Federated Learning with Social Data Sharing

Mahran Jazi[1], Ilai Bistritz[1], Nicholas Bambos[2], Irad Ben-Gal[1]
[1]Department of Industrial Engineering, Tel Aviv University, Israel
[2]Department of Electrical Engineering, Stanford University, USA
mohranjazi@mail.tau.ac.il, ilaibistritz@tauex.tau.ac.il, bambos@stanford.edu, bengal@tauex.tau.ac.il

***Abstract*—Collaboration between edge devices has the potential to dramatically scale up machine learning (ML) through access to an unprecedented quantity of data. Federated learning (FL) is a collaborative algorithm in which clients learn from each other without sharing their private data. However, edge devices tend to have various data distributions since they are naturally exposed to different data sources. This heterogeneity of the data, also known as non-IID data distributions, degrades the performance of FL. We study how data sharing among users can mitigate this performance degradation. Data sharing among users can occur naturally in a social group (e.g., friends, colleagues, and family) or can be incentivized by the platform based on different criteria. We test the performance gains of data sharing for several standard ML models and common datasets. We empirically show that modest data sharing between members of a social group among the different experiments boosts the learning performance significantly for the non-IID case. We also show that data sharing can, surprisingly, boost performance for the IID case. By normalizing the dataset sizes, we verify that this performance boost is significant even if data sharing does not increase the number of data points per client. Data sharing is thus a simple and efficient technique for improving FL.**

I. Introduction

Users manage many of their daily digital activities through edge devices. Each edge client collects private data, such as pictures, videos, and texts. User applications increasingly employ machine learning (ML) tools that use private data for training and modeling to automate tasks and personalize services. However, while the data is generated on the edge client, and the application also runs on the edge client, ML training is traditionally performed on central servers with greater computational power. These central servers collect a significant portion of the users' private data. Hence, traditional centralized ML often violates users' privacy, which is recognized as a major concern [1], [2]. Moreover, transmitting private data to the server is a huge communication overhead. To address these concerns, scalable and efficient on-device learning techniques are needed to avoid sending private data to the central server and reduce communication overload.

One of the first breakthroughs in on-device learning was federated learning (FL) [1]. In FL, each client has a private training dataset that is never uploaded to the server. As Figure 1 shows (ignoring the dashed lines), each client instead uploads its local model (e.g., weights of its deep neural network (DNN)) to the server that averages them to a single global model and sends it back to the clients. This process continues iteratively until all clients converge to the same model.

Despite its advantages [3], the performance of FL deteriorates dramatically when the data distribution is heterogeneous [4], which is commonly referred to as non-independent and identically distributed (Non-IID) data. Training non-IID data slows convergence since the gradient descent updates of clients "push" toward different local models, leading to a noisy "average" model. As a result, the global model achieves poor accuracy [4]. Since users typically generate non-IID data in many applications, the poor performance of FL, in this case, hinders its effectiveness in practice. In this paper, we observe that it is essential to distinguish between the user *generated* data and the *stored* data. Everyday data, such as pictures and texts on mobile phones, are shared with our social connections. Therefore, we are interested in how FL performs with users who generate non-IID data but partially share it with their trusted social connections. We demonstrate that even partial data sharing can dramatically improve the performance of FL over non-IID data and even improve performance for IID data.

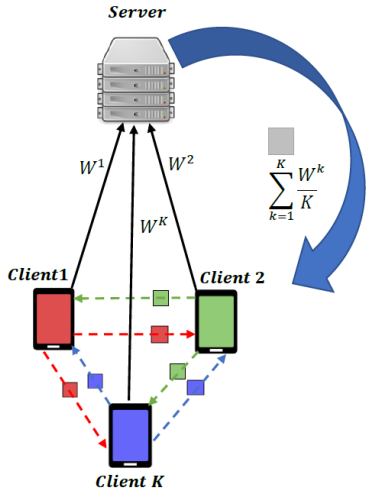

Fig. 1. The solid lines represent a traditional FL scheme, while the dashed lines represent data sharing between clients, termed "social federated learning"

A. Contributions

We propose data sharing between clients as a simple and efficient technique to boost FL performance, as illustrated by the dashed lines in Figure 1. Our results have implications for two different settings: 'active' and 'passive' data sharing.

