Federated Learning via Decentralized Dataset Distillation in Resource-Constrained Edge Environments

Rui Song*, 1, 2, Dai Liu*, 2, Dave Zhenyu Chen2, Andreas Festag1, 3, Carsten Trinitis2, Martin Schulz2, Alois Knoll2

1Fraunhofer IVI  2Technical University of Munich  3Technische Hochschule Ingolstadt

{rui.song, andreas.festag}@ivi.fraunhofer.de
{rui.song, dai.liu, zhenyu.chen, carsten.trinitis}@tum.de,
{schulzm, knoll}@in.tum.de, andreas.festag@thi.de

Abstract—In federated learning, all networked clients contribute to the model training cooperatively. However, with model sizes increasing, even sharing the trained partial models often leads to severe communication bottlenecks in underlying networks, especially when communicated iteratively. In this paper, we introduce a federated learning framework FedD3 requiring only one-shot communication by integrating dataset distillation instances. Instead of sharing model updates in other federated learning approaches, FedD3 allows the connected clients to distill the local datasets independently, and then aggregates those decentralized distilled datasets (e.g. a few unrecognizable images) from networks for model training. Our experimental results show that FedD3 significantly outperforms other federated learning frameworks in terms of needed communication volumes, while it provides the additional benefit to be able to balance the trade-off between accuracy and communication cost, depending on usage scenario or target dataset. For instance, for training an AlexNet model on CIFAR-10 with 10 clients under non-independent and identically distributed (Non-IID) setting, FedD3 can either increase the accuracy by over 71% with a similar communication volume, or save 98% of communication volume, while reaching the same accuracy, compared to other one-shot federated learning approaches.

I. INTRODUCTION

Federated learning has become an emerging paradigm for collaborative learning in large-scale distributed systems with a massive number of networked clients, such as smartphones, connected vehicles or edge devices. Due to the limited bandwidth between clients [1], previous research [2]–[8] attempts to speed up convergence and improve communication efficiency. However, for modern neural networks with over hundreds of million parameters, this kind of cooperative optimization still leads to extreme communication volumes, which require substantial network bandwidth (up to the Gbps level [9]) in order to work reliably and efficiently. This drawback hinders any large-scale deployment of federated learning models in commercial wireless mobile networks, e.g., vehicular communication networks [10] or industrial sensor networks [11].

Motivated by this communication bottleneck, prior federated learning algorithms attempt to reduce the number of communication rounds and with that the communication volume, to reach a good learning performance. Guha et al. [12] introduce a one-shot federated learning approach aimed at reducing communication overhead during the training process of a support vector machine (SVM). By exchanging information in a single communication round, it offers significant efficiency improvements. Kasturi et al. [13] provide a fusion federated learning method that uploads both model and data distribution to the server, but characterizing a distribution of a real dataset can be difficult. The one-shot federated learning based on knowledge transfer from Li et al. [14] is general, but it requires additional communication overhead to transmit multiple student models to the server.

Inspired by the one-shot scheme [12], we introduce a federated learning training scheme with one-shot communication via dataset distillation [15]–[18]. Intuitively, dramatically smaller but more informative datasets, which include dense features, are synthesized and transmitted. This way, more informative training data is transmitted across the limited bandwidth without any privacy violation.

Specifically, we introduce a novel federated learning framework incorporating dataset distillation, FedD3, which is shown in Fig. 1. It enables efficient federated learning via transmitting the locally distilled dataset to the server in a one-shot manner, which can be applied as a pre-trained model and used for personalized [19], [20] and fairness-aware [21] learning. Note that dataset distillation can keep the advantage of privacy in federated learning [22]–[30]. It anonymously maps distilled datasets from the original client data without any exposure, which is analogous to the shared model parameters in previous federated learning methods, but substantially more efficient and effective.

*Equal Contribution.

This work was supported by the German Federal Ministry for Digital and Transport (BMVI) in the projects “KIVI – Ki im Verkehr Ingolstadt” and “5Gdug – 5G Innovation Concept Ingolstadt”.

