FedDC: Federated Learning from Small Datasets

Michael Kamp
Institute for AI in medicine (IKIM)
and Ruhr-University Bochum and Monash University
University Hospital Essen
Essen, Germany
michael.kamp@uk-essen.de

Jonash Fischer
Max Planck Institute for Informatics
Saarbrücken, Germany
fischer@mpi-inf.mpg.de

Jilles Vreeken
CISPA Helmholtz Center for Information Security
Saarbrücken, Germany
Jilles Vreeken <vreeken@cispa.de>

Abstract

Federated learning allows multiple parties to collaboratively train a joint model without having to share any local data. This enables applications of machine learning in settings where data is inherently distributed and undisclosable, such as in the medical domain. Joint training is usually achieved by aggregating local models. When local datasets are small, though, locally trained models can vary greatly from a globally good model. Bad local models can arbitrarily deteriorate the aggregate model quality, causing federating learning to fail. We propose a novel approach that avoids this problem by interleaving model *aggregation* and *permutation* steps. During a permutation step we redistribute local models across clients through a coordinator, while preserving data privacy. This exposes each local model to a daisy chain of local datasets, which enables successful training in data-sparse domains. Combined with aggregation steps we achieve effective learning even if the local datasets are extremely small, while retaining the privacy benefits of federated learning.

1 Introduction

How can we learn *high quality* models when data is *inherently distributed* across sites and cannot be shared or pooled? In federated learning, the solution is to iteratively train models locally at each site and share these models with a coordinator to be aggregated to a global model. As only models are shared, data usually remains undisclosed. This process, however, requires sufficient data to be available at each site in order for the locally trained models to achieve a minimum quality—even a single bad model can render aggregation arbitrarily bad [39]. In many relevant applications this requirement is not met: In healthcare settings we often have as little as a few dozens of samples [9, 30, 41]. Also in domains where deep learning is generally regarded as highly successful, such as natural language processing and object detection, applications often suffer from a lack of data [17, 25].

To tackle this problem, we propose a new building block for federated learning in which models traverse clients in the form of a daisy chain. In a nutshell, at each client a model is trained locally, sent to the coordinator, and then—instead of aggregating local models—sent to a random other client as is (see Fig. 1). This way, each local model is exposed to a daisy-chain of clients and their local datasets. This allows us to learn from small, distributed datasets simply by consecutively training the model with the data available at each site. We should not do this naively, however, since it

Preprint. Under review.

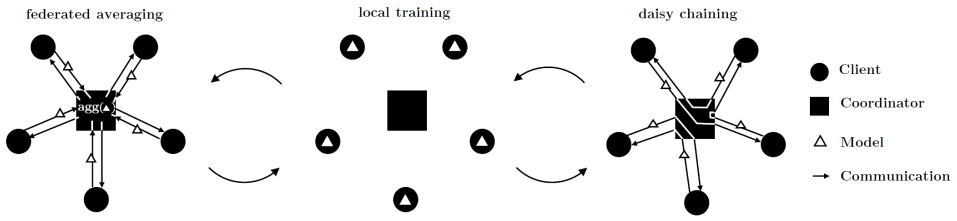


Figure 1: *Federated learning settings.* A standard federated learning setting with training of local models at clients (middle) with aggregation phases where models are communicated to the coordinator, aggregated, and sent back to each client (left). We propose to add daisy chaining (right), where local models are sent to the coordinator and then redistributed to a random permutation of clients as is.

would lead to overfitting—a common problem in federated learning which can cause learning to diverge [10]. It further violates privacy, since a client can infer from a model upon the data of the client it received it from [40]. To alleviate these issues, we propose to combine daisy-chaining of local datasets with aggregation of models, both orchestrated by a coordinator, and term this method *federated daisy-chaining* (FEDDC).

We show that our approach maintains privacy of local datasets, while it provably converges, and guarantees improvement of model quality in convex problems with a suitable aggregation method. For that it requires just slightly more communication than standard federated learning for the additional daisy-chaining rounds. The additional communication is, however, negligible in the cross-SILO scenarios we consider (e.g., healthcare) since communication is not a bottleneck there. Moreover, we show that even with the same amount of communication as FEDDC, current federated learning techniques fail on the considered tasks. Formally, we show convergence for FEDDC on non-convex problems. We then show for convex problems that FEDDC succeeds on small datasets where standard federated learning fails. For that, we analyze FEDDC combined with aggregation via the Radon point from a PAC-learning perspective. We substantiate this theoretical analysis for convex problems by showing that FEDDC in practice matches the accuracy of a model trained on the full data of the SUSY binary classification dataset with only 2 samples per client, beating standard federated learning by a wide margin. In fact, FEDDC allows us to match the model quality of centralized training with only 2 samples per client. For non-convex settings, we provide an extensive empirical evaluation, showing that FEDDC outperforms naive daisy-chaining, vanilla federated learning FEDAVG [29], FEDPROX [23], FEDADAGRAD, FEDADAM, and FEDYOGI [35] on low-sample CIFAR10 [19], including non-iid settings, and, more importantly, on two real-world medical imaging datasets. Not only does FEDDC provide a wide margin of improvement over existing federated methods, but it is also on par with a (centralized) neural network of the same architecture trained on the pooled data.

In summary, our contributions are (i) FEDDC, a novel approach to federated learning from small datasets via a combination of model permutations across clients and aggregation, (ii) a formal proof of convergence for FEDDC, (iii) a theoretical guarantee that FEDDC improves models in terms of ϵ, δ-guarantees which standard federated learning can not, (iv) a discussion of the privacy aspects and mitigations suitable for FEDDC, including an empirical evaluation of differentially private FEDDC, and (v) an extensive set of experiments showing that FEDDC substantially improves model quality for small datasets, e.g., successfully training a ResNet18 on a pneumonia dataset with only 8 samples per client.

2 Related Work

Learning from small datasets is a well studied problem in machine learning. In the literature, we find both general solutions, such as using simpler models and transfer learning [43], and more specialized ones, such as data augmentation [12] and few-shot learning [33, 45]. In our scenario data is abundant, but the problem is that the local datasets at each site are small and cannot be pooled. Yurochkin et al. [51] propose local data augmentation for federated learning, but their method requires a sufficient quality of the local model for augmentation which is the opposite of the scenario we are considering. [11] provide generalization bounds for federated averaging via the NTK-framework, but requires one-layer infinite-width neural networks and infinitesimal learning rates.

