Interpretable Data Fusion for Distributed Learning: A Representative Approach via Gradient Matching

Mengchen Fan[a], Baocheng Geng[a], Keren Li[b], Xueqian Wang[c] and Pramod K. Varshney[d]
[a]*Department of Computer Science, University of Alabama at Birmingham, Birmingham, US*
[b]*Department of Mathematics, University of Alabama at Birmingham, Birmingham, US*
[c]*Department of Electronic Engineering, Tsinghua University, Beijing, China*
[d]*Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, US*

***Abstract*—This paper introduces a representative-based approach for distributed learning that transforms multiple raw data points into a virtual representation. Unlike traditional distributed learning methods such as Federated Learning, which do not offer human interpretability, our method makes complex machine learning processes accessible and comprehensible. It achieves this by condensing extensive datasets into digestible formats, thus fostering intuitive human-machine interactions. Additionally, this approach maintains privacy and communication efficiency, and it matches the training performance of models using raw data. Simulation results show that our approach is competitive with or outperforms traditional Federated Learning in accuracy and convergence, especially in scenarios with complex models and a higher number of clients. This framework marks a step forward in integrating human intuition with machine intelligence, which potentially enhances human-machine learning interfaces and collaborative efforts.**

***Index Terms*—Data fusion, distributed learning, interpretability, human integrated AI.**

I. Introduction

Fusion of information from diverse sources, such as sensors, provides a richer, more comprehensive perspective on a phenomenon of interest, thereby enhancing predictive accuracy and decision-making efficiency [1, 2, 3, 4, 5]. In numerous application scenarios, data are gathered and maintained locally by various agents or facilities. Recently, the imposition of stricter data privacy regulations and limitations on data communication bandwidth have driven the development of distributed learning systems capable of training a centralized model without the need to exchange raw data.

One notable approach, Federated Learning (FL) allows for the exchange of model parameter updates instead of raw data, thus preserving privacy, minimizing data transmission, and reducing the reliance on centralized data repositories. A variety of FL algorithms have been devised to tackle different machine learning (ML) tasks, including deep neural networks (DNNs) [6, 7, 8], gradient boosted decision trees (GBDTs) [9, 10], logistic regression [11], and support vector machines (SVMs) [12]. These architectures are tailored to meet specific requirements such as data distribution, communication limits, and privacy needs in distributed, decentralized, and hierarchical structures [13, 14, 15, 16, 17].

In the design and implementation of these ML systems, a major challenge is the effective integration of human insights into decision-making and oversight. Traditionally, the emphasis has been solely on the accuracy of ML models, neglecting the need for comprehensibility. This approach does not recognize the collaborative essence of such systems in critical environments, where decisions are made through the combined efforts of humans and ML models. In scenarios where small details have significant impacts on outcomes, exclusive reliance on automated decision-making systems is not advisable [18].

Several studies within the signal processing and ML literature have investigated collaborative human machine decision-making, focusing on how humans interpret and perceive data and ML outputs [18, 19, 20, 21]. Yet, these investigations frequently neglect a vital consideration: the ML models must be designed to be interpretable, ensuring that they can interact with humans at a reasonable pace and in an easily understandable format. Many ML models are black boxes that offer little explanation of their training and inference processes in terms of understandability to humans. This is particularly true in FL, where the model parameter updates from each client lack intuitive physical meaning and are incomprehensible to human operators. For example, while humans can easily interpret an image, understanding the model parameter updates in a Deep Neural Network (DNN) system is far from straightforward.

Inspired by the score-matching representative (SMR) proposed in [22], we present a distributed learning framework centered around the concept of representative. A

B. Geng and K. Li are the corresponding authors.

'representative data point', or simply '*representative*' is defined as a synthesized data point derived from each data block on decentralized devices. This representative is computed to have the same effect as the original raw dataset on the model's weight parameters. Specifically, SMR is designed to replicate the gradient of the loss function on the model parameters, ensuring that its impact on model updates mirrors that of the raw data.

There have been previous studies [23, 24, 25] utilizing gradient matching techniques that have demonstrated the ability to compress large datasets into smaller, manageable subsets. However, this process is accompanied by a certain degree of loss in model performance. Our method extends beyond mere dataset reduction to model training in distributed environments in real-time. Moreover, it ensures the training accuracy of using a representative is closer to that of the original data by introducing a residual parameter. Additionally, our approach enhances the interpretability of the representative, by pertubating a small noise based on the mean of the dataset, so that representatives are understandable to humans. Also, our approach favors a real-time decision-making method that enhances efficiency, responsiveness, and accuracy by enabling immediate actions based on current data in the distribution environment.