Authorized licensed use limited to: University of Southern Queensland. Downloaded on September 05,2023 at 06:23:31 UTC from IEEE Xplore. Restrictions apply. The authors would like to thank all the reviewers for their valuable comments and suggestions, which greatly contributed to the improvement of this paper.
We perform an extensive analysis of our method in various scenarios to showcase the effectiveness of our method in massively distributed systems and on Non-IID (non-independent and identically distributed) datasets. Specifically, our experiments highlight the trade-off between accuracy and communication cost. To handle this trade-off, we propose a new evaluation metric, the $\gamma$-accuracy gain. By tuning the importance of accuracy gain $\gamma$ to the communication cost, the communication efficiency in federated learning is scored accordingly. We also investigate the effects of specific external parameters, including Non-IID datasets, number of clients and local contributions, and demonstrate a great potential for our framework in networks with constrained communication budgets in federated learning. We experimentally show that FedD3 has the following advantages: (i) Compared to conventional multi-shot federated learning, FedD3 significantly reduces the amount of bits that needs to be communicated, making our approach practically feasible even in low-bandwidth environments; (ii) Compared to other approaches for one-shot federated learning, FedD3 achieves a much better performance even with less communication volume, where the accuracy in a distributed system with 500 clients is enhanced by over $2.3 \times$ (from 42.08% to 94.74%) on Non-IID MNIST and $3.6 \times$ (from 10.74% to 38.27%) on Non-IID CIFAR-10 compared to FedAvg in one single round; (iii) Compared to centralized dataset distillation, FedD3 achieves much better results due to the broader data resource via federated learning.

**Contributions** To summarize, our contributions are as four-fold:

- We introduce a decentralized dataset distillation scheme in federated learning systems, where distilled data instead of models are uploaded to the server;
- We formulate and propose a novel framework, FedD3, for efficient federated learning in a one-shot manner, and demonstrate FedD3 with two different dataset distillation instances in clients;
- We propose a novel evaluation metric $\gamma$-accuracy gain, which can be used to tune the importance of accuracy
and analyze communication efficiency;

- We conduct an extensive analysis of the proposed framework. The experiments showcase the great potentials of our framework in networks with constrained communication budget in federated learning, especially considering the trade-off between accuracy and communication costs. The software implementation of FedD3 is publicly available as open source at GitHub\(^1\).

II. BACKGROUND AND RELATED WORK

Federated Learning. Federated learning was first introduced by McMahan et al. [1], where models can be learned collaboratively from decentralized data through model exchange between clients and a central server without violating privacy. The proposed federated learning scheme FedAvg [1] aggregates the received models and updates global model by averaging their parameters.

Compared to other distributed optimization approaches, federated optimization addresses more practical challenges, e.g., communication efficiency [2], data heterogeneity [31], privacy protection [32], system design [33], which enables a large-scale deployment in real-world application scenarios [34]–[43].

In a federated learning scenario, given a set of clients indexed by \( k \), machine learning models with weights \( w_i \) are trained individually on local client datasets \( D_k = \{(x_i)|i = 1, 2, ..., n_k\} \), where \( x_i \) is one data point with its label \( y_i \) in client \( k \) and \( n_k \) is the number of the local data points. The goal of local training in client \( k \) is to minimize

\[
F_k(w) = \frac{1}{n_k} \sum_{i=1}^{n_k} f_i(w) \tag{1}
\]

where \( f_i(w) \) is the loss function on one data point \( x_i \) with the label \( y_i \). Finally, the goal is to minimize aggregated local goals \( F_k(w) \) in (1):

\[
f(w) = \sum_{k=1}^{m} \frac{n_k}{n} F_k(w) \tag{2}
\]

i.e. \( f(w) = \mathbb{E}_{D_m}[F_k(w)] \). Note that the datasets across clients can be non-IID in federated learning.

However, most federated optimization methods exchange models or gradients for learning updates, which can still lead to excessive communication volumes when a model has numerous parameters. This is even more problematic in wireless networks common in many mobile applications, as frequently exchanging data leads to higher error likelihood when connections are unstable, which can cause federated learning to fail.