Federated learning and its variants have been shown to learn from incomplete local data sources, e.g., non-iid label distributions [23, 47] and differing feature distributions [24, 36], but fail in case of large gradient diversity [10] and strongly dissimilar label distribution [28]. For small datasets, local empirical distributions may vary greatly from the global distribution—while the difference of empirical to true distribution decreases exponentially with the sample size (e.g., according to the Dvoretzky–Kiefer–Wolfowitz inequality), for small samples the difference can be substantial, in particular if the distribution differs from a Normal distribution [20]. Shamir and Srebro [39] have shown the adverse effect of bad local models on averaging, proving that even due to a single bad model averaging can be arbitrarily bad.

A different approach to dealing with biased local data is by learning personalized models at each client. Such personalized FL [1, 22] can reduce sample complexity, e.g., by using shared representations [6] for client-specific models, e.g., in the medical domain [49]. It is not applicable, however, to settings where you do not want to learn the biases or batch effects of local clients (e.g., in many medical applications), since by design it cannot train a global model. Kiss and Horvath [18] propose a decentralized and communication-efficient variant of federated learning that migrates models over a decentralized network, storing incoming models locally at each client until sufficiently many models are collected on each client for an averaging step, similar to Gossip federated learing [13]. The variant without averaging is similar to simple daisy-chaining which we compare to in Section 7. FEDDC is compatible with any aggregation operator, including the Radon machine [15] and the geometric median [32]. It can also be straightforwardly combined with approaches to improve communication-efficiency, such as dynamic averaging [16], and model quantization [37]. We provide empirical results for FEDDC in combination with standard averaging, the Radon machine, and FedProx [23] and compare against classical federated averaging (FEDAVG) [29], FEDPROX [23], FEDADAGRAD, FEDYOGI, and FEDADAM [35] in an extensive study.

3 Preliminaries

We assume iterative learning algorithms [cf. Chp. 2.1.4 14] $\mathcal{A} : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathcal{H}$ that update a model $h \in \mathcal{H}$ using a dataset $D \subset \mathcal{X} \times \mathcal{Y}$ from an input space $\mathcal{X}$ and output space $\mathcal{Y}$, i.e., $h_{t+1} = \mathcal{A}(D, h_t)$. Given a set of $m \in \mathbb{N}$ clients with local datasets $D^1, \ldots, D^m \subset \mathcal{X} \times \mathcal{Y}$ drawn iid from a data distribution $\mathcal{D}$ and a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, the goal is to find a single model $h^* \in \mathcal{H}$ that minimizes the risk

$$\varepsilon(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell(h(x), y) \right] \ . \tag{1}$$

In *centralized learning*, the datasets are pooled as $D = \bigcup_{i\in[m]} D^i$ and $\mathcal{A}$ is applied to D until convergence. Note that applying $\mathcal{A}$ on D can be the application to any random subset, e.g., as in mini-batch training, and convergence is measured in terms of low training loss, small gradient, or small deviation from previous iterate. In standard *federated learning* [29], $\mathcal{A}$ is applied in parallel for $b \in \mathbb{N}$ rounds on each client locally to produce local models $h^1, \ldots, h^m$. These models are then centralized and aggregated using an aggregation operator $\mathrm{agg} : \mathcal{H}^m \to \mathcal{H}$, i.e., $\overline{h} = \mathrm{agg}(h^1, \ldots, h^m)$. The aggregated model $\overline{h}$ is then redistributed to local clients which perform another b rounds of training using $\overline{h}$ as a starting point. This is iterated until convergence of $\overline{h}$. When aggregating by averaging, this method is known as federated averaging (FEDAVG). Next, we describe FEDDC.

4 Federated Daisy-Chaining

We propose federated daisy chaining as an extension to federated learning and hence assume a setup where we have m clients and one designated coordinator node.[1] We provide the pseudocode of our approach as Algorithm 1.

The client: Each client trains its local model in each round on local data (line 4), and sends its model to the coordinator every b rounds for aggregation, where b is the aggregation period, and every d rounds for daisy chaining, where d is the daisy-chaining period (line 6). This re-distribution of models results in each individual model following a daisy-chain of clients, training on each local dataset. Such a daisy-chain is interrupted by each aggregation round.

[1] This star-topology can be extended to hierarchical networks in a straight-forward manner. Federated learning can also be performed in a decentralized network via gossip algorithms [13].

Algorithm 1: Federated Daisy-Chaining FEDDC

Input: daisy-chaining period d, aggregation period b, learning algorithm $\mathcal{A}$, aggregation operator agg, m clients with local datasets $D^1, \ldots, D^m$, total number of rounds T
Output: final model aggregate $\overline{h}_T$

1 initialize local models $h_0^1, \ldots, h_0^m$
2 **Locally** *at client i at time t* **do**
3 sample S from D_i
4 $h_t^i \leftarrow \mathcal{A}(S, h_{t-1}^i)$
5 **if** $t \,\%\, d = d-1$ *or* $t \,\%\, b = b-1$ **then**
6 send h_t^i to coordinator
7 receive new h_t^i from coordinator // either aggregate $\overline{h}_t$ or some h_t^j
8 **end**
9 **At coordinator** *at time t* **do**
10 **if** $t \,\%\, b = b-1$ **then** // aggregation
11 $\overline{h}_t \leftarrow \mathrm{agg}(h_t^1, \ldots, h_t^m)$ send $\overline{h}_t$ to all clients
12 **end**
13 **else if** $t \,\%\, d = d-1$ **then** // daisy chaining
14 draw permutation π of [1,m] at random
15 for all $i \in [m]$ send model h_t^i to client $\pi(i)$
16 **end**

The coordinator: Upon receiving models (line 10), in a daisy-chaining round (line 11) the coordinator draws a random permutation π of clients (line 12) and re-distributes the model of client i to client $\pi(i)$ (line 13), while in an aggregation round (line 15), the coordinator instead aggregates all local models (line 16) and re-distributes the aggregate to all clients (line 17).

