeneous datasets. Our interest lies in the efficacious training of an FL model for the AM scenario, given the diverse custom design configuration. Enhancing prediction accuracy through refined local model training is another key concern. We propose a systematic approach for training the local model as shown in Fig. 5(*a*). This serves as a comprehensive illustration of the components in Fig. 1. For each communication round, the local model receives updated weights via step 1, followed by local training (step 2.a). After the training, additional assessments are incorporated into the performance evaluation procedure to dynamically fine-tune the model. The performance evaluation involves a sequential assessment of various performance metrics, an example of which is shown as the diamond block for accuracy assessment in the flowchart. This metric is selected and exclusively considered here because it is widely recognized as a fundamental indicator of the system's performance. As shown in the diamond block of Fig. 5(*a*), the assessment process begins with an evaluation of prediction accuracy to determine if improvements are necessary. If improvements are needed, the model will be refined through step 2.b. Subsequently, if acceleration in convergence is necessary, a request will be sent

Table 1 Comparison of the average performance of LT, CT, FL, and CL-employed models (LT-CL, CT-CL, and FL-CL)

	RMSE		MAE		MAPE	
Model	Train	Test	Train	Test	Train	Test
LT	8.5332 (1.0821)	17.1221 (2.0932)	7.0956 (0.0711)	12.9507 (0.1439)	0.0179 (0.0321)	0.0286 (0.0611)
CT	6.4553 (0.3841)	6.9628 (0.2102)	3.5692 (0.1342)	4.2039 (0.2927)	0.0084 (0.0010)	0.0092 (0.0012)
FL	8.0268 (3.6791)	12.1878 (4.2129)	6.3872 (2.6563)	6.8471 (3.1680)	0.0091 (0.0065)	0.0122 (0.0041)
LT-CL	7.5972 (1.0112)	**11.4562 (1.8730)**	4.5692 (0.0807)	**8.5623 (0.1269)**	0.0078 (0.0012)	**0.0179 (0.0025)**
CT-CL	6.1068 (0.3177)	**6.4781 (0.1973)**	3.2521 (0.1259)	**3.8647 (0.2987)**	0.0078 (0.0011)	**0.0075 (0.0010)**
FL-CL	7.9073 (2.8834)	**10.2421 (3.7281)**	6.2387 (2.3040)	**6.3892 (3.3462)**	0.0073 (0.0012)	**0.0075 (0.0011)**

Note: The numbers in each cell represent the average values from five replications, with standard deviations in parentheses. Models that consistently outperform their standard loss function counterparts across all metrics, as shown by the bold test metrics.

Table 2 Performance of trials across clients after 1600 training rounds, averaged over 5 replications with standard deviations in parentheses

Experiment		Client 1	2	3	4	5	6	7	8
Trial_1	RMSE	**7.1313 (0.0503)**	15.3064 (0.6130)	16.3799 (0.6153)	11.6111 (0.5872)	8.8964 (0.3207)	10.4832 (1.2159)	16.8851 (0.8179)	16.6970 (0.6708)
	MAE	**4.4262 (0.0281)**	10.4606 (0.3732)	11.3517 (0.3490)	7.1766 (0.2055)	5.6666 (0.2011)	6.3164 (0.6199)	11.1171 (0.3228)	10.7386 (0.3798)
	MAPE	**0.0078 (0.0000)**	0.0168 (0.0011)	0.0180 (0.0011)	0.0121 (0.0004)	0.0096 (0.0002)	0.0108 (0.0011)	0.0167 (0.0007)	0.0176 (0.0011)
Trial_6	RMSE	16.2367 (2.0768)	19.1696 (1.5673)	20.2821 (1.5371)	11.4949 (1.3196)	9.8758 (0.8303)	**8.1225 (0.8408)**	22.8691 (1.8892)	19.1698 (1.7247)
	MAE	12.0486 (1.6683)	14.1590 (0.8815)	14.6228 (0.7867)	8.2019 (0.7335)	6.8118 (0.7118)	**5.3371 (0.8505)**	15.6537 (1.4015)	14.5123 (1.1474)
	MAPE	0.0184 (0.0026)	0.0212 (0.0022)	0.0214 (0.0019)	0.0131 (0.0011)	0.0112 (0.0012)	**0.0090 (0.0011)**	0.0223 (0.0024)	0.0216 (0.0025)
$\text{Trial}_{1,6}$	RMSE	**8.2146 (0.7176)**	10.6883 (1.7725)	12.2639 (1.8201)	9.1959 (1.4345)	8.4586 (1.1182)	**8.6225 (1.2804)**	15.5501 (1.9471)	10.4505 (1.8720)
	MAE	**5.3362 (0.5975)**	7.3724 (1.4186)	8.5507 (1.5096)	6.2017 (1.2784)	5.6163 (1.0977)	**5.7934 (1.2661)**	10.5181 (1.5394)	7.0993 (1.4404)
	MAPE	**0.0089 (0.0007)**	0.0119 (0.0020)	0.0134 (0.0020)	0.0103 (0.0016)	0.0093 (0.0014)	**0.0098 (0.0017)**	0.0156 (0.0018)	0.0117 (0.0019)
$\text{Trial}_{1,3}$	RMSE	**9.6500 (0.8085)**	8.7615 (1.7428)	**9.6687 (1.8903)**	11.1466 (0.9567)	12.1375 (2.1217)	10.4715 (1.5731)	17.1491 (1.0332)	9.2248 (1.5259)
	MAE	**6.2235 (0.4925)**	5.9816 (1.2212)	**6.6344 (1.3672)**	7.3410 (0.5583)	8.1920 (1.5924)	7.8192 (1.5029)	11.1094 (0.6458)	6.1976 (1.1006)
	MAPE	**0.0104 (0.0008)**	0.0100 (0.0016)	**0.0107 (0.0018)**	0.0120 (0.0008)	0.0131 (0.0022)	0.0125 (0.0021)	0.0169 (0.0009)	0.0105 (0.0016)