In *active data sharing*, we modify the FL algorithm to instruct each client to share a target portion of its data with

a given subset of other clients. While this compromises some of the client's privacy, the key point is that the client never sends any data to the server but only shares data locally with their peers. The application on their device can incentivize users to send part of their data to their social connections (friends, family, or colleagues), with whom they can share data regularly. Thus, privacy violations are minimized or eliminated. The behavior of the users given the incentive mechanism can then be analyzed using game theory [5].

In *passive data sharing*, we run FL while picking the group of clients wisely. Since social groups share data between members regularly, grouping clients with social connections in mind results in significant data sharing between clients. For example, all our texts are shared in messaging applications, and we share pictures regularly with family and friends. Our results show that such grouping can boost FL performance even when compared with FL with no data sharing but the same number of data points per client (after data sharing).

Nevertheless, our experiments do not depend on how and when data was shared (actively or passively); instead, they study the performance of sharing a certain amount of data. We show that data sharing boosts performance for the non-IID **and**, somewhat surprisingly, in IID data settings. Furthermore, a small percentage of data sharing is enough to achieve a significant and almost maximal boost in performance. The key takeaway of this paper is, therefore: *Running FL on a social group of clients performs better than on an arbitrary group.*

B. Related Work

Data heterogeneity often leads to learning challenges in real-world distributed settings [6], [7], particularly for FL with non-IID data [8]. As expected, when the data are far from IID, FL performance significantly degrades. To mitigate this degradation, [4] proposed using a shared artificial dataset that is uniformly distributed over the data. The central server then distributes this dataset to all clients, making their data "more IID." However, this simple and elegant approach has significant drawbacks: it requires the central server to know the data distribution and clients to store external files. In [9] a scheme that only shares a limited amount of private data with the server was proposed. While this improves the users' privacy, it still leaks private information. We bypass these limitations by studying the data randomly shared between clients as part of their social interactions. Thus, the server does not need to know the data distribution, and clients do not need to store any additional data. Furthermore, we show that data sharing can boost FL performance even in the IID case.

Another approach to address non-IID data challenges is to change the FL training algorithm, typically based on SGD [10], [11]. Although SGD has theoretical guarantees, mainly for convex problems [12], [13], it has performed remarkably well in training DNNs, a highly nonconvex training problem. In FL problems with IID data distribution, such good performance is often maintained [14], even though there are no theoretical guarantees for this performance in nonconvex problems. Unfortunately, this is no longer the case for non-IID data, as FL can perform poorly both empirically and theoretically. Modifying the training algorithm [15]–[17] has improved performance for the non-IID case. Our proposed approach complements the SGD approach since we aim to improve data distribution through data sharing. Therefore, the data and SGD approaches can be combined to mitigate FL learning degradation in the non-IID case. Surprisingly, we show that data sharing can also boost the performance of nonconvex FL problems, even with IID data.

Clustered-FL [18]–[21] enhances FL by learning client clustering 'on the fly,' such that each cluster has a different optimal global model. This way, clients whose data distributions "push" toward a different global model will not affect each other. This elegant technique can mitigate non-IID data distributions in multiclass classification scenarios, where different clients learn and classify objects into different classes based on their local dataset. Instead, we are interested in the case where all clients have the same classification task. Thus, theoretically, the exact optimal classification model should be used, but the different distributions of the data among the clients hinder the convergence of this global model. This is often the case when the classes are distributed unequally among clients due to their private data and local exposure (e.g., dog owners will have more pictures of dogs).

II. Problem Formulation

In this section, we describe our social federated learning (SFL) scheme, which runs FL on a set of clients that share a portion of their data. Our experiments do not consider how data was shared (e.g., active or passive data sharing); instead, they report the benefits of sharing a certain amount of data.

Data sharing consistently improves learning performance compared to a conventional FL without data sharing, which is trained over the same number of data points after sharing. This comparison showed that the improvement offered by data sharing does not stem merely from having more data. The benchmark FL system with no data sharing has more original data points in the network than our SFL system.