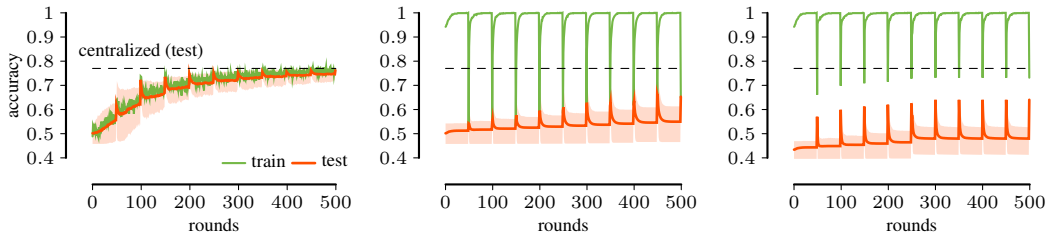
Communication complexity: Note that we consider cross-SILO settings, such as healthcare, were communication is not a bottleneck and, hence, restrict ourselves to a brief discussion in the interest of space. Communication between clients and coordinator happens in $O(\frac{T}{d} + \frac{T}{b})$ many rounds, where T is the overall number of rounds. In other words, FEDDC communicates in every dth and bth round out of the T overall rounds. Therefore, the amount of communication rounds is similar to FEDAVG with averaging period $b_{FedAvg} = \min\{d, b\}$. That is, FEDDC increases communication over FEDAVG by a constant factor depending on the setting of b and d. The amount of communication per communication round is linear in the number of clients and model size, similar to federated averaging. In each communication round of FEDDC, each local client sends its parameters securely to the coordinator and receives a set of parameters, regardless of the type of communication round; just the internal operation of the coordinator is different, which does not affect communication. Furthermore, in our experiments, we show that FEDDC outperforms FEDAVG with the same amount of communication, as well as FEDAVG with an optimal setting of its aggregation period.

5 Theoretical guarantees

In this section, we formally show that FEDDC converges for averaging. We, further, provide theoretical bounds on the model quality in convex settings, showing that FEDDC has favorable generalization error in low sample settings compared to standard federated learning. More formally, we first show that under standard assumptions on the empirical risk, it follows from a result of Yu et al. [50] that FEDDC converges when using averaging as aggregation and SGD for learning—a standard setting in, e.g., federated learning of neural networks. We provide all proofs in the appendix.

Corollary 1. *Let the empirical risks $\mathcal{E}_{emp}^i(h) = \sum_{(x,y)\in D^i} \ell(h_i(x), y)$ at each client $i \in [m]$ be L-smooth with σ^2-bounded gradient variance and G^2-bounded second moments, then* FEDDC *with averaging and SGD as learning algorithm has a convergence rate of $\mathcal{O}(1/\sqrt{mT})$, where $T \in \mathbb{N}$ is the number of local updates.*

Since model quality in terms of generalization error does not necessarily depend on convergence of training [10, 16], we additionally analyze model quality in terms of probabilistic worst-case guarantees



(a) FEDDC with Radon point with $d = 1, b = 50$.

(b) Federated learning with Radon point with $b = 50$.

(c) Federated learning with averaging with $b = 50$.

Figure 2: *Results on SUSY.* We visualize results in terms of train (green) and test error (orange) for FEDDC (a) and standard federated learning (b), both using Radon points for aggregation, and FEDAVG (c). The network has 441 clients with 2 data points per client. The performance of a central model trained on all data is indicated by the dashed line.

on the generalization error [38]. The average of local models can yield as bad a generalization error as the worst local model, and, hence, using averaging as aggregation scheme in standard federated learning can yield arbitrarily bad results [cf. 39]. As the probability of bad local models starkly increases with smaller sample sizes, this trivial bound, hence, often carries over to our considered practical settings. The Radon machine [15] is a federated learning approach that overcomes these issues for a wide range of learning algorithms and allows us to analyze (non-trivial) quality bounds of aggregated models under the assumption of convexity. Next, we show that FEDDC can improve model quality for small local datasets where standard federated learning fails to do so.

A Radon point [34] of a set of points S from a space $\mathcal{X}$ is – similar to the geometric median – a point in the convex hull of S with a high centrality (more precisely, a Tukey depth [8, 44] of at least 2). For a Radon point to exist, the size of S has to be sufficiently large; the minimum size of $S \subset \mathcal{X}$ is denoted the Radon number of the space $\mathcal{X}$ and for $\mathcal{X} \subseteq \mathbb{R}^d$ the radon number is $d + 2$. Here, the set of points S are the local models, or more precisely their parameter vectors. We make the following standard assumption [46] on the local learning algorithm $\mathcal{A}$.

Assumption 2 ((ϵ, δ)-guarantees)**.** *The learning algorithm $\mathcal{A}$ applied on a dataset drawn iid from $\mathcal{D}$ of size $n \geq n_0 \in \mathbb{N}$ produces a model $h \in \mathcal{H}$ s.t. with probability $\delta \in (0, 1]$ it holds for $\epsilon > 0$ that $\mathbb{P}(\varepsilon(h) > \epsilon) < \delta$. The sample size n_0 is monotonically decreasing in δ and ϵ (note that typically n_0 is a polynomial in ϵ^{-1} and $\log(\delta^{-1})$).*

Here $\varepsilon(h)$ is the risk defined in Equation 1. Now let $r \in \mathbb{N}$ be the Radon number of $\mathcal{H}$, $\mathcal{A}$ be a learning algorithm as in assumption 2, and risk ε be convex. Assume $m \geq r^h$ many clients with $h \in \mathbb{N}$. For $\epsilon > 0, \delta \in (0, 1]$ assume local datasets $D_1, \ldots, D_m$ of size larger than $n_0(\epsilon, \delta)$ drawn iid from $\mathcal{D}$, and $h_1, \ldots, h_m$ be local models trained on them using $\mathcal{A}$. Let $\mathfrak{r}_h$ be the iterated Radon point with h iterations computed on the local models. Then it follows from Theorem 3 in [15] that for all $i \in [m]$ it holds that

$$\mathbb{P}(\varepsilon(\mathfrak{r}_h) > \epsilon) \leq (r\,\mathbb{P}(\varepsilon(h_i) > \epsilon))^{2^h} \tag{2}$$

where the probability is over the random draws of local datasets. That is, the probability that the aggregate $\mathfrak{r}_h$ is bad is doubly-exponentially smaller than the probability that a local model is bad. Note that in PAC-learning, the error bound and the probability of the bound to hold are typically linked, so that improving one can be translated to improving the other [46]. Eq. 2 implies that the iterated Radon point only improves the guarantee on the confidence compared to that for local models if $\delta < r^{-1}$, i.e. $\mathbb{P}(\varepsilon(\mathfrak{r}_h) > \epsilon) \leq (r\,\mathbb{P}(\varepsilon(h_i) > \epsilon))^{2^h} < (r\delta)^{2^h} < 1$ only holds for $r\delta < 1$. Consequently, local models need to achieve a minimum quality for the federated learning system to improve model quality.