Note: Bold values highlight the top clients during local training for each scenario, as well as the best test values across all scenarios.

from step 4.b to the global model server in Fig. 5(*b*) for the refinement of the FL framework. In this work, our current emphasis is on preliminary experiments, we have simplified the refinement process. For step 2.b, the model's loss function is adjusted during local model training. Step 5 represents opportunities for further improvement of the ML framework considering additional assessments on other performance metrics.

4 Case Studies

4.1 Experiment Settings. Three case studies are discussed to assess the efficacy of our FL framework and understand the underlying mechanisms. The experiment settings, from the perspective of local and global model training, are crafted with the consideration of the above objectives. Grid search was conducted on the three hyperparameters in the custom loss function: λ (0.1, 0.5, 1, 5), γ (1, 2, 5, 10), and θ (0.001, 0.01, 0.02, 0.05) to determine the appropriate hyperparameter combination. This tuning was performed during local client model training, and the local validation RMSE results were averaged across clients to ensure robustness. Among several candidate combinations where the performance was closely matched, we selected the hyperparameters by also considering values that were not overly biased within the range. Specifically, the λ value is set at 1.0 for the MSE of the E_{avg} value to ensure its impact on the overall loss function. The default value of 1/6 is derived from evenly distributing importance across the six features in the standard loss function. We use $\lambda = 1.0$ instead of the default value 1/6 to ensure that the model gives adequate attention to this feature. γ is set to 5.0 as a regularization factor to prevent potential over-smoothing in predictions. Preliminary experiments indicated that without this penalty, the model sometimes generated predictions for E_{avg} that were excessively constant, failing to capture nuanced variations in the data. Such constant predictions could lead to oversights in identifying critical anomalies in overheating detection. θ is determined as 0.01 after comparing its candidate values from the grid search results with the standard deviation of the dataset. This is because θ serves as a regularized threshold to control the variance of the prediction values for the output. Therefore, the selection of θ needs to consider not only the grid search results but also its practical applicability. Consequentially, we deter the model from producing such oversimplified outputs and encourage it to capture the inherent intricacies in the data.

Model training hyperparameters are selected based on established standards and insights from prior experiments. We choose timestamp $\tau = 10$, $B = 32$, and learning rate $\eta = 0.01$. Stochastic gradient descent (SGD) is used across all the experiments as the optimizer in the LSTM model to be consistent with the original FedAvg algorithm, as it has been tested to perform well in millions of mobile device collaborative training [19]. SGD offers simplicity with potentially better generalization in certain scenarios due to its noisy updates, which can be more stable for the FL architecture. For α_k in the aggregation process, we assign value 1 across all clients, treating each client with equal importance. This simplifies the implementation and isolates the analysis of other factors on model performance, allowing for a more focused evaluation.

In our regression tasks, we use standard performance metrics: the root mean square error (RMSE), the mean absolute error (MAE), and the mean absolute percentage error (MAPE). RMSE is computed as the standard deviation of the residuals, reflecting the model's accuracy in predicting section-wise average emissions. MAE measures the average magnitude of the errors in a set of predictions, and MAPE expresses the prediction residuals as a percentage, facilitating a scale-independent assessment of the model's accuracy. Both MAE and MAPE are particularly useful for comparing the model's performance across different scales of data. Lower values of RMSE, MAE, and MAPE indicate a better-performing model. From the FL global training side, we assume that under the AM scenario, there are no stragglers during each communication round, and all the selected C fraction of the total K clients