We consider N clients engaged in a classification task where the goal is to learn a function that maps every input data point (element) to the correct class out of K possible options. Each client n has access to its own private dataset $\mathcal{D}_n = \{\boldsymbol{x}_i^n\}_{i=1}^{M_n}$ of M_n inputs and their corresponding labels $\{\boldsymbol{y}_i\}$. Each client n has a model (e.g., a DNN) with p parameters (e.g., weights), $\boldsymbol{\omega}^n \in \mathbb{R}^p$, and the architecture is identical across clients.

Let $l(\boldsymbol{\omega}, \boldsymbol{x}, \boldsymbol{y})$ be the loss on a training data point $(\boldsymbol{x}, \boldsymbol{y})$. The local training loss function of client n is then

$$\mathcal{L}_n\left(\boldsymbol{\omega}^n\right) \triangleq \sum_{\boldsymbol{x} \in \mathcal{D}_n} l\left(\boldsymbol{\omega}^n, \boldsymbol{x}, y\left(\boldsymbol{x}\right)\right). \tag{1}$$

The goal of the training in FL is to learn a common model $\boldsymbol{\omega}$ that minimizes the total loss across clients, defined as

$$\mathcal{L}\left(\boldsymbol{\omega}\right) = \sum_{n=1}^{N} \mathcal{L}_n\left(\boldsymbol{\omega}\right). \tag{2}$$

The iterations t of the FL algorithm we employ consist of two parts. First, the server collects the client models and computes the average model's parameters

$$\overline{\boldsymbol{\omega}}_t = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{\omega}_t^n. \tag{3}$$

Then, each client performs a local SGD step to update the average model, with a momentum parameter $0 \leq \beta < 1$:

$$\boldsymbol{\omega}_{t+1}^n = \overline{\boldsymbol{\omega}}_t - \eta_t v_{t+1}^n \tag{4}$$

where η_t is the step-size sequence and

$$v_{t+1}^n = \beta v_t^n + g_n(\overline{\boldsymbol{\omega}}_t) \tag{5}$$

which coincides with standard SGD for $\beta = 0$. The stochastic gradient $g_n(\overline{\boldsymbol{\omega}}_t)$ is taken with respect to a random subset of data points $\mathcal{S}_n \subset \mathcal{D}_n$ of size B (i.e., batch size):

$$g_n(\overline{\boldsymbol{\omega}}_t) = \sum_{\boldsymbol{x}\in\mathcal{S}_n} \nabla l(\overline{\boldsymbol{\omega}}_t, \boldsymbol{x}, y(\boldsymbol{x})). \tag{6}$$

III. Data Sharing Mechanism

We study the performance of FL with data sharing. After the data are shared, we use the FL algorithm described above. An essential aspect of data sharing is that clients never send their data to the server. Therefore, clients' privacy is maintained because their data are shared only locally with their social connections, which may occur without intervention.

Algorithm 1 shows how we assign data points to clients. The algorithm ensures that the number of data points for each client after data sharing is fixed, regardless of the amount of data shared. In practice, data sharing naturally increases the number of data points per client. However, our goal is to empirically show that data sharing boosts performance, not just because of larger datasets, which automatically improve learning performance. Hence, the FL benchmark system with $\Delta = 0\%$ uses the same final number M_n of data points per client, and since it has no overlap, this benchmark system *has more original data points* than our data sharing FL system.

Our results assume that each pair of clients shares the same quantity of data. The reason for this simplifying assumption is to quantify the amount of data sharing using a single interpretable parameter Δ.

IV. Experimental Results

Data sharing improves performance by steering FL into better stationary points with non-convex losses. Hence, to evaluate the proposed SFL's performance, we conducted extensive empirical analysis on such practical non-convex losses, which arise in common ML models. The experiments were done using Google Colab Pro with a high GPU(A100) and RAM to run the complex datasets and models in the Python environment. We report the percentage of accurately classified data points in the test dataset (i.e., "accuracy") using the model FL obtained after the training.

Algorithm 1 Data Sharing Mechanism

Initialization: Let M_0 and M be the initial and final (post-data sharing) number of data points per client, respectively. Let $\mathcal{D}$ be the set of all data points. Let $0 \leq \Delta \leq 1$ be the ratio of shared data. For the process to be feasible, we require $M_0 > \Delta M$ and $M_0 + N\Delta M \leq M$.