Our framework aims to ensure that employing a single representative data point achieves outcomes comparable to those derived from using the raw data set within the block. This approach offers several advantages:

- Reduced Communication Overhead: By condensing large datasets into a single representative, our method lowers the communication burden between the clients and the servers.
- Privacy Preservation: The representative is a virtually constructed data point that encapsulates aggregated data information, thus protecting individual-specific details and ensuring privacy.
- Enhanced Interpretability: Unlike FL, where only model parameter updates are shared, our approach maintains data's original structure and format through the representative, enhancing interpretability for human operators. This feature is crucial for systems requiring human-centric decision-making.
- Improved Performance: Numerical results show that our framework surpasses the FL approach, specifically FedAVG, in accuracy metrics. In addition, our method demonstrates smoother and more rapid convergence, highlighting its efficiency and effectiveness.

II. Problem Formulation

In distributed learning tasks, we define a dataset distribution $\mathscr{D}$ composed of n data points, represented as $x = (x_1, \ldots, x_n)^T$, where each $x_i \in \mathbb{R}^d$, and the corresponding system outputs (or labels) as $y = (y_1, \ldots, y_n)^T$ for $i = 1, \ldots, n$. The task of learning and inference in this context can be articulated as an optimization problem:

$$\min_{\mathrm{w}} L(\mathrm{w}; x, y) = l(\mathrm{w}; x, y) + r(\mathrm{w}; \lambda). \tag{1}$$

Here, L symbolizes the total loss of the system, with w representing the model parameters to be estimated. The term $l(\mathrm{w}; x, y)$ denotes a loss function or the negative of a log-likelihood function [26], and $r(\mathrm{w}; \lambda)$ denotes the regularization term, where λ is the tuning parameter.

Considering $\mathscr{D}$ as a dataset dispersed across K parallel clients, we introduce $I = 1, 2, \ldots, n$ as the complete set of data indices. The subset of data on the k-th client is given by $\mathscr{D}_k = \{(x_i, y_i) | i \in I_k\}$, with $I_k \subseteq I$ denoting the indices subset associated with node k. The size of each node, n_k, is determined by the number of indices in I_k. The collection of indices $I_1, I_2, \ldots, I_K$ effectively partitions I. A fusion center (or central server) $\mathcal{C}$, is employed to facilitate the model training process.

Li and Yang (2022) [22] examined a scenario in which the loss function corresponds to a convex objective function derived from the likelihood of a generalized linear model (GLM). The score-matching representative (SMR) method enables each client/node to compute its representative data by aligning the score function—specifically, the gradient of the log-likelihood function l in (1) with respect to w—with the client's raw dataset [26]. This approach demonstrated that the SMR-based estimator $\tilde{\mathrm{w}}^{SMR}$ achieves accuracy equivalent to the ground truth estimator $\hat{\mathrm{w}}$. Furthermore, it was established that $\tilde{\mathrm{w}}^{SMR}$ and $\hat{\mathrm{w}}$ differ by an order of magnitude proportional to $\sqrt{\Delta}$, where $\Delta = \max_k \max_{i,j} \|x_i - x_j\|_2$ represents the maximum Euclidean distance (L2 norm) between any two data points within the dataset.

In the next section, we present a significant extension to our analysis to include general non-convex functions, particularly focusing on DNNs. Our goal is to enhance the effectiveness and applicability of representation learning in artificial intelligence by tackling the challenges of non-convex optimization problems. To achieve this goal, we will introduce innovative frameworks and techniques tailored specifically to accommodate the complexities inherent in non-convex optimization landscapes. By addressing these challenges, we seek to equip representation learning with the flexibility and strength necessary to solve various optimization and learning tasks. These efforts aim to transform the field of distributed machine learning and push the boundaries of AI research forward.

III. Methodology

In machine learning tasks, especially within DNNs utilized for a variety of standard classification tasks, the objective function often exhibits complicated non-convex characteristics. Consider an input feature vector $x_i \in \mathbb{R}^d$

for data sample i, and a corresponding label $y_i \in [M]$, where M denotes the number of potential classes. A commonly employed loss function is the cross-entropy loss. For example, when $M = 2$, the cross-entropy loss is defined as $l(\mathrm{w}, x, y) = -\sum_{i=1}^{N} y_i \log(f(\mathrm{w}, x_i))$, in which y_i represents the true label distribution for the input x_i, w denotes the model parameters, and f symbolizes the model function. In this analytical exploration, we disregard the regularization term in (1), thereby having $L(\mathrm{w}, x, y) = l(\mathrm{w}, x, y)$.