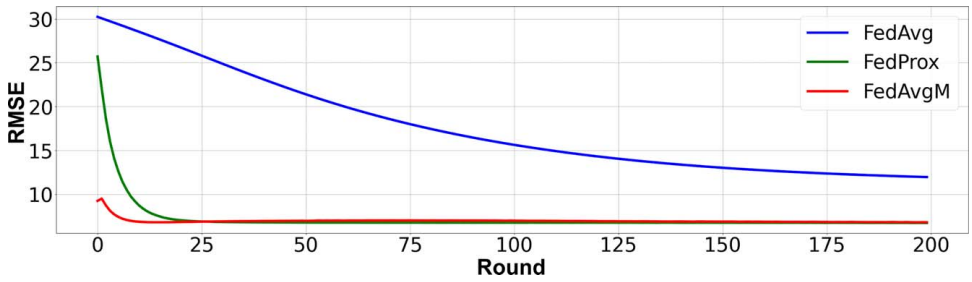

Fig. 6 Test RMSE over rounds for FedAvg, FedProx, and FedAvgM methods

will join the training. In the following case studies, the first seven clients are used for training and the eighth client is served as the test dataset, if not illustrated specifically. Each experiment is replicated five times to ensure the robustness of the results.

4.2 Assessment of Local, Central, and Federated Learning Training With Custom Loss Functions. The first study focuses on comparing the efficacy of localized training (LT), centralized training (CT), and FL under two main criteria: the performance of custom versus standard loss functions and the overall effectiveness of each training method. LT trains individualized models on client-specific data, aiming for high accuracy on local data distributions. We have trained seven separate models through the LT approach. CT aggregates all client data to train a single model, while FL distributes the training process across clients without pooling their data. Both loss functions are tested for their effects on model convergence and performance metrics. All the models are given enough training rounds until they converge. For this scenario, $C = 1$ is used for FL training, which implies that all seven clients participate in each training round.

The experiments were conducted across five replications, with the results summarized in Table 1, including both average performance metrics and their respective standard deviations. Notably, the FL model exhibits higher variability during the training phase, with a standard deviation of RMSE at 3.6791. This is significantly higher than that of the LT model at 1.0821 and the CT model at 0.3841. The lower variability in CT suggests a more stable model performance. In test results, however, the FL model shows improved consistency as its standard deviation of RMSE aligns more closely with that of the training data. In contrast, the LT model presents higher variance and poorer average performance in test scenarios, indicating potential overfitting issues. The CT model maintains consistent performance levels between the training and testing phases, as shown by its comparably low standard deviations, demonstrating reliable model behavior across different datasets. It can be seen from Table 1 that models employing the custom loss (CL) function (named LT-CL, CT-CL, and FL-CL models, respectively) consistently outperform their standard loss function counterparts (LT, CT, and FL models, respectively) across all metrics. When the LT model is enhanced with the custom loss function (LT-CL), it shows a significant reduction in RMSE from 17.1221 to 11.4562 in the test set. Similarly, the central training model (CT-CL) and federated learning model (FL-CL) with the custom loss function also exhibit improved accuracy with lower RMSE, MAE, and MAPE values compared to their standard loss function equivalents. This pattern validates the effectiveness of the custom loss function in improving model performance, likely due to its ability to capture more nuanced features and penalize critical prediction errors more effectively.

When comparing the CT and FL, as well as CT-CL and FL-CL models, we notice that FL demonstrates performance discrepancy to CT while outperforming LT. This is evident from the close RMSE values in the test set (6.9628 for CT, 12.1878 for FL, and 17.1221 for LT) and similarly close MAE and MAPE values. The introduction of the custom loss function into FL-CL further narrows this performance gap, with FL-CL achieving a lower RMSE of 10.2421 in the test set, compared to CT-CL which has an RMSE of 6.4781. These results suggest that FL, particularly when combined with a custom loss function, is an effective approach for training models that are nearly as accurate as those trained centrally, with the added benefits of privacy preservation and decentralized data handling intrinsic to FL.

4.3 Federated Learning Performance With Limited Clients. SMEs can employ similar printers and geometries while varying in process parameters for printing objects. Training models collaboratively can use data of different process parameters, but limited client participation will lead to data scarcity, especially in dimensions such as laser power and speed. Understanding how the FL training process is performed under conditions of limited client participation helps SMEs make the most of available data to improve product quality. By conducting FL with data from 1 or 2 clients out of seven, we evaluate how such selective participation influences model performance, exploring the bounds of client participation in FL.

This case study uses the custom loss function since it has been empirically demonstrated its performance better than the standard loss function in the previous case study. The choice of participating clients also allows for an analysis of the influence of data heterogeneity on the model. With only one client, the model may become highly specialized in that client's data distribution. With two clients, there is an opportunity to observe how the introduction of additional data diversity impacts the model's performance and generalization. A series of experiments are conducted, each designed to elucidate the performance metrics under varying client data conditions. For each trial, we employed a consistent set of clients, randomly selected at the outset, and used throughout the entire training period. This method differs from the approach of sampling a subset of clients for each communication round. By maintaining a fixed group of clients, we aim to gain clearer insights into the training dynamics that emerge from a limited client base. Trial_1 utilizes data solely from client 1 to establish a foundational understanding of the process under controlled conditions. Trial_6 explores the performance of the model trained by client 6. To delve into the compound effects that arise from combining client datasets, $\text{Trial}_{1,3}$ aggregates models from clients 1 and 3 with the FL approach, while $\text{Trial}_{1,6}$ examines the outcomes from clients 1 and 6. This deliberate structuring of experiments serves to dissect the individual contributions of each client as well as to uncover the dynamics that emerge from their collaboration, thereby enriching our comprehension of the LPBF process in a multiclient AM scenario. Each trial is trained with $T = 1600$ rounds.