1) **Initial Data Points:** Assign M_0 data points from $\mathcal{D}$ to each client, with no overlap, and mark these data points as assigned.
2) **Data Sharing:** For any two clients n, m, send ΔM data points from client n to client m, uniformly at random from client n's initial dataset from Step 1.
3) **Equalize the dataset sizes:** Let M_n be the number of data points of client n after Step 2. Assign additional $M - M_n$ unassigned data points from $\mathcal{D}$ to client n, with no overlap between the clients.

To provide a common estimate and benchmark, we used the following standard datasets:

- MNIST: contains 70,000 pictures of handwritten digits from 0 to 9. We split the dataset into 60,000 pictures for training and 10,000 pictures for testing [22].
- CIFAR-10: contains 60,000 32x32 color images of objects from 10 classes (0–9) [23]. We split the dataset into 50,000 pictures for training and 10,000 for testing.
- CIFAR-100: contains 60,000 32x32 color images of objects from 100 classes [23]. We split the dataset into 50,000 pictures for training and 10,000 for testing.

To establish SFL as a model-independent approach, we tested four conventional ML models: logistic regression (LR), a simple multilayer perceptron with two hidden layers with 200 units each; ReLU activation functions (2NN) [1], [4]; the LeNet-5 convolutional neural network (CNN) [22]; and a DNN (called DenseNet-121) [24], representing four levels of gradual increasing complexity. We trained these models using SGD (as described in Section 2) over 100 communication rounds, with a batch size of $B = 32$, a local epoch of $E = 1$ (so SGD went over the local dataset once), a learning rate (step-size) of $\eta = 0.01$, and a momentum parameter of $\beta = 0.9$. Each experiment was executed for ten realizations; the curves are the average results. Error bars indicate one standard deviation above and below the curve.

A comparison between the proposed data-sharing mechanism and the Federated Learning (FL) algorithm FedAvg [1] was conducted to substantiate the effectiveness of data-sharing. Our results are compared with lower and upper bounds, namely, the benchmarks of using the same models and datasets with zero data sharing (i.e., no data sharing (FedAvg)) and complete data sharing (where all clients share all of their data). Full data sharing is equivalent to a centralized algorithm that obtains all the data points in the network and takes a gradient step using an average of N stochastic gradients.

A. IID Data Distribution

The SFL results for the IID data are presented in Figure 2. In this setting, there are ten clients, and before data sharing, each client has only $M_0 = 1,000$ random data points from the specific dataset (MNIST, CIFAR-10, or CIFAR-100); thus, the data distributions among the clients are similar. After data sharing, the final number of data points is $M = 6,000$. Thus, here, "20% data sharing" denotes that each client shares the same random subset of 20% of its data with all other clients.

Surprisingly, data sharing significantly improves the FL performance even for the IID case across the four models and three datasets. In the non-IID case, part of the SFL improvement can be attributed to making the data "more IID," but this is, of course, irrelevant when the data is already fully IID. Nevertheless, data-sharing is more robust than making the dataset distributions similar - it makes the datasets *themselves* more similar. This encouraging finding is reminiscent of ensemble learning, and bagging [25]. In bagging, an ensemble of weak learners is trained on bootstrapped data sets that are samples with replacements from the original dataset and, therefore, have overlapping data points like in data-sharing.

The SFL performance improvement in the IID case is smaller than in the non-IID case; however, this difference is mainly explained by the better accuracy achieved in the IID case, even with no data sharing. Hence, as observed above, the data-sharing effect is naturally less significant when the classification task is more straightforward.

B. Non-IID Data Distribution

The SFL results for the non-IID data are presented in Figure 3. In this setting, there are ten clients, and before data sharing, client i has only data points of class i in MNIST or CIFAR-10, for $i = 1, \ldots, 10$, or only data points of classes $10(i-1), 10(i-1)+1, \ldots, 10(i-1)+9$ in CIFAR-100. Hence, no classes overlap between the clients for either dataset. This represents the "most" non-IID setting possible. The initial number of data points per client was $M_0 = 1000$. The final number of data points, after data sharing, is $M = 6,000$. Here, "20% data sharing" means that each client shares the same random subset of 20% of its data with all other clients (which is different than having $\Delta = 0.2$ in Algorithm 1).