One-shot Federated Learning. Federated learning with one-shot communication has been studied in several projects [12]–[14]. Guha et al. [12] introduce an algorithm for training a support vector machine in one-shot fashion. The framework proposed by Kasturi et al. [13] uploads models and additionally the local dataset distribution, which hard to do when training on real datasets. Li et al. [14] utilize knowledge transfer to distill models in each client, which can change the original model structure and may cause much communication cost for sharing student models.

Instead of distilling models, our framework distills input data into smaller synthetic datasets in clients, which can be used for training a more general model by only relying on one-shot communication.

Dataset Distillation. Dataset distillation [15] has become an attractive paradigm. It attempts to synthesize a significantly smaller dataset from a large dataset, aiming to maintain the same training performance in terms of test accuracy. Dataset distillation was proposed by Wang et al. [15]. Methods like matching outputs or gradients [16], [17], [44], [45] have been achieving outstanding results. Beside updating synthetic datasets with forward and backward propagation, Nguyen et al. [18] perform Kernel Inducing Points (KIP) and Label Solve (LS) for the optimal solution. Relating back to our work, dataset distillation has been successfully applied in centralized training in limited fashion.

Dataset Distillation in Federated Learning. Though some previous works in dataset distillation, e.g. [44], have mentioned the dataset distillation might be beneficial for federated learning, they have not provided detailed analysis and experimental evaluation on it. Only little research has explored the dataset distillation approaches in federated learning: Zhou et al. [46] and Goetz et al. [47] have attempted to employ the approaches proposed by Wang et al. [15] and Sucholutsky and Schonlau [48] in federated learning, respectively. However, more advanced methods, e.g. [18] [16] with more stable performance have been developed, and further studies on decentralized dataset distillation in federated settings are needed. Furthermore, both of them have not pointed out the dataset distillation can improve the training on heterogeneous data, which is one of the biggest challenges in federated learning.

In fact, the computation abilities in distributed edge devices are normally limited, while most dataset distillation methods require high computation power. For instance, the approach from [16] can lead to computation overhead, though it can generate satisfactory distilled dataset. Therefore, in this work, we consider the coreset-based and KIP-based [18], [49] methods for decentralized dataset distillation in our federated learning framework, and focus on improving the communication efficiency while training on federated datasets.

III. FedD3: Federated Learning from Decentralized Distilled Datasets

To explore the dataset distillation in federated settings, we introduce FedD3, a federated learning framework from decentralized distilled datasets in this section. Specifically, we consider a joint learning task with \( m \) clients, where the client \( k \) owns local dataset \( D_k \). If we distill a synthetic dataset \( \tilde{D}_k (\tilde{n}_k = |\tilde{D}_k|) \) in the client \( k \) from its local dataset \( D_k (n_k = |D_k|) \), the goal of the dataset distillation instance is to

\[\text{Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.}\]
Algorithm 1 FedD3: Federated Learning from Decentralized Distilled Datasets

Server Input: number of the learning epochs $E$
Client Input: learning rate $\eta$
Client Input: number of the client $m$
Client Input: number of the distillation epochs $E_k$
Client Input: distillation rate $\eta_k$
Client Input: decentralized dataset $D_k$