Optimization of this expected loss typically employs gradient-based algorithms such as Stochastic Gradient Descent (SGD) or Adam. These methods iteratively update the parameters w in the direction that reduces the expected loss, following the update rule $\mathrm{w}_{\text{new}} = \mathrm{w}_{\text{old}} - \eta \nabla_{\mathrm{w}} L(\mathrm{w}, x, y)$, where η represents the learning rate.

A. Representative-based Centralized Learning

We begin by exploring the representative-based centralized model training framework, utilizing the dataset $\mathscr{D}$. At each epoch, rather than selecting a batch of data of size B from $\mathscr{D}$, denoted as $\mathcal{B} = \{(x_i, y_i) \mid i \in I_{\mathcal{B}}\}$, to train the model f, we compute a representative (x^r, y^r) such that training the model with the representative[1] induces the same parameter updates as would training with the entire batch $\mathcal{B}$. This section presents key algorithms for constructing the representative that will be applied in distributed learning in Section III.B.

To ensure that a single virtual point can effectively represent the data within a batch, all data points in $\mathcal{B}$ are supposed to share the same label, such that $y_i = y^{\mathcal{B}}$ for all $i \in I_{\mathcal{B}}$. Hence, we set the representative label y^r equal to $y^{\mathcal{B}}$.

Let $\nabla_{\mathrm{w}} L(\mathcal{B}) = \sum_{i \in I_{\mathcal{B}}} \nabla_{\mathrm{w}} L(\mathrm{w}, x_i, y_i)$ represent the gradient with respect to w for the current model f based on training $\mathcal{B}$. Our objective is to determine $x^r \in \mathbb{R}^d$ such that the representative (x^r, y^r) generates a gradient on w for f that matches the mean gradient produced by the data points in $\mathcal{B}$, $\nabla_{\mathrm{w}} L(\mathcal{B})/B$. The optimization problem becomes searching for x^r that minimizes the representative loss l^r:

$$l^r = \left\| \nabla_{\mathrm{w}} L(\mathrm{w}, x^r, y^r) - \frac{\nabla_{\mathrm{w}} L(\mathcal{B})}{B} \right\|_2. \tag{2}$$

To ensure that the representative x^r remains close to the original data points in batch $\mathcal{B}$, thereby further preserving intuitive insight into the data, we search for x^r by adding a certain magnitude of perturbations δ to the mean of the data points, denoted as $\bar{x} = \frac{1}{B} \sum_{i \in I_{\mathcal{B}}} x_i$. This process is similar to adversarial training, with the distinction that our goal is to identify a perturbation δ^* that adheres to a magnitude constraint (e.g., $|\delta|_2 \leq s$ for a given norm p and a specified constant s) to minimize the loss l^r. The optimization problem is thus formulated as:

$$\delta^* = \underset{\|\delta\|_2 \leq s}{\arg\min} \left\| \nabla_{\mathrm{w}} L(\mathrm{w}, \bar{x} + \delta, y^r) - \frac{\nabla_{\mathrm{w}} L(\mathcal{B})}{B} \right\|_2, \tag{3}$$

where δ^* is determined through gradient descent for a total H epochs. Ultimately, we set $x^r = \bar{x} + \delta^*$, and effectively construct (x^r, y^r) as the representative that closely aligns with the batch's original data while also minimizing the targeted loss function.

Once the representative point (x^r, y^r) is determined, we train the model using this synthesized point instead of the raw data from batch $\mathcal{B}$. The gradient with respect to the model parameters at this point is denoted as $\nabla \mathrm{w}^r = \nabla_{\mathrm{w}} L(\mathrm{w}, x^r, y^r)$. Model parameters are then updated based on the gradient $B \nabla \mathrm{w}^r$ for the current epoch.