Our results confirm that the performance of the conventional (FedAvg) zero-sharing FL (red line in Figure 3) in such settings is poor [4]. Nevertheless, applying the SFL with modest data sharing of 20% is enough to significantly boost the learning performance across all four models and all three datasets. The figure emphasizes the diminishing-return effects of the data sharing percentage and the number of communication rounds by analyzing the settings with 0%, 20%, 40%, or 100% data sharing, where all the clients always have a private dataset of 6,000 data points. Hence, the performance improvement is not due to a larger training dataset but due to data-sharing. This diminishing return effect is encouraging since it indicates there is little to be gained by increasing the data-sharing percentage (e.g., above 20%) or the number of communication rounds (e.g., above 40). In general, the effect of data sharing is more prominent when the model faces harder learning tasks, such as the complex DenseNet-121 model and the CIFAR-100 dataset. The effectiveness of SFL aligns with the balance between model complexity and data complexity. Matching the model complexity to that of the dataset is crucial to fully benefiting from the SFL effect. The performance improvement in the non-IID case is consistent with the idea that data sharing makes the data "less" non-IID, therefore improving the convergence of FL to a global model. However, the improvement cannot be explained by making the data more IID alone, since we have seen that SFL improves performance even for the IID case.

V. Conclusion

Motivated by the poor performance of FL for non-IID data, we studied the effect of data sharing between users. This data sharing is done locally within a social group. Therefore, clients never send data to the central server; they only send it to their social connections (e.g. family, friends, and colleagues). We showed empirically that data sharing can dramatically improve the performance for the non-IID case. Surprisingly, data sharing improved performance even for the IID case. Moreover, only a small percentage of shared data was required to significantly boost performance for the non-IID and IID cases.

Data sharing can occur naturally between social connections (passive data sharing) or can be incentivized through the platform (active data sharing). Our work showed that data sharing from either source can be highly effective for FL.

This paper focused on data sharing within a social group, where data is shared uniformly across the group. Future research aims to study data sharing over real-life social networks, where each client shares data only with their neighbors.

Acknowledgements

The authors gratefully acknowledge funding from the Koret Foundation grant for Smart Cities and Digital Living 2030.

References

[1] 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.

[2] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," *arXiv preprint arXiv:1511.03575*, 2015.

[3] L. Zhang, J. Xu, P. Vijayakumar, P. K. Sharma, and U. Ghosh, "Homomorphic encryption-based privacy-preserving federated learning in iot-enabled healthcare system," *IEEE Transactions on Network Science and Engineering*, 2022.

[4] 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.

[5] A. Buratto, E. Guerra, M. Miozzo, P. Dini, and L. Badia, "Energy minimization for participatory federated learning in iot analyzed via game theory," in *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 2024, pp. 249–254.

[6] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, "The non-iid data quagmire of decentralized machine learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4387–4398.