Corollary 3. *Let $\mathcal{H}$ be a model space with Radon number $r \in \mathbb{N}$, ε a convex risk, and $\mathcal{A}$ a learning algorithm with sample size $n_0(\epsilon, \delta)$. Given $\epsilon > 0$ and any $h \in \mathbb{N}$, if local datasets $D_1, \ldots, D_m$ with $m \geq r^h$ are smaller than $n_0(\epsilon, r^{-1})$, then federated learning using the Radon point does not improve model quality in terms of (ϵ, δ)-guarantees.*

In other words, when using aggregation by Radon points alone, an improvement in terms of (ϵ, δ)-guarantees is strongly dependent on large enough local datasets. Furthermore, given $\delta > r^{-1}$, the guarantee can become arbitrarily bad by increasing the number of aggregation rounds.

Federated Daisy-Chaining as given in Alg. 1 permutes local models at random, which is in theory equivalent to permuting local datasets. Since the permutation is drawn at random, the amount of permutation rounds T necessary for each model to observe a minimum number of distinct datasets k can be given with high probability via a variation of the coupon collector problem.

Lemma 4. *Given $\delta \in (0, 1]$, $m \in \mathbb{N}$ clients, and $k \in [m]$, if Algorithm 1 with daisy chaining period $d \in \mathbb{N}$ is run for $T \in \mathbb{N}$ rounds with*

$$T \geq d \frac{\ln \delta}{\ln\left(\frac{m-1}{m}\right)(m-k+1)m}$$

then each local model has seen at least k distinct datasets with probability $1 - \gamma$.

From Lem. 4 it follows that when we perform daisy-chaining with m clients, and local datasets of size n, for at least $d \ln \gamma((\ln(m-1) - ln(m))(m-k+1)m)^{-1}$ rounds, each local model will with probability at least $1 - \gamma$ be trained on at least kn distinct samples. For an ϵ, δ-guarantee, we thus need to set b large enough so that $kn \geq n_0(\epsilon, \sqrt{\delta})$ with probability at least $1 - \sqrt{\delta}$. This way, the failure probability is the product of not all clients observing k distinct datasets and the model having a risk larger than ϵ, which is $\sqrt{\delta}\sqrt{\delta} = \delta$.

Proposition 5. *Let $\mathcal{H}$ be a model space with Radon number $r \in \mathbb{N}$, ε a convex risk , and $\mathcal{A}$ a learning algorithm with sample size $n_0(\epsilon, \delta)$. Given $\epsilon > 0$, $\delta \in (0, r^{-1})$ and any $h \in \mathbb{N}$, and local datasets $D_1, \ldots, D_m$ of size $n \in \mathbb{N}$ with $m \geq r^h$, then Alg. 1 using the Radon point with aggr. period*

$$b \geq d \frac{\ln \sqrt{\delta}}{\ln\left(\frac{m-1}{m}\right)\left(m - \left\lceil \frac{n_0(\epsilon, \sqrt{\delta})}{n} \right\rceil + 1\right)m} \tag{3}$$

improves model quality in terms of (ϵ, δ)-guarantees.

Prop. 5 implies that if enough daisy-chaining rounds are performed in-between aggregation rounds (given by Eq. 3), then federated learning via the iterated Radon point [5] is guaranteed to improve model quality in terms of (ϵ, δ)-guarantees— the resulting model has generalization error smaller than ϵ with probability at least $1 - \delta$.

In Fig. 2 we show that this theoretical result holds in practice as well. For that, we compare FEDDC using the iterated Radon point with standard federated learning on the SUSY binary classification dataset [3], training a linear model on 441 clients with only 2 samples per client. After 500 rounds FEDDC reached the test accuracy of a model that has been trained on the centralized dataset (ACC=0.77) beating federated learning by a large margin (ACC=0.65). In Sec. 7 we show that the empirical results for averaging are similar to those for the Radon machine. Before that, we discuss the privacy-aspects of FEDDC in the following section.

6 Data Privacy

A major advantage of federated over centralized learning is that local data remains undisclosed to anyone but the local client, only model parameters are exchanged. This provides a natural benefit to data privacy, which is the main concern in applications such as healthcare. In federated learning, the main threat to data privacy are inference attacks: An attacker can make inferences about local data from model parameters [27] and model updates [53], which requires infiltrating the coordinator or communication channels to intercept a model. In the daisy-chaining rounds

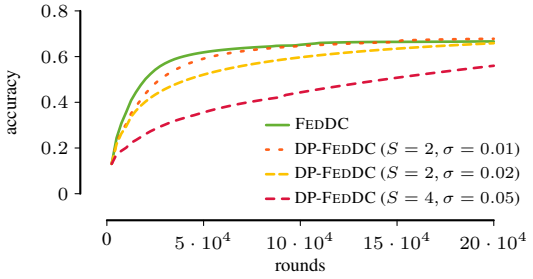


Figure 3: *Differential privacy results.* We show the performance of FEDDC (top solid line) compared to runs with clipped parameter updates and Gaussian noise (dashed lines) on CIFAR10 with 250 clients.

of FEDDC clients receive a model that was directly trained on the local data of another client, instead of a model aggregate. This facilitates membership inference attacks [40] of the type where the attacker tries to infer whether a known record is present in that training set. As this requires prior knowledge of a data point, e.g., a patient record, and the site the model originates from is unknown, these attacks are of limited interest compared to those where the goal is to infer unknown records from the model parameters. Doing the latter requires obtaining model updates [53] which are not available to the attacker, because the coordinator randomly permutes the order of clients within a daisy-chain round. Hence, the malicious client cannot link models to source clients, and can neither infer the location of a patient record, nor model updates.