Table 2 summarizes the average performance results from each trial across five replications. Columns highlighted in bold indicate the training results of the respective trials, while the remaining clients are test results since they are not used for training in these trials. Using a single client is equivalent to LT, as mentioned in the first study. Trial_1 and Trial_6 encounter overfitting to their own client training models, while they perform poorly to other clients. This is also reflected by the standard deviation values. For instance, both trials exhibit low variance in their respective training outputs but show significantly higher variance and unstable performance

Table 3 Metrics comparison of FL algorithms for training clients, averaged over five replications with standard deviations in parentheses

Experiment		Client 1	2	3	4	5	6	7	8
FedAvg	RMSE	8.4608 (0.2728)	10.7780 (0.8125)	12.2334 (0.5495)	10.3742 (0.5246)	**8.4353 (0.3568)**	10.9410 (1.5816)	16.2122 (0.4036)	12.6341 (2.1858)
	MAE	5.3138 (0.1167)	7.5420 (0.4929)	8.5565 (0.3937)	6.4919 (0.3029)	**5.5770 (0.2707)**	7.7382 (0.9182)	10.6570 (0.2222)	8.3783 (1.3185)
	MAPE	0.0099 (0.0001)	0.0121 (0.0008)	0.0131 (0.0006)	0.0113 (0.0005)	**0.0099 (0.0003)**	0.0125 (0.0015)	0.0161 (0.0005)	0.0141 (0.0027)
FedProx	RMSE	**6.7320 (0.0695)**	6.7461 (0.0532)	7.4784 (0.0980)	6.8199 (0.0335)	6.8153 (0.0444)	7.0446 (0.2587)	8.7739 (0.1795)	**6.5573 (0.0462)**
	MAE	**4.1438 (0.0849)**	4.1873 (0.0642)	4.9751 (0.0820)	4.3625 (0.0440)	4.3503 (0.0640)	4.7434 (0.3655)	5.9633 (0.0976)	**3.9515 (0.0765)**
	MAPE	**0.0074 (0.0001)**	0.0075 (0.0001)	0.0084 (0.0001)	0.0077 (0.0001)	0.0077 (0.0001)	0.0082 (0.0004)	0.0097 (0.0001)	**0.0072 (0.0001)**
FedAvgM	RMSE	6.6870 (0.1181)	6.6983 (0.0599)	7.3162 (0.0835)	**6.7754 (0.0669)**	6.8371 (0.1995)	7.2599 (0.9625)	8.4438 (0.1701)	6.5618 (0.1251)
	MAE	4.0756 (0.1398)	4.1154 (0.0444)	4.8177 (0.1673)	**4.2989 (0.0801)**	4.3709 (0.2636)	4.9434 (1.1167)	5.7543 (0.1534)	3.9126 (0.1683)
	MAPE	0.0074 (0.0002)	0.0074 (0.0001)	0.0082 (0.0002)	**0.0076 (0.0001)**	0.0077 (0.0003)	0.0084 (0.0014)	0.0095 (0.0002)	0.0071 (0.0002)

Note: Bold values highlight the top clients during local training for each scenario, as well as the best test values across all scenarios.

in testing across different clients. This outcome emphasizes the issue of data heterogeneity from different clients and how single-client training struggles to produce a model that generalizes well. $\text{Trial}_{1,6}$ and $\text{Trial}_{1,3}$ present a mitigation of the overfitting issue, with notable improvements in performance metrics compared to single-client trials. Although the overall performance is still not comparable to employing all clients in the FL training, these trials illustrate the benefits of incorporating diverse datasets into the training process. This is also supported by the stable variance observed in test clients compared to the clients used for training. For instance, $\text{Trial}_{1,6}$ shows improved performance metrics on test clients relative to Trial_1, while maintaining similar standard deviations between the training and test datasets. Such integration of diverse datasets can enhance the performance of the FL global model.

Experiments involving training with a single client exhibit more randomness, as indicated by the disparate KLD values of clients 1 and 6, yet both yield poor test results. However, in testing scenarios, client 5 consistently outperforms other clients under both configurations. Furthermore, when training involves two clients, client 5 continues to achieve the best results with clients 1 and 6, whereas client 2 surpasses all others when trained with clients 1 and 3. Improved performance is observed on client 8 when training transitions from a client with lower KLD values (client 1) to a more diverse client (client 6), and in the two-client training scenarios between $\text{Trial}_{1,3}$ and $\text{Trial}_{1,6}$. This suggests that datasets with less heterogeneity tend to yield better performance.