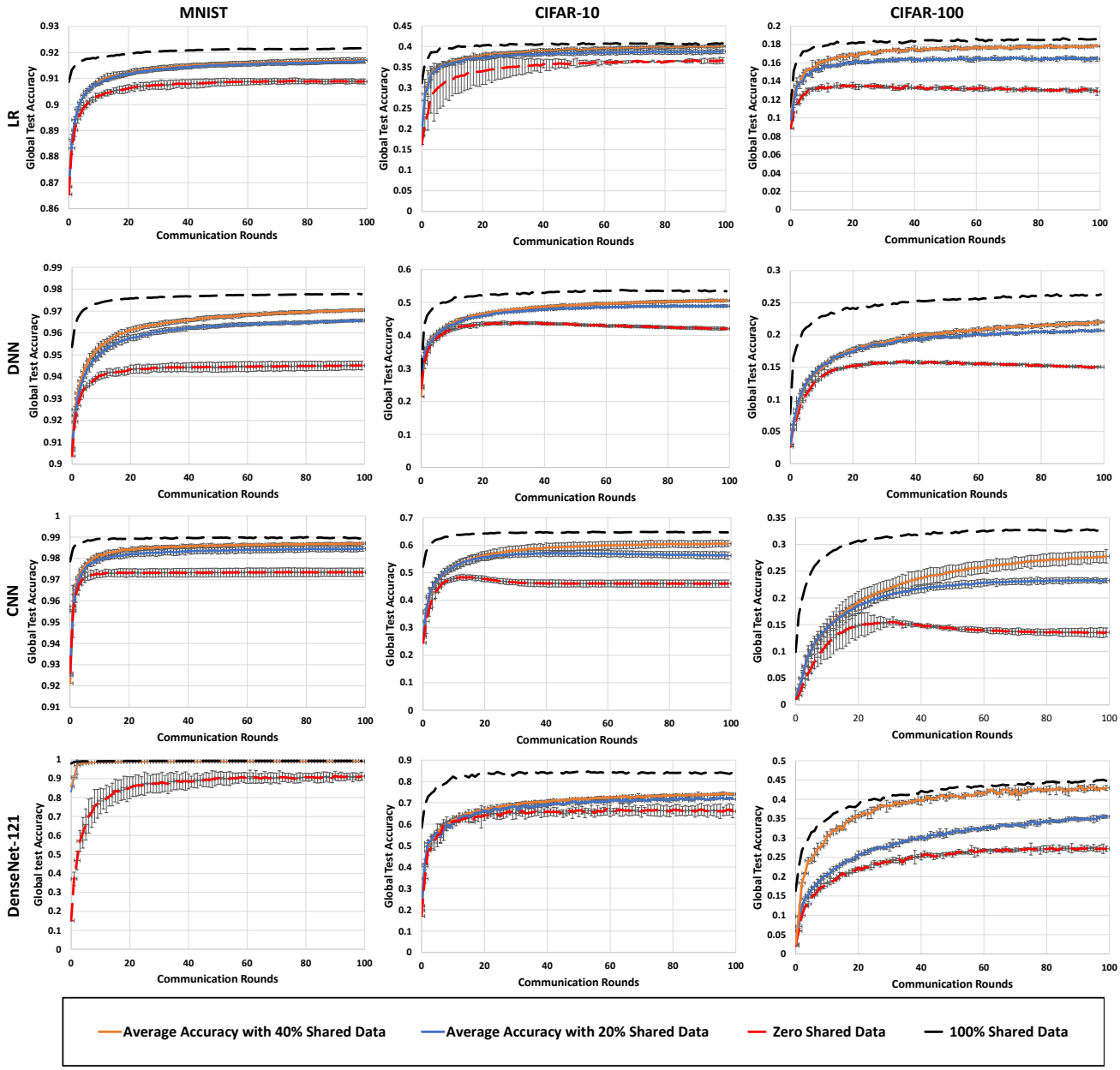

Fig. 2. Data sharing with IID data distributions. The columns correspond to the MNIST, CIFAR-10, and CIFAR-100 datasets. The rows correspond to the models LR, 2NN, LeNet-5, and DenseNet-121.

[7] T. Sery, N. Shlezinger, K. Cohen, and Y. C. Eldar, "Over-the-air federated learning from heterogeneous data," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3796–3811, 2021.

[8] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021.

[9] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, "Hybrid-fl for wireless networks: Cooperative learning mechanism using non-iid data," in *ICC 2020-2020 IEEE International Conference On Communications (ICC)*. IEEE, 2020, pp. 1–7.

[10] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research*, vol. 12, no. 7, 2011.

[11] D. P. Kingma and L. Ba, "Jimmy.(2015). ADAM: A method for stochastic optimization," in *International conference on learning representations*.

[12] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," *arXiv preprint arXiv:1109.5647*, 2011.

[13] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[14] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

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

[16] X. Li, M. JIANG, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," in *Interna-*