Output: $\bar{w}$

1. **Server:**
   1. select instance with hyper-parameters $\Theta$
2. **for** $k \in \{1, 2, \ldots, m\}$ **in parallel**
3. $$\tilde{D}_k \leftarrow \text{ClientDatasetDistillation}(k, \Theta)$$
4. **end for**
5. $D \leftarrow \text{aggregate}(\tilde{D}_1, \tilde{D}_2, \ldots, \tilde{D}_m)$
6. **for** epoch $e = 1, 2, \ldots, E$ **do**
7. **for** each batch $\bar{B} = (\bar{X}_b, \bar{y}_b)$ of $\tilde{D}$ **do**
8. $\bar{w} \leftarrow \bar{w} - \eta (\nabla l(\bar{X}_b, \bar{y}_b; \bar{w}))$
9. **end for**
10. **end for**
11. return $\bar{w}$
12. **ClientDatasetDistillation($k, \Theta$)**
13. $\Theta_k \leftarrow \Theta$
14. initialize $\tilde{D}_k$ with a subset of $D_k$ and $\tilde{n}_k = |\tilde{D}_k|$
15. **for** epoch $e_k = 1, 2, \ldots, E_k$ **do**
16. **for** each pair of batches $\tilde{B}_k = (\tilde{X}_b, \tilde{y}_b)$ of $\tilde{D}_k$ and $B_k = (X_b, y_b)$ of $D_k$ **do**
17. $\tilde{X}_b \leftarrow \tilde{X}_b - \eta_k (\frac{\partial H_k(x, y; \Theta_k)}{\partial X_b})$
18. update $\tilde{B}_k$ and $\tilde{D}_k$ with $\tilde{X}_b$
19. **end for**
20. **if** converged **then**
21. break
22. **end if**
23. **end for**
24. return $\tilde{D}_k$

minimize $H_k(\tilde{X}_k, \tilde{y}_k; \Theta_k)$, where the $\tilde{X}_k$ represents the matrix of stacked distilled data points in the client $k$, $\tilde{x}_{k,i}$, $\tilde{y}_k$ contains the corresponding labels, and $\Theta_k$ indicates a set of parameters in the instance models. Note that $H_k$ can vary, depending on the instance used in the client $k$.

A. Core-set-based Methods

We start from core-set-based methods to distill the decentralized datasets. We assume that there exists a synthetic dataset $D_k$, which can approximate the statistical distribution of the original dataset $D_k$. Through minimizing the error in a small subset of $D_k$, we can generate the coreset $\tilde{D}_k$ using a specific instance, e.g. Kernel Herding [50]. More generally, we consider a clustering-based methods to generate a coreset in the client $k$ for one of classes $s \in S_k$ ($S_k$ is the set of local classes), then the goal is to minimize a clustering loss, for instance generating coreset using a Gaussian mixture model (GMM) [51] for each class $s$ in all clients.

B. KIP-based Methods

We review and adopt KIP [18], [49] to federated fashion for its fast divide up gradient computation. Each client $k$ aims to distill the original local dataset (a.k.a. target dataset) $D$ to a synthetic dataset (a.k.a. support dataset) $\bar{D}$ by minimizing the kernel ridge-regression (KRR) loss $H_k(\tilde{X}_k)$ as follows:

$$H_k(\tilde{X}_k) = \frac{1}{2} ||y_k - K_k(\tilde{X}_k, \tilde{X}_k)(K_k(\tilde{X}_k, \tilde{X}_k) + \lambda_k I)^{-1}y_k||^2_2$$ (3)

where $K_k$ is the kernel used in the client $k$ and $\lambda_k$ is the regularization constant deduced by ridge regression in KIP. $I$ is an identity matrix. We refer the readers to the work of Nguyen et al. [49] for further details regarding kernels.

C. Aggregation and Learning

After decentralized dataset distillation, the distilled datasets in all connected clients are transmitted to the server and aggregated as $\bar{D} = \{D_k|k = 1, 2, \ldots, m\}$. We consider a non-convex neural network objective in the server and train a machine learning model on $\bar{D}$ instead of on original dataset $D$. The objective is then minimizing:

$$f(w) = \frac{1}{m} \sum_{k=1}^{m} \sum_i f_i(w) = E_D[f_i(w)]$$ (4)

where $f_i(w) = l(\hat{x}_i, \hat{y}_i; w)$ is the loss of the prediction on one distilled data point $\hat{x}_i$, with its label $\hat{y}_i$ and model weights $w$. If the $\tilde{X}_k$ could be distilled from $X_k$ perfectly, the learning result of minimizing $f(w)$ on $\bar{D}$ should be similar to it on $D$. The FedD3 pseudocode is given in Algorithm 1.