4.4 Comparison of Aggregation Algorithms. Traditional FL research primarily focuses on FedAvg due to its simplicity and effectiveness. However, this algorithm struggles with non-IID data and client drift. It takes a longer time to converge when training with a more heterogeneous dataset. With limited clients involved in the training, the converge speed gets even slower. Other algorithms like FedProx and FedAvgM have been designed to handle better for heterogeneous datasets and converge faster. This case study is to compare the efficacy of these FL algorithms when combined with the designed custom loss function. The custom loss function has been demonstrated in the previous case study to perform better than a standard loss function. In this case study, we use the first seven clients as training data and client 8 as test data. Each algorithm is trained for $T = 600$ rounds, with performance metrics recorded every round.

In the FedProx algorithm, the selection of μ is guided by a heuristic that adapts to the observed loss trends. For this study, we have determined μ to be 0.02. This value is selected to prevent local models from leveraging their unique data insights while avoiding significant deviations from the global model. Similarly, for the FedAvgM algorithm, β is set to 0.9. This value is commonly adopted in the field due to its effectiveness in stabilizing and accelerating convergence. The integrated knowledge from past updates enables momentum to reduce the risk of getting trapped in local optima, thereby improving model convergence performance.

Figure 6 shows the test RMSE over rounds for each algorithm. FedAvg shows a stable but much slower convergence, indicating its difficulty in handling client heterogeneity. FedProx demonstrates rapid convergence and uses fewer rounds to converge compared to FedAvg. FedAvgM outperforms the other two algorithms with an initial good performance and converges with around 10 rounds.

Table 3 presents a comparison of the three FL algorithms in the RMSE, MAE, and MAPE metrics. The results for both the training and test clients are presented, with each experiment undergoing five replications, each consisting of several training rounds until model convergence is reached. The training scores are listed for all seven local clients, with their standard deviation included. The test scores are also averaged from the data of client 8 across these replications. FedAvg demonstrates significant variability across replications between the training and test clients, whereas FedProx and FedAvgM show stable performance. Both models surpass FedAvg in terms of average performance across both training and test clients. Notably, FedProx matches FedAvgM in performance

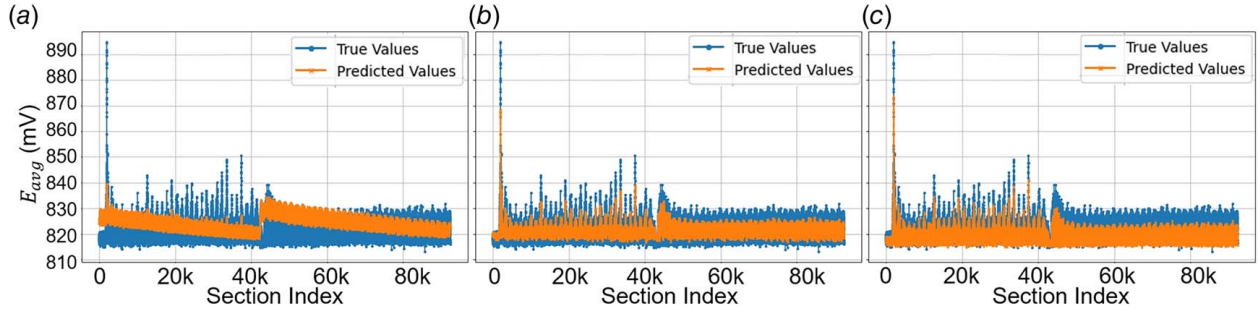

Fig. 7 Test predictions across (*a*) FedAvg, (*b*) FedProx, and (c) FedAvgM on average emission

and exhibits the lowest variance among different clients. Additionally, FedProx achieves superior results on test client 8, even though FedAvgM displays stronger training performance in five out of seven local clients.

The results in Table 3 also illustrate the impact of data heterogeneity on the three aggregation algorithms, shedding light on how each algorithm performs when faced with heterogeneous data. FedAvg shows higher variability in performance metrics across clients, with significant standard deviations noted especially for RMSE and MAPE in clients 6 and 7. This suggests FedAvg struggles with heterogeneity. FedProx exhibits generally better performance across all clients compared to FedAvg. Its standard deviation values are also smaller, indicating more stable performance across diverse client data. FedAvgM also shows improved performance, demonstrating some resilience to heterogeneity. From the KLD value's perspective, higher KLD values (such as 0.81 and 1.38 for clients 6 and 7) correlate with greater variability in performance metrics, suggesting that more significant differences in data distribution may lead to poorer and less stable performance. Lower KLD values (such as 0.04 for client 1) are associated with more consistent and better performance metrics, indicating a more similar distribution to the test client and hence better generalization.