Furthermore, we acknowledge the challenge of achieving an exact match between the gradient produced by the representative and that generated by the full data batch. There exists a discrepancy between $B \nabla \mathrm{w}^r$ and $\nabla_{\mathrm{w}} L(\mathcal{B})$. To address this, we introduce an error residual term τ_i to account for the difference between these two gradient terms. This allows us to integrate the residual error into the computation of x^r at each round of training. Consequently, the optimization problem for determining δ^* is updated as follows:

$$\delta^* = \underset{\|\delta\|_2 \leq s}{\arg\min} \left\| \nabla_{\mathrm{w}} L(\mathrm{w}, \bar{x} + \delta, y^r) - \frac{\nabla_{\mathrm{w}} L(\mathcal{B})}{B} + \tau \right\|_2, \tag{4}$$

where $\tau = \nabla \mathrm{w}^r - \nabla_{\mathrm{w}} L(\mathcal{B})/B$ represents the residual error accrued in the previous epoch. Simulation results presented in Section IV demonstrate that incorporating the residual error significantly enhances the efficacy of representative-based training.

The overall optimization process in this representative-based central learning framework is summarized as follows:

$$\min_{\mathrm{w}} \mathbb{E}_{\mathcal{B} \sim \mathcal{D}} L(\mathrm{w}, x^r, y^r), \tag{5}$$

with x^r, y^r denoting the representative of batch $\mathcal{B}$, where $x^r = \bar{x} + \delta^*$, and δ^* is derived according to (3) at every epoch. The details of this proposed framework are shown in Algorithm 1.

It is important to highlight that synthesizing all data points in batch $\mathcal{B}$ into a single virtual representative simplifies the visualization of the training process for humans. In each training epoch, observers need only focus on a single data point rather than B individual points.

[1]The representative (x^r, y^r) preserves the data dimensionality of the raw data points, which facilitates its interpretability and comprehensibility for humans.

Algorithm 1: Representative-based Centralized Learning

Input: Initialize model parameters w, dataset $\mathcal{D}$, batch size B, learning rate η_{w}

Output: Trained model

Initialize residual term $\tau = 0$; **for** $t = 1$ **to** T **do**
  Sample a batch of data $\mathcal{B}$ with same label $y^{\mathcal{B}}$;
  Compute the gradient of the raw data in $\mathcal{B}$ on model parameters, denoted by $\nabla_{\mathrm{w}} L(\mathcal{B})$;
  Compute the representative of the batch (x^r, y^r) such that $x^r = \bar{x} + \delta^*$ as in (4) and $y^r = y^{\mathcal{B}}$;
  Update model parameters using the gradient computed from (x^r, y^r): $\mathrm{w} = \mathrm{w} - \eta_{\mathrm{w}} B \nabla \mathrm{w}^r$;
  Update the residual term: $\tau = \nabla \mathrm{w}^r - \nabla_{\mathrm{w}} L(\mathcal{B})/B$;
return w

This significantly aids human oversight by facilitating the detection of outliers and enhancing the comprehension of the training dynamics, among other benefits.

B. Representative Data Fusion for Distributed Learning

In the distributed learning paradigm, datasets are dispersed across K clients. This setup reduces to the centralized training as a special case when $K = 1$. Traditionally, Federated Learning (FL) techniques, particularly the FedAVG [13] serving as the baseline comparison model in this paper, have been employed to avoid raw data transmission and preserve privacy. In this section, we introduce a representative-based distributed learning strategy that not only preserves the benefits but also enhances human comprehension of the training data. Moreover, this method has the potential to exceed FL techniques in convergence speed and accuracy.

Consider a scenario where the total dataset $\mathcal{D}$ is distributed among K clients, with each client possessing a subset of the dataset, denoted by $\mathcal{D}_k$. Similar to the FL methodology, at the beginning of each epoch, the server broadcasts the global model parameters w to all clients. Subsequently, each client randomly samples a batch of data points that share the same label. Then, a representative (x^k, y^k) is computed based on the data points in the batch and the global model. This representative, along with the batch size B_k, is then transmitted back to the server. Utilizing these representatives, the server updates the global model. In particular, each client maintains a residual term to correct the gradient discrepancy between its representative and the entirety of its batch data. This procedure is iterated over T rounds to refine the global model.

For the sake of clarity, we assume a uniform batch size B across all clients. This constraint can be adapted to heterogeneous batch sizes to reflect real-world application scenarios. The steps of this representative-based distributed learning strategy are shown in Algorithm 2.