D. Gamma Communication Efficiency

In federated learning, communication cost is even more expensive than computational cost [1]. It is worth studying how much communication cost is needed for achieving a dedicated gain of the model performance in terms of accuracy. To tackle the trade-off between model performance and required communication cost, we define the Gamma Communication Efficiency (GCE) using $\gamma$-accuracy gain per binary logarithmic bit as follows:

$$GCE = \frac{ACC}{(1 - ACC)\gamma * \sum_{t=1}^{T} \log_2(V_t + 1)}$$ (5)

where $T \in \mathbb{N}^+$ is the total communication rounds and $V_t \in \mathbb{R}^+$ is the required communication volume in each round.

We use the binary logarithmic bit $\sum_{t=1}^{T} \log_2(V_t + 1)$ to describe the communication cost from communication round $t = 1$ to $T$. Then the gain per binary logarithmic bit can be defined by $1/\sum_{t=1}^{T} \log_2(V_t + 1)$, where the 0 communication cost gives us infinitive high gain and infinitive high communication cost lead to 0 gain. $ACC$ is the prediction accuracy value. $\gamma \in \mathbb{R}^+$ is a tunable parameter to represent the importance of the prediction accuracy. If $\gamma \to 0$, the accuracy and communication cost in logarithmic bit is near proportional. The higher $\gamma$ is defined, the more test accuracy
is weighted. A tiny test accuracy gain can be scored very well with an infinitely high $\gamma$. Through selecting an appropriate $\gamma$, we can evaluate the performance of federated learning approaches, considering both test accuracy and communication cost based on the application scenarios.

IV. EXPERIMENT

A. Experimental Settings

We conduct experiments mainly on MNIST [52] and CIFAR-10 [53], as they are widely used for federated learning evaluation. For FedD3, we use GMM for coreset generation in coreset-based instance [51] and employ a four-layer fully connected neural network model with the width of 1024 as instance for KIP-based instance [18].

We compare FedD3 with eight other baselines, where four federated learning methods are evaluated in both multi-shot federated learning (MSFL) and one-shot federated learning (OSFL). The federated learning methods are: FedAvg [1], FedProx [54], FedNova [55] and SCAFFOLD [31]. The required binary logarithmic bit for communication is shown in Tab. I. We train an LeNet [56] and an AlexNet [57] on MNIST and CIFAR-10, respectively.

Considering properties in federated learning [1], we demonstrate our methods and baselines on IID and Non-IID datasets in distributed systems with a different number of clients. We vary the number of classes in each client $C_k$ to design Non-IID datasets [58]. Guided by federated learning benchmark datasets made by Caldas et al. [59], we use Non-IID to denote $C_k = 2$ (pathological Non-IID [19]), unless specified otherwise.

B. Robust Training on Heterogeneous Data

First, we focus on the performance of FedD3 on data heterogeneity and consider FedAvg as the baseline here. The accuracy changing with increasing total communication volume required in each method is observed in Fig. 2. In experiment with IID decentralized datasets, though FedAvg with unlimited communication cost has higher accuracy than FedD3, FedD3 outperforms FedAvg at the same communication volume. The best performance of one-shot FedAvg can be reached by using FedD3 with only around half of the communication volume. Note that we consider the communication volume only for uploading. In fact, MSFL methods still require further cost for downloading the global models.

Additionally, as shown in Fig. 2, the performance of FedD3 is obviously more robust than FedAvg, when the data heterogeneity across clients increases from left to right. With decreasing number of classes in local datasets, the prediction accuracy FedAvg reduces notably, however, the results of FedD3 are not much affected. In the scenario with extreme data heterogeneity, i.e. $C_k = 1$, the standard federated learning can even not easily converged and one-shot federated learning performances very badly, while FedD3 can achieve the similar accuracy as in IID scenarios. We believe the reason is that aggregating distilled datasets allows server to train a model on a similar distribution of original datasets.