While Table 3 demonstrates the numeric model performance at the end of training, Fig. 7 shows the detailed comparison of actual values and predicted values over E_{avg} on the test client's dataset, providing a more nuanced understanding of each algorithm's performance. The figure visualizes the predicted values in yellow color compared to actual values in blue color, across the three algorithms. FedAvg's predictions are not closely aligned with the true values, suggesting a disparity in capturing the data's underlying patterns. The predictions from FedProx are more consistent with the true values, as shown in the visualization, demonstrating its capability to handle data heterogeneity effectively. While FedAvgM's predictions are an improvement over FedAvg, the visuals show that it tends to predict lower values than the actual ones, unlike FedProx, which delivers a uniform output that closely matches the sequence of the true values.

4.5 Discussion. The assessment of model training with LT, CT, and FL approaches in Sec. 4.2 reveals that while local training is highly specialized, it may not generalize well. Central training offers robust generalization but at the cost of privacy and potential bias toward dominant data sources. FL emerges as a potent alternative, balancing specialization with generalization, especially when paired with a custom loss function. The introduction of a custom loss function significantly enhances the predictive performance of both local and central training paradigms. Furthermore, federated learning, especially with the custom loss function, achieves a level of accuracy that is competitive with central learning models, affirming its viability as a decentralized learning strategy.

The analysis of the FL performance with limited clients in Sec. 4.3 emphasizes the challenges of data heterogeneity in FL and the potential of multiclient data integration to enhance model robustness. It reinforces the notion that even limited data diversity can significantly impact the effectiveness of FL models, which is a valuable insight for situations in AM where data may be scarce or diverse client participation is constrained.

The comparison of FL algorithms in Sec. 4.4 shows the significance of choosing an appropriate FL algorithm when dealing with complex loss functions and heterogeneous data. The refinement of aggregation algorithms significantly affects the computational time. FedProx and FedAvgM show promise in such settings, while FedAvg needs longer rounds to converge.

The experiments' exploration into these FL algorithms reveals their potential in enhancing AM processes. However, no single FL algorithm consistently excels in every scenario, indicating that algorithm selection and customization remain critical challenges. For instance, customized models may consider additional performance metrics such as convergence time. The diamond block in Fig. 5(*a*) represents the current evaluation phase for local models, focusing specifically on assessing accuracy improvement. If accuracy enhancement is needed, a model refinement process is initiated to update the model's architecture. Otherwise, the analysis may proceed to examine additional performance metrics (step 5). Figure 8 is provided to offer a detailed technical route for step 5, showing the further analyses that can be carried out when additional performance metrics are considered, potentially leading to a robust theoretical framework that offers clear guidelines for the optimal adjustment of the FL framework. This approach involves additional refinements for both local and global model training, such as the aggregation of local models' hyperparameters, examination of model convergence speed, and the selection of aggregation algorithms for the global model. Refinements to the FL framework are implemented as part of step 4.b in the global model training process. The insights from a more robust methodology could guide practitioners in selecting and customizing FL algorithms for specific applications, particularly those where data privacy, non-IID data, and complex problem constraints are prevalent.

In addition, while FL facilitates training across heterogeneous data, it exhibits suboptimal performance in highly heterogeneous environments in terms of predictive accuracy and convergence rate. Therefore, there is still room for further improvement. For instance, if a new client with a different set of process configurations and an uneven quantity of datasets is added to the FL framework, it could detrimentally affect the model's overall performance. In the proposed method, our current framework has explored variants of averaging-based FL aggregation algorithms, incorporating strategies such as

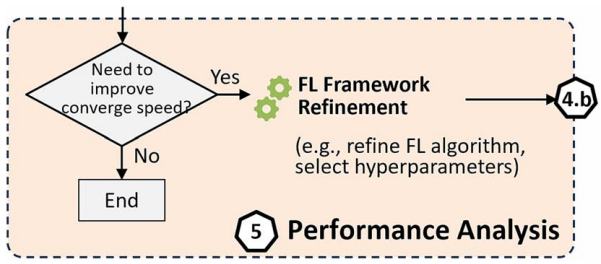

Fig. 8 Flowchart of the performance analysis for FL framework

regularization and momentum to address data heterogeneity. These modifications have demonstrated performance enhancements over the "vanilla" FedAvg algorithm. Future research could continue in this direction by exploring heterogeneous datasets from SMEs and their impact on adopting FL in the AM industry.

5 Conclusion

To address the need for SMEs in AM to train heterogeneous datasets collaboratively without compromising data privacy and proprietary information, this article has revealed efficient FL training in the context of AM through a detailed exploration of case studies with thermal emission data. This article demonstrates that custom loss functions can significantly improve the performance of FL models. Through experimentation, it was shown that these functions yield better accuracy and generalization across different client datasets when compared to standard loss functions. The challenges posed by data heterogeneity are identified in FL. By training models with data from varying numbers of clients, we revealed that a model's ability to generalize effectively is compromised when trained on data from a single client but improves with the introduction of more diverse client data.