Should a malicious client obtain model updates through additional attacks, a common defense is applying appropriate clipping and noise before sending models. This guarantees ϵ, δ-differential privacy for local data [48] at the cost of a slight-to-moderate loss in model quality. This technique is also proven to defend against backdoor and poisoning attacks [42]. Moreover, FEDDC is compatible with standard defenses against such attacks, such as noisy or robust aggregation [26]—FEDDC with the Radon machine is an example of such a robust aggregation. We illustrate the effectiveness of FEDDC with this differential privacy technique in the following experiment. We train a small ResNet on 250 clients using FEDDC with $d = 2$ and $b = 10$, postponing the details on the experimental setup to App. A.9.1 and A.9.2. Differential privacy is achieved by clipping local model updates and adding Gaussian noise as proposed by [7]. The results as shown in Figure 3 indicate that the standard trade-off between model quality and privacy holds for FEDDC as well. Moreover, for mild privacy settings the model quality does not decrease. That is, FEDDC is able to robustly predict even under differential privacy. We provide extended discussion on the privacy aspects of FEDDC in App. A.8.

7 Experiments on Deep Learning

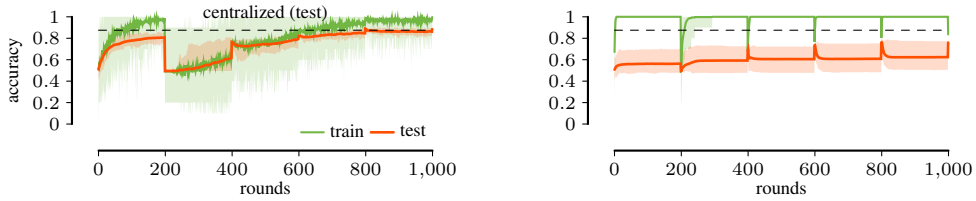
Our approach FEDDC, both provably and empirically, improves model quality when using Radon points as aggregation which, however, require convex problems. Many practical applications are so far only solved by non-convex approaches, as is the case for example in many imaging problems, where convolutional neural networks (CNNs) are among the state of the art. We, hence, evaluate FEDDC with averaging against the state of the art in federated learning on synthetic and real world data using neural networks. As baselines, we consider standard Federated averaging (FEDAVG) [29], FEDAVG with equal communication as FEDDC, and simple daisy-chaining without aggregation. We further consider the 4 state-of-the-art methods FEDPROX [23], FEDADAGRAD, FEDYOGI, and FEDADAM [35]. As datasets we consider a synthetic classification dataset as a simple test for vanilla neural networks, the image classification problem CIFAR10 [19], and two real-world applications: (i) publicly available MRI scans for brain tumors[2], and (ii) chest X-rays for pneumonia[3] (e.g., from COVID-19). For reproducibility, we provide details on architectures and experimental setup in App. A.9.1,A.9.2. The implementation of the experiments is publicly available at `https://anonymous.4open.science/r/FedDC-19E0`.

Synthetic Data: We first investigate the potential of FEDDC on a synthetic binary classification dataset generated by the sklearn [31] `make_classification` function with 100 features. On this dataset, we train a simple fully connected neural network with 3 hidden layers on $m = 50$ clients with $n = 10$ samples per client. We compare FEDDC with daisy-chaining period $d = 1$ and aggregation period $b = 200$ to FEDAVG with $b = 200$. The results presented in Fig. 4 show that FEDDC achieves a test performance of 0.89. This is comparable to centralized training on all data which achieves a test accuracy of 0.88. It hence substantially outperforms FEDAVG. The results indicate that the main issue of FEDAVG is overfitting of local clients, since for FEDAVG *train* accuracy reaches 1.0 quickly after each averaging step. With these promising results on vanilla neural networks, we next turn to real-world image classification problems typically solved with CNNs.

CIFAR10: As a first challenge for image classification, we consider the well-known CIFAR10 image benchmark. We first investigate the effect of the aggregation period b on FEDDC and FEDAVG,

[2] kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection

[3] kaggle.com/praveengovi/coronahack-chest-xraydataset



(a) FEDDC with $d = 1$, $b = 200$.

(b) FEDAVG with $b = 200$.

Figure 4: *Synthetic data results.* Comparison of FEDDC (left) and FEDAVG (right) for training fully connected neural networks on a synthetic dataset. Mean and confidence test accuracy of each client reported in orange, the test accuracy of centralized training is indicated by the dashed black line.

using a setting of 250 clients with a small version of ResNet, and 64 local samples each, drawn at random without replacement (details in App. A.9.2). We report the results in Figure 5 and set the period for FEDDC to $b = 10$, and consider federated averaging with periods of both $b = 1$ (equivalent communication to FEDDC with $d = 1, b = 10$) and $b = 10$ (less communication than FEDDC by a factor of 10) for all subsequent experiments.

Next, we consider a subset of 9600 samples spread across 150 clients (i.e. 64 samples per client), which corresponds to our small sample setting. Each client is equipped with an untrained ResNet18.[4] Note that the combined amount of examples is only one fifth of the original training data, hence we cannot expect the typical performance on this dataset. Results are reported in Table 1. We observe that FEDDC achieves substantially higher accuracy of more than 6 percentage points over the baseline set by federated averaging with the same amount of communication. Upon further inspection of the results, we see that FEDAVG drastically overfits, achieving training accuracies of 0.97 (see App. A.1), a similar trend as reported in Figure 4 for synthetic data. We also see that daisy-chaining alone, besides its privacy issues, performs worse than FEDDC. Intriguingly, also the state of the art shows similar performance trends. FedProx, run with $b = 10$ and $\mu = 0.1$, only achieves an accuracy of 0.511. The methods FEDADAGRAD, FEDYOGI, and FEDADAM with global learning rate $\eta = 1.0$ show even worse performance of accuracies around 0.22, 0.31, and 0.34, respectively. While applied successfully on large-scale data, these methods seem to have shortcomings when it comes to small sample regimes. To investigate the extent of what is possible on this subset of CIFAR10, we run centralized learning, which results in an accuracy of around 0.61. Interestingly, FEDDC is therefore slightly better even than the centralized version. We, however, do not want to over-interpret this difference, as the variance of the performance of the centralized model for the other datasets is rather high and hence could be—if repeated often enough—be on par with FEDDC.