Algorithm 2: Representative-based Distributed Learning with Server and Multiple Clients

Input: Model parameters w, batch size B, number of clients K, learning rate η_{w}, global training epoch T and local training epoch H

Output: Globally trained model

Server executes:
Initialize global model parameters w_0;
for $t = 1$ ***to*** T **do**
  Select a subset of clients S_t for training;
  for *each client* $k \in S_t$ *in parallel* **do**
    Each client updates its representative: $(x_t^k, y_t^k) \leftarrow$ **ClientRep**$(k, \mathrm{w}_{t-1}, \tau_{t-1}^k)$;
    $\mathrm{w} \leftarrow \mathrm{w} - \eta_{\mathrm{w}} B \nabla_{\mathrm{w}} L(\mathrm{w}, x_t^k, y_t^k)$;
return w;
ClientRep(k, w, τ):
  Sample a batch of data $\mathcal{B}$ with same label $y^{\mathcal{B}}$;
  Compute the gradient of the raw data in $\mathcal{B}$ on model parameters, denoted by $\nabla_{\mathrm{w}} L(\mathcal{B})^k$;
  Each client learns δ^* as in (4) using H epochs and compute its representative (x^k, y^k) such that $x^k = \bar{x} + \delta^*$ and set $y^k = y^{\mathcal{B}}$;
  Update its residual term: $\tau^k = \nabla_{\mathrm{w}} L(\mathrm{w}, x^k, y^k) - \nabla_{\mathrm{w}} L(\mathcal{B})^k / B$;
  return (x^k, y^k);

Computational Complexity: The computational complexity in representative-based distributed learning is influenced by three main factors: 1) Participation Proportion P: This factor measures the percentage of clients involved in each computation round, indicating the extent of collaborative efforts utilized for model training across distributed clients at each stage. 2) Local Iterations H: This represents the number of training iterations performed by each client on its local dataset to calculate its representative in every round. H indicates the computational efforts towards ensuring the representative's impact closely mirrors that of the raw data. 3) Batch Size B: This parameter specifies the size of the local batch used for generating the representative. It is a measure of the level of abstractness in data representation. Exploring the trade-off among these parameters to discern their collective impact on system performance is a potential research direction. Given the page constraints, these considerations are reserved for future investigation.

IV. Simulation Results

A. Experimental Setup

In our experiments, we assess our methodology using two datasets. The first is the Image Segmentation dataset [27], a compact multivariate dataset comprising image data characterized by high-level, numeric-valued attributes. The second dataset is MNIST, a standard benchmark featuring 60,000 images of handwritten digits. For the distributed learning problem considered here, we adopt a non-Independent and Identically Distributed (non-IID) strategy, as discussed in [28, 29], to partition the dataset across various clients, ensuring each client contains data with the same label, thus representing non-IID data characteristics. Specifically, we allocate 200 images with the same label to each client, mirroring real-world scenarios of non-IID data distribution. We examine the performance of two neural network architectures: a Multilayer Perceptron (MLP) model, which consists of several layers with ReLu activation functions; and a Convolutional Neural Network (CNN) with two convolutional layers and one linear layer, which is widely adopted for its capability in processing image data through effective feature extraction. To optimize these models, we employ Stochastic Gradient Descent (SGD) with a learning rate of 0.001, and each model is trained over 100 epochs. This strategy replicates real-world conditions of non-IID data distribution among network clients, which enables an assessment of the model's ability to generalize effectively.

B. Multivariate Data Experiments

In the centralized training approach, the baseline performance involves training the model with raw data points in a conventional manner, with the batch size set at 50. As illustrated in Table I, our representative-based centralized learning method achieves the same accuracy as the baseline, demonstrating that for a small dataset, adopting the representative-based approach does not sacrifice prediction accuracy while it enhances interpretability for human understanding. For the representative-based distributed training over non-IID data distribution, our method is compared against FedAVG in Table II, showing a slight performance difference of 0.95% lower than FedAVG. However, subsequent sections will demonstrate that as the complexity of the ML model increases, our method's performance is the same with or even surpasses that of FedAVG in the distributed setting.

TABLE I
Comparisons between different centralized learning methods on MLP model for Multivariate dataset.

Method	Accuracy	Batch Size
Baseline	94.29%	50
Our Method	94.29%	50

TABLE II
Comparisons between different distributed learning methods on MLP model for Multivariate datasets.

Method	Accuracy	Clients	Batch Size
FedAVG	92.38%	2	50
Our Method	91.43%	2	50

C. MNIST Data for Centralized Training

In this section, we assess our methodology by employing MLP and CNN models on the high-dimensional MNIST dataset using a centralized training approach. We compare the performance of our representative-based training against the baseline for both models, with the findings shown in Table III and Table IV, respectively. The results indicate a correlation between model complexity and the performance of our method: for the MLP model, the representative-based method has a performance 2.25% lower than the baseline. Conversely, with the more complex CNN model, our approach narrows the gap to just 0.13% away from the baseline. This suggests that for high-dimensional datasets such as MNIST, the representative-based approach in centralized training aligns more closely with baseline performances as model complexity increases.