Future research will focus on the enhancement of the predictive capabilities of FL models within even more diverse manufacturing environments and heterogeneous datasets. The scope of investigation will expand to include FL models trained on datasets characterized by a broader array of part geometries, printing configurations, and material types, among other variables. This will provide insights into the robustness and adaptability of FL in the face of increased data heterogeneity, a step closer to its widespread application in the dynamic field of AM. Further exploration into these factors will also aid in incorporating physics-related information into model refinement that can handle the complexity and variability inherent in manufacturing data, ultimately enhancing the predictive power and efficiency of FL models in real-world industrial settings.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Gardan, J., 2017, "Additive Manufacturing Technologies: State of the Art and Trends," Int. J. Prod. Res., **54**(10), pp. 149–168.
[2] Huang, Y., Leu, M. C., Mazumder, J., and Donmez, A., 2015, "Additive Manufacturing: Current State, Future Potential, Gaps and Needs, and Recommendations," ASME J. Manuf. Sci. Eng., **137**(1), p. 014001.
[3] Tapia, G., and Elwany, A., 2014, "A Review on Process Monitoring and Control in Metal-Based Additive Manufacturing," ASME J. Manuf. Sci. Eng., **136**(6), p. 060801.
[4] Chen, Y., Peng, X., Kong, L., Dong, G., Remani, A., and Leach, R., 2021, "Defect Inspection Technologies for Additive Manufacturing," Int. J. Extreme Manuf., **3**(2), p. 022002.
[5] Scime, L., and Beuth, J., 2018, "Anomaly Detection and Classification in a Laser Powder Bed Additive Manufacturing Process Using a Trained Computer Vision Algorithm," Addit. Manuf., **19**, pp. 114–126.
[6] Wang, C., Tan, X., Tor, S. B., and Lim, C., 2020, "Machine Learning in Additive Manufacturing: State-of-the-Art and Perspectives," Addit. Manuf., **36**(B), p. 101538.
[7] Martinsuo, M., and Luomaranta, T., 2018, "Adopting Additive Manufacturing in SMEs: Exploring the Challenges and Solutions," J. Manuf. Technol. Manag., **29**(6), pp. 937–957.
[8] Duray, R., 2002, "Mass Customization Origins: Mass or Custom Manufacturing?," Int. J. Oper. Prod. Manag., **22**(3), pp. 314–328.
[9] Pan, S. J., and Yang, Q., 2009, "A Survey on Transfer Learning," IEEE Trans. Knowl. Data Eng., **22**(10), pp. 1345–1359.
[10] Gou, J., Yu, B., Maybank, S. J., and Tao, D., 2021, "Knowledge Distillation: A Survey," Int. J. Comput. Vis., **129**(6), pp. 1789–1819.
[11] Choi, S., Kim, H., and Cho, H., 2024, "Defect Detection in the Manufacturing Domain Using Product Design Data and Self-Knowledge Distillation," 2024 International Conference on Electronics, Information, and Communication (ICEIC), Taipei, Taiwan, Jan. 28–31, IEEE, pp. 1–4.
[12] Zhang, Y., and Yang, Q., 2021, "A Survey on Multi-Task Learning," IEEE Trans. Knowl. Data Eng., **34**(12), pp. 5586–5609.
[13] Gu, D. D., Meiners, W., Wissenbach, K., and Poprawe, R., 2012, "Laser Additive Manufacturing of Metallic Components: Materials, Processes and Mechanisms," Int. Mater. Rev., **57**(3), pp. 133–164.
[14] Liu, C., Tian, W., and Kan, C., 2022, "When AI Meets Additive Manufacturing: Challenges and Emerging Opportunities for Human-Centered Products Development," J. Manuf. Syst., **64**(7), pp. 648–656.
[15] Buckholtz, B., Ragai, I., and Wang, L., 2015, "Cloud Manufacturing: Current Trends and Future Implementations," ASME J. Manuf. Sci. Eng., **137**(4), p. 040902.
[16] Kanagavelu, R., Li, Z., Samsudin, J., Hussain, S., Yang, F., Yang, Y., Goh, R. S., and Cheah, M., 2021, "Federated Learning for Advanced Manufacturing Based on Industrial IoT Data Analytics," *Implementing Industry 4.0: The Model Factory as the Key Enabler for the Future of Manufacturing*, Vol. 202, Springer, Cham, pp. 143–176.
[17] Byrd, D., and Polychroniadou, A., 2020, "Differentially Private Secure Multi-Party Computation for Federated Learning in Financial Applications," Proceedings of the First ACM International Conference on AI in Finance, New York, NY, Oct. 15–16, pp. 1–9.