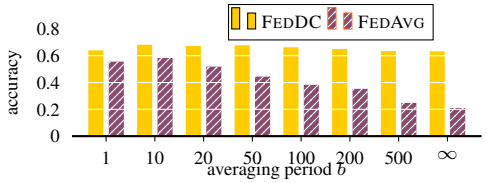


Figure 5: *Averaging periods on CIFAR10.* For 150 clients with small ResNets and 64 samples per client, we visualize the test accuracy (higher is better) of FEDDC and FEDAVG for different aggregation periods b.

We investigate the effect of non-iid data on FEDDC by studying the "pathological non-IID partition of the data" [29] here each client only sees examples from 2 out of the 10 classes of CIFAR10. We again use a subset of the dataset. The results in Tab. 2 show that FEDDC outperforms FEDAVG by a wide margin. It also outperforms FEDPROX, a method specialized on heterogeneous datasets.

As a final experiment on CIFAR10, we consider daisy-chaining with different combinations of aggregation methods, and hence its ability to serve as a building block that can be combined with other federated learning approaches. In particular, we consider the same setting as before and combine FEDPROX with daisy chaining. The results, reported in Tab. 2, show that this combination is not only successful, but also outperforms all others in terms of accuracy.

[4] Due to hardware restrictions we are limited to training 150 ResNets – hence 9600 samples across 150 clients – at once.

	CIFAR10	MRI	Pneumonia
FEDDC (ours)	**62.9** ±0.02	**78.4** ±0.61	**83.2** ±0.84
DC (baseline)	58.4 ±0.85	57.7 ±1.57	79.8 ±0.99
FEDAVG (b=1)	55.8 ±0.78	74.1 ±1.68	80.1 ±1.53
FEDAVG (b=10)	48.7 ±0.87	75.6 ±1.18	79.4 ±1.11
FEDPROX	51.1 ±0.80	76.5 ±0.50	80.0 ±0.36
FEDADAGRAD	21.8 ±0.01	45.7 ±1.25	62.5 ±0.01
FEDYOGI	31.4 ±4.37	71.3 ±1.62	77.6 ±0.64
FEDADAM	34.0 ±0.23	73.8 ±1.98	73.5 ±0.36
central	61.7 ±1.17	79.9 ±6.23	81.1 ±3.04

Table 1: Results on image data, reported is the average test accuracy of the final model over three runs (± denotes maximum deviation from the average).

	CIFAR10
FEDDC	**62.9** ±0.02
FEDDC +FEDPROX	**63.2** ±0.38
	Non-IID CIFAR10
FEDDC	**34.2** ±0.61
FEDAVG (b=1)	30.2 ±2.11
FEDAVG (b=10)	24.9 ±1.95
FEDPROX	32.8 ±0.00
FEDADAGRAD	11.7 ±0.00
FEDADAM	13.0 ±0.00
FEDYOGI	12.5 ±0.04

Table 2: Combination of FEDDC with FEDAVG and FEDPROX and non-iid results on CIFAR10 (clients see only 2 out of 10 classes).

Medical image data: Finally, we consider two real medical image datasets representing actual health related machine learning tasks, which are naturally of small sample size. For the brain MRI scans, we simulate 25 clients (e.g. hospitals) with 8 samples each. Each client is equipped with a CNN (see App. A.9.1). The results for brain tumor prediction evaluated on a test set of 53 of these scans are reported in Table 1. Overall, FEDDC performs best among the federated learning approaches and is close to the centralized model. Furthermore, the predictions are much more stable in terms of the variance of the performance compared to the centralized version. Whereas FEDPROX performed comparably poorly on CIFAR10, it is now, in this extremely small sample setup, the second best approach. Also FEDYOGI and FEDADAM fare much better than on CIFAR10, although there is still a reasonably large margin towards FEDDC.

For the pneumonia dataset, we simulate 150 clients training ResNet18 (see again App. A.9.1) with 8 samples per client, the hold out test set are 624 images. The results, reported in Table 1, show again similar trends as for the other datasets, with FEDDC outperforming all baselines and the state of the art, and being within the performance of the centrally trained model. Moreover it highlights that FEDDC enables us to train a ResNet18 to high accuracy with as little as 8 samples per client.

8 Discussion and Conclusion

We propose to combine daisy-chaining and aggregation to effectively learn high quality models in a federated setting where only little data is available locally, such as in healthcare. We formally prove convergence of our approach FEDDC, and for convex settings provide PAC-like generalization guarantees when aggregating by iterated Radon points. Empirical results on the SUSY benchmark underline these theoretical guarantees, with FEDDC matching the performance of centralized learning with as few as 2 samples per client.

Extensive empirical evaluation shows that the proposed combination of daisy-chaining and aggregation enables federated learning from small datasets in practice, consistently outperforming the state of the art, matching the performance of centralized training on a real world health dataset with as few as 8 samples per client. When using averaging as aggregation scheme, we improve upon the state of the art for federated deep learning by a large margin for the considered small sample settings, often reaching the accuracy of centralized learning. Furthermore, we show that daisy-chaining—as a building block—can be successfully combined with different aggregation operators and federated learning methods, such as FEDAVG, Radon machines, and FEDPROX.

FEDDC naturally permits differential privacy mechanisms that introduce noise on model parameters, e.g., to safeguard against membership inference, poisoning and backdoor attacks. As the coordinator assigns random permutations for daisy-chaining rounds, FEDDC is also robust against model update inference attacks. FEDDC uses additional daisy-chaining rounds, resulting in a linear increase in communication, which is, however, not an issue in our considered cross-SILO applications.

Furthermore, FEDDC outperforms FEDAVG in practice, independent of the amount of communication. Improving the communication efficiency considering settings with limited bandwidth, e.g., model training on mobile devices, would make for engaging future work.

We conclude that daisy-chaining lends itself as an effective building block to improve federated learning when little data is available per client. FEDDC, thus, might offer a solution to the long standing open problem of federated learning in healthcare, where very few, undisclosable samples are available at each site.