C. Scalable Communication Efficiency

Then, we compare the results from 2 variants of FedD3 with 8 baselines on both IID and Non-IID, MNIST and CIFAR10 distributed in 500 clients. As shown in Tab. II, on IID datasets, FedD3 can achieve a comparable test accuracy to other federated learning, while the GCE is significantly higher. On Non-IID datasets, FedD3 is outperforms others for both test accuracy and GCE. We consider various $\gamma$ to indicate the evaluation based on different importance of accuracy. We assign $\gamma$ for CIFAR-10 with greater values than for MNIST, as a high accuracy for training model on CIFAR-10 is harder to achieve and should deserve to spend more communication cost for it.

We conduct further experiments for training a ResNet-18 [60] on Fashion-MNIST [61] and SVHN [62], and training a CNN consisting of 5 convolutional and 3 fully connected layers on CIFAR-100 [62], using MSFL and OSFL with FedAvg, and FedD3 with KIP-instances. As shown in Tab. III, we observe consistent results.

To perform the scalability of communication efficiency, we meanwhile evaluate FedD3 with increasing number of images per class in each client. Fig. 3 shows the test accuracy raises and thereby GCE with higher $\gamma$ increases, when each client contributes more distilled images to the server. However, GCE with $\gamma = 0.01$ reduces, because much communication cost is required for only a small accuracy gain. FedD3 provides the opportunity to optimize the GCE by adjusting the Img/Cls in decentralized dataset distillation, considering different importance of accuracy and constraint communication budget in practical applications.

D. Evaluation with System Parameters

Finally, we explore the performance of FedD3 affected by federated system parameters, including the number of clients and distilled images. In Fig. 4, we run FedD3 in federated systems containing various number of clients with IID and Non-IID decentralized datasets. We hold the total number of distilled images constant by varying the number of distilled images in each client. Fig. 4 shows that the prediction accuracy decreases at a larger number of clients. This can be led by the following reason: When increasing the client number, both the local data volume and distilled image per client reduce in our experimental setup. This results in lower granularity and thereby decreases prediction accuracy.

In fact, federated learning considers massively distributed systems in practical applications. Thus, even if each client

| MODELS   | FedAvg | FedProx | FedNova | SCAFFOLD |
|----------|--------|---------|---------|----------|
| log$_2$(P + 1) | log$_2$(P + 1) | log$_2$(P + 9) | log$_2$(2*P + 1) |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
provides a small number of distilled images, a promising training performance can be achieved, when the number of clients is large. As we can observe in Fig. 4, when each client provides the same number of distilled images, a better model can be trained with more clients, due to more distilled images received in the server. Moreover, additional clients in real applications can enrich the training dataset, and hence reach a better prediction accuracy, which is consistent with the motivation of deploying federated learning.

V. DISCUSSION

Distilled Datasets in Multi-round Due to the robustness on data heterogeneity, we further explore the potential benefits of distilled datasets in federated learning. We believe that sharing such synthetic data might bridge the information silos. For that, we extend FedD3 to multiple shots and consider a hybrid
federated learning method by adding a spoon of distilled datasets from other clients via D2N (Device to Networks) or D2D (Device to Device) networks, before the first round in standard federated learning.

**Network Assumption** Compared to multi-shot schemes, a one-shot scheme is less affected by network heterogeneity. Despite, to evaluate the impact of network heterogeneity and address the network assumption in federated learning application scenarios, we consider the Quality of Service (QoS) of **at most once** for the one-shot scheme and compare the performance of FedD3 and OSFL with different ratios of stragglers. As shown in Fig 5, FedD3 outperforms OSFL.

**Computation Costs** Most dataset distillation methods can result in high computation loads. In our experiment setup, we take into account computation limitations in KIP and scale the number of distilled images on each client. In our FedD3 setup, we use FC-4 kernel without pooling and convolutional kernels for dataset distillation. In each client, we only distill a small number of images, i.e. 1 Img/Cls. In Federated Learning, local training can result in a linear increase in computational cost with the number of epochs. We also disregard the computational cost caused by a large number of epochs and aim to achieve better results in the baselines for a fair comparison only for communication efficiency. Additionally, we measure the required time on a compact NVIDIA Quadro RTX 4000 GPU for training an AlexNet on CIFAR-10: On average 73.69s are needed for FedD3 with 800 distillation steps for 1 Img/Cls, and 89.96s for one-shot FedAvg with 10 epochs.