[18] Li, L., Fan, Y., Tse, M., and Lin, K.-Y., 2020, "A Review of Applications in Federated Learning," Comput. Ind. Eng., **149**, p. 106854.
[19] McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A., 2017, "Communication-Efficient Learning of Deep Networks From Decentralized Data," Artificial Intelligence and Statistics, Ft. Lauderdale, FL, Apr. 20–22, PMLR, pp. 1273–1282.
[20] da Silveira Dib, M. A., Ribeiro, B., and Prates, P., 2021, "Federated Learning as a Privacy-Providing Machine Learning for Defect Predictions in Smart Manufacturing," Smart Sustain. Manuf. Syst., **5**(1), pp. 1–17.
[21] Mehta, M., and Shao, C., 2022, "Federated Learning-Based Semantic Segmentation for Pixel-Wise Defect Detection in Additive Manufacturing," J. Manuf. Syst., **64**(8), pp. 197–210.
[22] Truong, H. T., Ta, B. P., Le, Q. A., Nguyen, D. M., Le, C. T., Nguyen, H. X., Do, H. T., Nguyen, H. T., and Tran, K. P., 2022, "Light-Weight Federated Learning-Based Anomaly Detection for Time-Series Data in Industrial Control Systems," Comput. Ind., **140**, p. 103692.
[23] Hegiste, V., Legler, T., and Ruskowski, M., 2022 "Application of Federated Learning in Manufacturing," arXiv:2208.04664.
[24] Shi, N., and Kontar, R. A., 2023, "Personalized Federated Learning via Domain Adaptation With an Application to Distributed 3D Printing," Technometrics, **65**(3), pp. 328–339.
[25] Putra, M. A. P., Rachmawati, S. M., Abisado, M., and Sampedro, G. A., 2023, "HFTL: Hierarchical Federated Transfer Learning for Secure and Efficient Fault Classification in Additive Manufacturing," IEEE Access, **11**, pp. 54795–54807.
[26] Shi, Z., Li, Y., and Liu, C., 2024, "Knowledge Distillation-Based Information Sharing for Online Process Monitoring in Decentralized Manufacturing System," J. Intell. Manuf., pp. 1–16.
[27] Zhang, Z., Mou, S., Reisi Gahrooei, M., Pacella, M., and Shi, J., 2024, "Federated Multiple Tensor-on-Tensor Regression (FedMTOT) for Multimodal Data Under Data-Sharing Constraints," Technometrics, pp. 1–26.
[28] Aggour, K. S., Kumar, V. S., Cuddihy, P., Williams, J. W., Gupta, V., Dial, L., Hanlon, T., Gambone, J., and Vinciquerra, J., 2019, "Federated Multimodal Big Data Storage & Analytics Platform for Additive Manufacturing," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, Dec. 9–12, IEEE, pp. 1729–1738.
[29] Guo, S., Agarwal, M., Cooper, C., Tian, Q., Gao, R. X., Grace, W. G., and Guo, Y. B., 2022, "Machine Learning for Metal Additive Manufacturing: Towards a Physics-Informed Data-Driven Paradigm," J. Manuf. Syst., **62**(4), pp. 145–163.
[30] Li, J., Jin, R., and Hang, Z. Y., 2018, "Integration of Physically-Based and Data-Driven Approaches for Thermal Field Prediction in Additive Manufacturing," Mater. Des., **139**(4), pp. 473–485.
[31] Chen, M., and Guo, W., 2023, "DCGAN-CNN With Physical Constraints for Porosity Prediction in Laser Metal Deposition With Unbalanced Data," Manufacturing Letters, **35**(4), pp. 1146–1154.
[32] Hochreiter, S., and Schmidhuber, J., 1997, "Long Short-Term Memory," Neural Comput., **9**(8), pp. 1735–1780.
[33] Galkin, G., Gawade, V., Guo, W., Yi, J., and Guo, Y., 2022, "In-Situ and Real-Time 3D Pyrometry for Thermal History Diagnosis in Laser Fusion Process," Manuf. Lett., **33**, pp. 862–871.
[34] Lei, R., Guo, Y., and Guo, W., 2024, "Physics-Guided Long Short-Term Memory Networks for Emission Prediction in Laser Powder Bed Fusion," ASME J. Manuf. Sci. Eng., **146**(1), p. 011006.
[35] Gawade, V., Galkin, G., Guo, Y., and Guo, W. G., 2022, "Quantifying and Modeling Overheating Using 3D Pyrometry Map in Powder Bed Fusion," Manuf. Lett., **33**, pp. 880–892.
[36] He, K., Zhang, X., Ren, S., and Sun, J., 2015, "Delving Deep Into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification," Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, Dec. 7–13, pp. 1026–1034.
[37] Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V., 2020, "Federated Optimization in Heterogeneous Networks," Proceedings of Machine Learning and Systems, Austin, TX, Mar. 2–4.
[38] Hsu, T.-M. H., Qi, H., and Brown, M., 2019, "Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification," arXiv:1909.06335.