References

[1] Idan Achituve, Aviv Shamsian, Aviv Navon, Gal Chechik, and Ethan Fetaya. Personalized federated learning with gaussian processes. *Advances in Neural Information Processing Systems*, 34, 2021. 3

[2] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150, 2015. 19

[3] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9, 2014. 6, 14

[4] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643. PMLR, 2019. 19

[5] Kenneth L Clarkson, David Eppstein, Gary L Miller, Carl Sturtivant, and Shang-Hua Teng. Approximating center points with iterative radon points. *International Journal of Computational Geometry & Applications*, 6(03):357–377, 1996. 6

[6] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2089–2099. PMLR, 18–24 Jul 2021. 3

[7] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017. 7

[8] Ran Gilad-Bachrach, Amir Navot, and Naftali Tishby. Bayes and tukey meet at the center point. In *International Conference on Computational Learning Theory*, pages 549–563. Springer, 2004. 5

[9] Kristin L Granlund, Sui-Seng Tee, Hebert A Vargas, Serge K Lyashchenko, Ed Reznik, Samson Fine, Vincent Laudone, James A Eastham, Karim A Touijer, Victor E Reuter, et al. Hyperpolarized mri of human prostate cancer reveals increased lactate with tumor grade driven by monocarboxylate transporter 1. *Cell metabolism*, 31(1):105–114, 2020. 1

[10] Farzin Haddadpour and Mehrdad Mahdavi. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019. 2, 3, 4

[11] Baihe Huang, Xiaoxiao Li, Zhao Song, and Xin Yang. Fl-ntk: A neural tangent kernel-based framework for federated learning analysis. In *International Conference on Machine Learning*, pages 4423–4434. PMLR, 2021. 2

[12] Marwa Ibrahim, Mohammad Wedyan, Ryan Alturki, Muazzam A Khan, and Adel Al-Jumaily. Augmentation in healthcare: Augmented biosignal using deep learning and tensor representation. *Journal of Healthcare Engineering*, 2021, 2021. 2

[13] Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems (TOCS)*, 23(3):219–252, 2005. 3

[14] Michael Kamp. *Black-Box Parallelization for Machine Learning*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, Universitäts-und Landesbibliothek Bonn, 2019. 3

[15] Michael Kamp, Mario Boley, Olana Missura, and Thomas Gärtner. Effective parallelisation for machine learning. In *Advances in Neural Information Processing Systems*, volume 30, pages 6480–6491, 2017. 3, 5, 20

[16] Michael Kamp, Linara Adilova, Joachim Sicking, Fabian Hüger, Peter Schlicht, Tim Wirtz, and Stefan Wrobel. Efficient decentralized deep learning by dynamic model averaging. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 393–409. Springer, 2018. 3, 4

[17] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8420–8429, 2019. 1

[18] Péter Kiss and Tomas Horvath. Migrating models: A decentralized view on federated learning. In *Proceedings of the Workshop on Parallel, Distributed, and Federated Learning*. Springer, 2021. 3

[19] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, 2009. 2, 7

[20] Sang Gyu Kwak and Jong Hae Kim. Central limit theorem: the cornerstone of modern statistics. *Korean journal of anesthesiology*, 70(2):144, 2017. 3

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 16

[22] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021. 3

[23] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Conference on Machine Learning and Systems, 2020a*, 2020. 2, 3, 7

[24] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. In *International Conference on Learning Representations*, 2020. 3

[25] Pei Liu, Xuemin Wang, Chao Xiang, and Weiye Meng. A survey of text data augmentation. In *2020 International Conference on Computer Communication and Network Security (CCNS)*, pages 191–195. IEEE, 2020. 1

[26] Pengrui Liu, Xiangrui Xu, and Wei Wang. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. *Cybersecurity*, 5(1):4, 2022. 7, 19

[27] Chuan Ma, Jun Li, Ming Ding, Howard H Yang, Feng Shu, Tony QS Quek, and H Vincent Poor. On safeguarding privacy and security in the framework of federated learning. *IEEE network*, 34(4):242–248, 2020. 6, 18

[28] Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2021. 3

[29] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017. 2, 3, 7, 8

[30] Corrie A Painter, Esha Jain, Brett N Tomson, Michael Dunphy, Rachel E Stoddard, Beena S Thomas, Alyssa L Damon, Shahrayz Shah, Dewey Kim, Jorge Gómez Tejeda Zañudo, et al. The angiosarcoma project: enabling genomic and clinical discoveries in a rare cancer through patient-partnered research. *Nature medicine*, 26(2):181–187, 2020. 1

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 7, 20

[32] Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019. 3

[33] Viraj Prabhu, Anitha Kannan, Murali Ravuri, Manish Chaplain, David Sontag, and Xavier Amatriain. Few-shot learning for dermatological disease diagnosis. In *Machine Learning for Healthcare Conference*, pages 532–552. PMLR, 2019. 2

[34] Johann Radon. Mengen konvexer Körper, die einen gemeinsamen Punkt enthalten. *Mathematische Annalen*, 83(1):113–115, 1921. 5

[35] Sashank J Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2020. 2, 3, 7, 20

[36] Amirhossein Reisizadeh, Farzan Farnia, Ramtin Pedarsani, and Ali Jadbabaie. Robust federated learning: The case of affine distribution shifts. In *Advances in Neural Information Processing Systems*, volume 33, pages 21554–21565. Curran Associates, Inc., 2020. 3

[37] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International Conference on Artificial Intelligence and Statistics*, pages 2021–2031. PMLR, 2020. 3

[38] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014. 5

[39] Ohad Shamir and Nathan Srebro. Distributed stochastic optimization and learning. In *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 850–857. IEEE, 2014. 1, 3, 5, 16

[40] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017. 2, 7, 19

[41] Xiaoping Su, Xiaofan Lu, Sehrish Khan Bazai, Eva Compérat, Roger Mouawad, Hui Yao, Morgan Rouprêt, Jean-Philippe Spano, David Khayat, Irwin Davidson, et al. Comprehensive integrative profiling of upper tract urothelial carcinomas. *Genome biology*, 22(1):1–25, 2021. 1

[42] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019. 7, 19

[43] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010. 2

[44] John W Tukey. Mathematics and picturing data. In *Proceedings of the International Congress of Mathematics*, volume 2, pages 523–531, 1975. 5

[45] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638, 2016. 2

[46] Ulrike Von Luxburg and Bernhard Schölkopf. Statistical learning theory: Models, concepts, and results. In *Handbook of the History of Logic*, volume 10, pages 651–706. Elsevier, 2011. 5

[47] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *International Conference on Learning Representations*, 2019. 3