TABLE III
Comparisons between different centralized learning methods on MLP model for MNIST dataset.

Method	Accuracy	Batch Size
Baseline	93.97%	64
Our Method	91.72%	64

TABLE IV
Comparisons between different distributed learning methods on CNN model for MNIST dataset.

Method	Accuracy	Batch Size
Baseline	98.87%	64
Our Method	98.74%	64

D. MNIST Data for Distributed Training

In this section, we assess our representative-based approach using MLP and CNN models on the MNIST dataset under a distributed learning scenario with non-IID data distribution across various clients. The performance comparisons of our method with FedAVG for both models are shown in Table V for the MLP model and Table VI for the CNN model. We note that for the MLP model our representative-based learning achieves comparable results to FedAVG, outperforming it by 0.18% with $K = 2$ clients, yet slightly underperforming by 0.32% when the client count increases to $K = 4$. Significantly, with the more complex CNN model, the representative method

consistently surpasses FedAVG, achieving a 0.14% improvement when $K = 2$ and a notable 1.62% increase when $K = 4$.

TABLE V
COMPARISONS BETWEEN DIFFERENT DISTRIBUTED LEARNING METHODS ON MLP MODEL FOR MNIST DATASET.

Method	Accuracy	Clients	Batch Size
FedAVG	90.75%	2	64
Our Method	90.93%	2	64
FedAVG	90.28%	4	64
Our Method	89.94%	4	64

TABLE VI
COMPARISONS BETWEEN DIFFERENT DISTRIBUTED LEARNING METHODS ON CNN MODEL FOR MNIST DATASET.

Method	Accuracy	Clients	Batch Size
FedAVG	98.18%	2	64
Our Method	98.32%	2	64
FedAVG	95.97%	4	64
Our Method	97.59%	4	64

The convergence patterns of representative-based distributed training for both MLP and CNN models, across varying numbers of clients, are shown in Fig. 1. It is observed that convergence is enhanced with a reduced number of clients for each model. Nonetheless, the convergence trajectory remains consistently smooth and fast across both models, irrespective of the client count, demonstrating no substantial decline in convergence speed as the number of clients rises from 2 to 4.

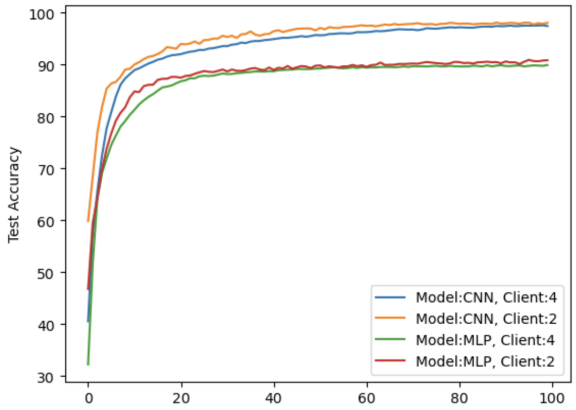

Fig. 1. Convergence of representative-based distributed learning for MINST dataset.

The convergence comparison between our method and FedAVG for training the CNN model is illustrated in Fig. 2. This comparison shows that our approach significantly outperforms FedAVG's training process in terms of both stability and convergence speed. FedAVG often exhibits poor convergence behavior due to the simple averaging of gradients from different clients, leading to considerable fluctuations in the learning curve. On the other hand, the improved accuracy, as detailed in Table VI, and the enhanced convergence observed in Fig. 2, can be attributed to the representative construction process. By integrating a specific magnitude of perturbations to the original dataset's average, the representative effectively introduces a form of regularization for each client. This form of regularization occurs as the server trains with ensembles of representatives from various clients in a batch, effectively minimizing the likelihood of divergence. These findings confirm that our representative-based strategy presents a viable and promising alternative to traditional distributed learning approaches.

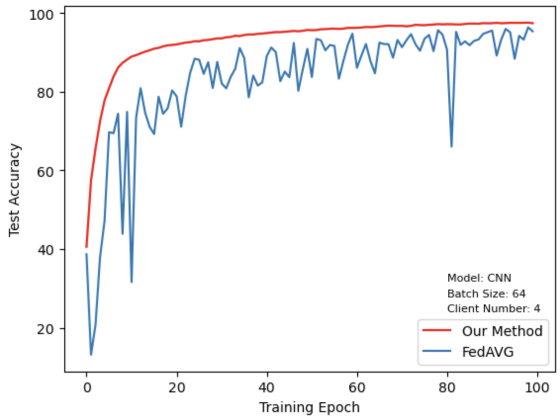

Fig. 2. Comparison of convergence between representative distributed learning and FedAVG for MNIST dataset.