**Beyond Kernel: Individual DD-instances** Because of stable performance, we mainly employ KIP methods as dataset distillation instances in FedD3. In fact, according to the actual local datasets, we can also consider individual instances to generate the synthetic dataset for uploading. Self-selecting instances in each client should be encouraged, which can be advantageous to the quality of distilled data, as the server and other clients should be not aware of the distribution of local data. Additionally, we believe the autonomy of local dataset distillation can enhance privacy.

**Lessons Learned for Orchestration** In FedD3, all local datasets should meet the requirement of the used dataset distillation method. Here we can think about two example scenarios: (i) If there is only one data point in one of the clients, $D_k$ and $D_k$ are the same at the first epoch in ClientDatasetDistillation in Algorithm 1; (ii) If there is only one data point with its label $x_i$ for a specific class $y_i$, the distilled dataset $\tilde{x}_i$ can be always the same as $x_i$, especially when only the loss is backpropagated on $\tilde{x}$. Both situations can break the privacy, as it will upload at least one raw data point to the server. To tackle this issue, we believe a good orchestration for dataset and client selection is recommended.

**VI. CONCLUSION**

In this work, we introduce a novel federated learning framework, FedD3, which reduces the overall communication volume and with that opens up the concept of federated learning to more application scenarios in particular in network constrained environments. It achieves this by leveraging local dataset distillation instead of traditional learning approaches (i) to significantly reduce communication volumes and (ii) to limit transfers to one-shot communication, rather than iterative multi-way communication. Our conducted experimental evaluation shows that FedD3 can well balance the trade-off between prediction accuracy and communication cost for federated learning. Compared to other federated learning, it provides a more robust training performance, especially on Non-IID data silos.

**REFERENCES**

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized
Fig. 4. Number of clients effects the test accuracy (in %) of model. We consider test accuracy changing with the different number of clients. As it is naturally clear that the test accuracy of the final trained model can be improved by the increasing number of the final distilled images number at the server, through varying the number of the distilled images at each client with respect to the number of clients, we keep the size of the global distilled dataset always the same across the experiments, and hence eliminate the effects from it. Note that for Non-IID dataset, if the number of clients is less than 10 (as shown in the gray area), not all classes in the entire dataset can be covered, therefore, the test accuracy reduces. Apart from this impact, the results show that (i) the test accuracy increases with smaller client numbers; (ii) the test accuracy on Non-IID datasets and on IID datasets are similar, which indicates the FedD3 is robust to data heterogeneity.

Fig. 5. Performance comparison of FedD3 and OSFL for training a CNN model on distributed Non-IID CIFAR-100 in heterogeneous networks (with different ratios of stragglers).

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
[51] J.-P. Baudry, A. E. Raftery, G. Celeux, K. Lo, and R. Gottardo, “Combining mixture components for clustering,” Journal of computational and graphical statistics, vol. 19, no. 2, pp. 332–353, 2010.

[52] Y. LeCun, C. Cortes, and C. Burges, “Mnist handwritten digit database,” ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist, vol. 2, 2010.

[53] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Technical report, 2009.

[54] 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.

[55] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” Advances in neural information processing systems, vol. 33, pp. 7611–7623, 2020.

[56] 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, doi:10.1109/5.726791.

[57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/c999862d389d6d7e48436c924a6bc43b-Paper.pdf

[58] Q. Li, Y. Diao, Q. Chen, and B. He, “Federated learning on non-iid data silos: An experimental study,” in IEEE International Conference on Data Engineering, 2022.

[59] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, “Leaf: A benchmark for federated settings,” arXiv preprint arXiv:1812.01097, 2018.

[60] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[61] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” arXiv preprint arXiv:1708.07747, 2017.

[62] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” 2011.