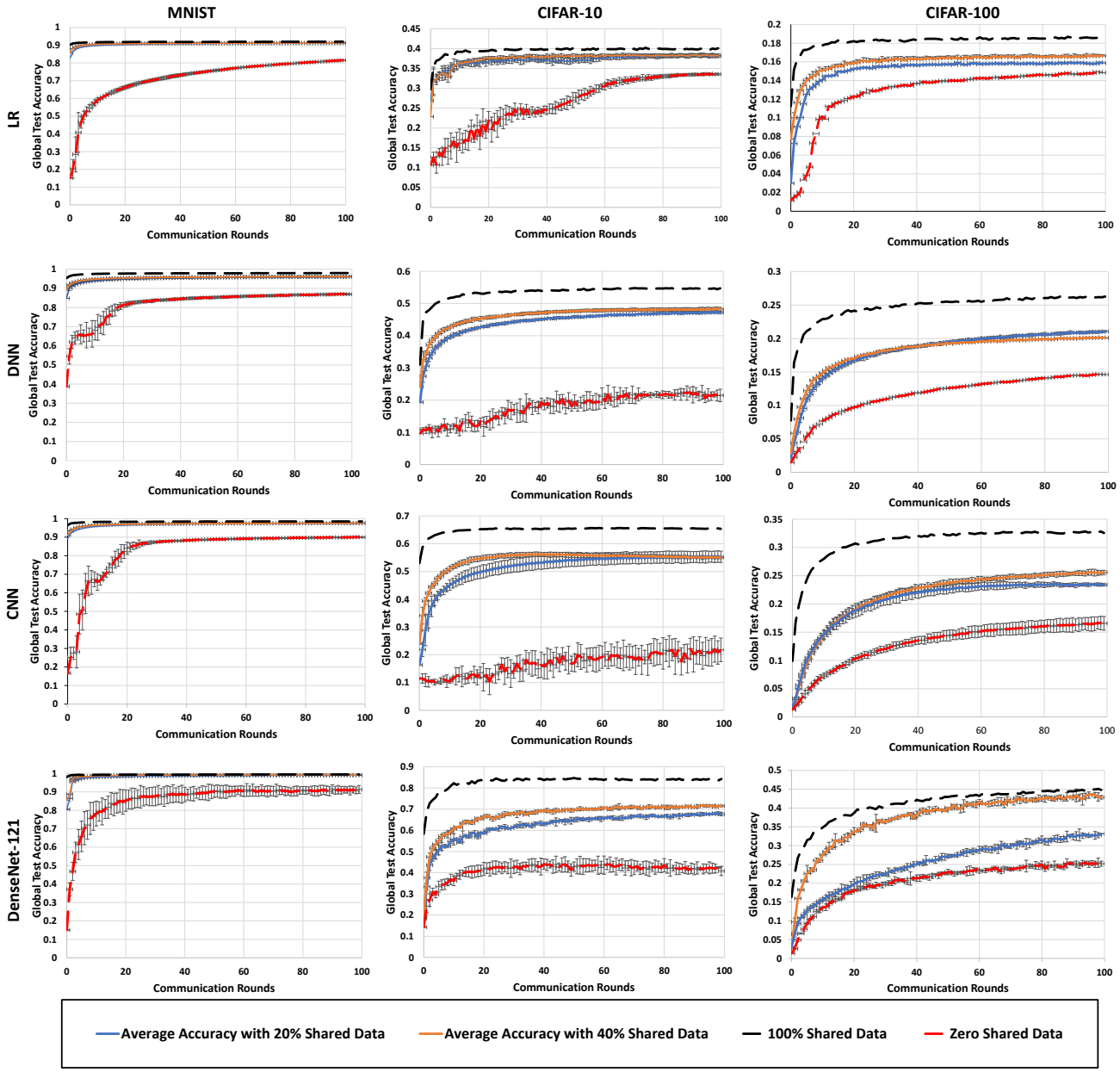

Fig. 3. Data sharing with non-IID data distributions. The columns correspond to the MNIST, CIFAR-10, and CIFAR-100 datasets. The rows correspond to the models LR, 2NN, LeNet-5, and DenseNet-121.

tional Conference on Learning Representations, 2020.

[17] D. M. S. Bhatti and H. Nam, "A performance efficient approach of global training in federated learning," in *2023 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 2023, pp. 112–115.

[18] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 8, pp. 3710–3722, 2020.

[19] X. Ouyang, Z. Xie, J. Zhou, J. Huang, and G. Xing, "Clusterfl: a similarity-aware federated learning system for human activity recognition," in *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, 2021, pp. 54–66.

[20] L. Yang, J. Huang, W. Lin, and J. Cao, "Personalized federated learning on non-iid data via group-based meta-learning," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2022.

[21] B. Gong, T. Xing, Z. Liu, W. Xi, and X. Chen, "Adaptive client clustering for efficient federated learning over non-iid and imbalanced data," *IEEE Transactions on Big Data*, 2022.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[23] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[25] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 161–168.