E. Improvements Achieved by Incorporating the Residual

In constructing the representative, we incorporate residuals to adjust for discrepancies between the gradient induced by the representative and that induced by the raw data. The positive impact of including residuals on model accuracy is illustrated in Fig. 3, which contrasts the training convergence and accuracy of the CNN model in a distributed framework, comparing scenarios with and without residual terms. The adoption of residual corrections has notably enhanced system performance, acting as an effective counterbalance to any potential decline in representative data quality. Furthermore, this strategy helps mitigate any adverse effects or uncertainties that arise from the process of representative aggregation at the server.

V. DISSCUSION

The simulations presented above demonstrate that our method can match or surpass the accuracy and convergence rates of conventional Federated Learning (FL) techniques, such as FedAVG. This advantage becomes increasingly apparent with the rising complexity of the model and the growth in the number of clients.

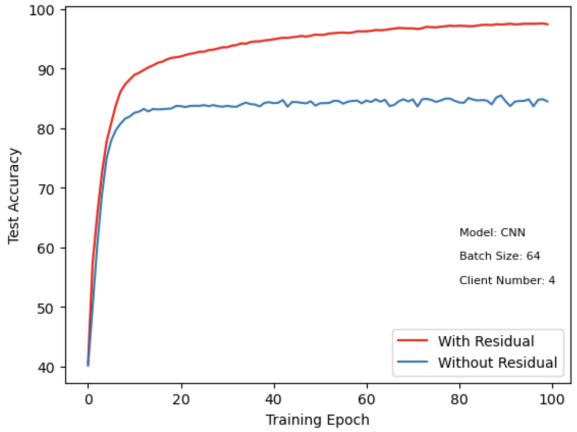

Fig. 3. Convergence of representative-based distributed learning for MINST dataset with and without residual.

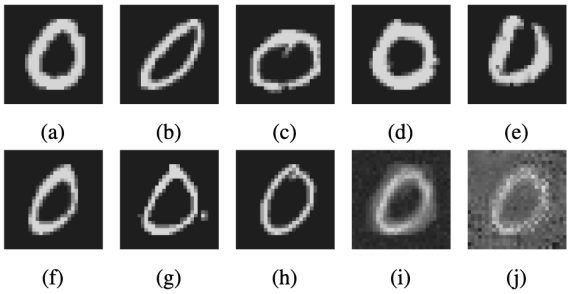

Fig. 4. Representatives constructed from raw images with MLP and CNN models.

This enhancement is likely due to the representative construction process, which involves introducing perturbations to the average of the raw dataset. This process effectively acts as a form of regularization, ensuring the maintenance of high-quality, informative inputs. Training the server with ensembles of these representatives helps to stabilize the learning process, in contrast to merely averaging diverse gradients. Such regularization facilitates more effective learning from representative data formats, a benefit that becomes more evident in complex models.

It is worth noting that one primary advantage of our approach is that it enhances the interpretability and comprehensibility of the data processing phase and improves the effectiveness of data transmission between the servers and the clients in a distributed learning setup. Unlike traditional compression or quantization methods, the representative technique allows for transmitting visually comprehensible information, facilitating a more intuitive understanding of the shared data among participants in the distributed network. In Fig. 4, we present a visualization of the representative for eight MNIST images labeled 0, ranging from (a) to (h), with (i) illustrating the representative generated via MLP and (j) via CNN. Notably, the representative from the simpler MLP model (i) appears closer to a natural image with label 0, whereas the CNN-generated representative (j) exhibits additional perturbations, particularly in the background. This observation suggests that constructing representatives with more complex models like CNNs necessitates more perturbations to the original input. This can add difficulty to human visualization and perception, indicating a trade-off between model complexity and the ease of interpreting representatives.

VI. Conclusions and Future Work

This study introduced a novel representative-based methodology that transforms multiple data points from a client into a single virtual data point, establishing a new approach in data processing and aggregation for distributed learning systems. This method enhances privacy and communication efficiency, which are crucial benefits in distributed learning contexts. Moreover, its capacity to simplify complex, diverse datasets into digestible forms facilitates the complex dynamics of human-ML collaboration. Simulation outcomes indicate that our approach can match or exceed the accuracy and convergence rates of traditional Federated Learning methods, particularly as model complexity increases and with the growth in the number of clients.

Moving forward, we will expand our analysis by incorporating communication requirements and conducting more comprehensive comparisons with a wider range of architectures to validate our findings. Additionally, we will explore how human oversight and interaction can be incorporated into the representative-based learning and decision-making process, aiming to improve situational awareness and achieve a seamless integration between human intuition and machine intelligence.

References

[1] P. K. Varshney, *Distributed detection and data fusion*. Springer Science & Business Media, 2012.

[2] V. V. Veeravalli and P. K. Varshney, "Distributed inference in wireless sensor networks," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 370, no. 1958, pp. 100–117, 2012.

[3] B. Kailkhura, V. S. S. Nadendla, and P. K. Varshney, "Distributed inference in the presence of eavesdroppers: A survey," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 40–46, 2015.

[4] S. Zhang, B. Geng, P. K. Varshney, and M. Rangaswamy, "Fusion of deep neural networks for activity recognition: A regular vine copula based approach," in *2019 22th International Conference on Information Fusion (FUSION)*. IEEE, 2019, pp. 1–7.

[5] C. Quan, S. Bulusu, B. Geng, Y. S. Han, N. Sriranga, and P. K. Varshney, "On ordered transmission based distributed gaussian shift-in-mean detection under byzantine attacks," *IEEE Transactions on Signal Processing*, 2023.

[6] M. Assran, N. Loizou, N. Ballas, and M. Rabbat, "Stochastic gradient push for distributed deep learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 344–353.

[7] Y. Liu, W. Xu, G. Wu, Z. Tian, and Q. Ling, "Communication-censored ADMM for decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2565–2579, 2019.

[8] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[9] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, D. Papadopoulos, and Q. Yang, "Secureboost: A lossless federated learning framework," *IEEE Intelligent Systems*, vol. 36, no. 6, pp. 87–98, 2021.

[10] Q. Li, Z. Wen, and B. He, "Practical federated gradient boosting decision trees," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 4642–4649.

[11] Y.-R. Chen, A. Rezapour, and W.-G. Tzeng, "Privacy-preserving ridge regression on distributed data," *Information Sciences*, vol. 451, pp. 34–49, 2018.

[12] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," *Advances in neural information processing systems*, vol. 30, 2017.

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

[14] S. U. Stich, "Local SGD converges fast and communicates little," *arXiv preprint arXiv:1805.09767*, 2018.

[15] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[16] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, "A unified theory of decentralized sgd with changing topology and local updates," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5381–5393.

[17] C. Quan, N. Sriranga, H. Yang, Y. S. Han, B. Geng, and P. K. Varshney, "Efficient ordered-transmission based distributed detection under data falsification attacks," *IEEE Signal Processing Letters*, vol. 30, pp. 145–149, 2023.

[18] B. Geng, Q. Li, and P. K. Varshney, "Human decision making with bounded rationality," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 5493–5497.

[19] B. Geng, X. Cheng, S. Brahma, D. Kellen, and P. K. Varshney, "Collaborative human decision making with heterogeneous agents," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 2, pp. 469–479, 2021.

[20] B. Geng, Q. Chen, and P. K. Varshney, "Cognitive memory constrained human decision making based on multi-source information," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5325–5329.

[21] B. Geng, S. Brahma, T. Wimalajeewa, P. K. Varshney, and M. Rangaswamy, "Prospect theoretic utility based human decision making in multi-agent systems," *IEEE Transactions on Signal Processing*, vol. 68, pp. 1091–1104, 2020.

[22] K. Li and J. Yang, "Score-matching representative approach for big data analysis with generalized linear models," *Electronic Journal of Statistics*, vol. 16, no. 1, pp. 592–635, 2022.

[23] B. Zhao, K. R. Mopuri, and H. Bilen, "Dataset condensation with gradient matching," *arXiv preprint arXiv:2006.05929*, 2020.

[24] J. Domke, "Generic methods for optimization-based modeling," in *Artificial Intelligence and Statistics*. PMLR, 2012, pp. 318–326.

[25] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu, "Dataset distillation by matching training trajectories," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4750–4759.

[26] P. McCullagh and J. Nelder, *Generalized Linear Models*, 2nd ed. Chapman and Hall/CRC, 1989.

[27] "Image Segmentation," UCI Machine Learning Repository, 1990, DOI: https://doi.org/10.24432/C5GP4N.

[28] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[29] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.