[48] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020. 7, 19

[49] Qian Yang, Jianyi Zhang, Weituo Hao, Gregory P. Spell, and Lawrence Carin. Flop: Federated learning on medical datasets using partial networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 3845–3853. Association for Computing Machinery, 2021. 3

[50] Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5693–5700, 2019. 4, 17

[51] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pages 7252–7261. PMLR, 2019. 2

[52] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. Batchcrypt: Efficient homomorphic encryption for cross-silo federated learning. In *USENIX Annual Technical Conference*, pages 493–506, 2020. 19

[53] Ligeng Zhu and Song Han. Deep leakage from gradients. In *Federated learning*, pages 17–31. Springer, 2020. 6, 7

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The contributions listed in the abstract and introduction directly reflect the theoretical contributions in Sec. 5, the privacy discussion in Sec. 6, and the empirical evaluation in Sec. 7.
   (b) Did you describe the limitations of your work? [Yes] We describe the limitations in terms of communication complexity in Sec. 4, and in terms of privacy in Sec. 6 and App. A.8.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We discuss the potential for privacy risks through the naïve implementation of FEDDC and discuss measures to defend data privacy, as well as to defend against poisoning and backdoor attacks.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] The setting is described in Sec. 3, assumptions specific to Cor. 1 are stated within the corollary, all further assumptions are explicitly stated in Assumptions 2.
   (b) Did you include complete proofs of all theoretical results? [Yes] All proofs are provided in the Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] An anonymized link to a git repository is provided in the paper that contains all code and data to directly reproduce the results. Furthermore, App. A.9 describes the experimental setup in detail.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Details are provided in App. A.9, specific implementation details are provided in the implementation.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report maximum deviation from the average over multiple runs.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Details are provided in App. A.9
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the creators of the datasets we use and cite the papers of the baselines we compare to.
   (b) Did you mention the license of the assets? [Yes] Our implementation is published under Apache 2.0 license, which is included in the git repository.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The implementation of our method and experiments is provided as a URL.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] Datasets we use are public, we use them under their respective license.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] The data we are using has been checked by the creators for not containing personal data or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Additional Empirical Results

Train and test accuracies on CIFAR10: In Table 3 we provide the accuracies on the entire training set for the final model, together with test accuracies, on CIFAR10. The high training accuracies of FEDAVG (≈ 0.97)—and to a lesser degree FEDPROX (0.96)—indicate overfitting on local data sets. The poor training performance of FEDADAGRAD, FEDYOGI, and FEDADAM hint at insufficient model updates. A possible explanation is that the adaptive learning rate parameter (which is proportional to the sum of past model updates) becomes large quickly, essentially stopping the training process. The likely reason is that due to large differences in local data distributions, model updates after each daisy-chaining round are large.

	Test	**Train**
FEDDC (ours)	62.9 $\pm$0.02	94.7 $\pm$0.52
DC (baseline)	58.4 $\pm$0.85	94.1 $\pm$2.31
FEDAVG (b=1)	55.8 $\pm$0.78	97.2 $\pm$0.87
FEDAVG (b=10)	48.7 $\pm$0.87	97.4 $\pm$0.23
FEDPROX	51.1 $\pm$0.80	95.9 $\pm$0.42
FEDADAGRAD	21.8 $\pm$0.01	31.7 $\pm$0.25
FEDYOGI	31.4 $\pm$4.37	72.4 $\pm$0.90
FEDADAM	34.0 $\pm$0.23	73.9 $\pm$0.89

Table 3: Train and test accuracy on CIFAR10 of the final model over three runs ($\pm$ denotes maximum deviation from the average).

Local Dataset Size: In our experiments, we used dataset sizes common in the medical domain, e.g., for radiological images. To further investigate the impact of local dataset sizes on the performance of FEDDC wrt. FEDAVG, we evaluate the performance for local dataset sizes ranging from 2 to 256 (given the size of CIFAR10, 256 is the maximum without creating overlap between batches). The results in Fig. 6 show that FEDDC outperforms all baselines for smaller local datasets. Only for as much as 256 examples FEDAVG performs nearly as good as FEDDC.

These further confirm that FEDDC is capable of handling heterogeneous data: for $n < 10$ the clients only see a subset of labels due to the size of their local datasets (with n=2, each client can at most observe two classes). We find this a more natural non-iid setting. These results indicate that the shuffle mechanism indeed mitigates data heterogeneity well. A further study of the impact of non-iid data, a comparison with personalized FL, and potential improvements to the shuffling scheme are interesting directions for future work

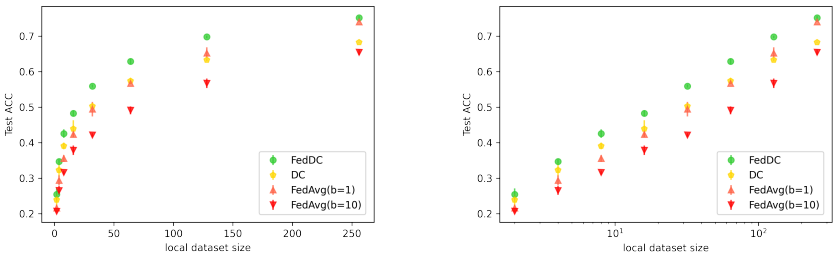
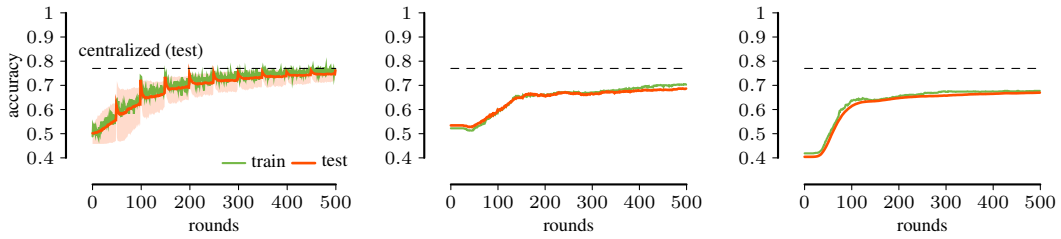


Figure 6: Test accuracy wrt. local dataset size on CIFAR10 with 150 clients ($n = 2^i$ for $i \in \{1, \dots, 8\}$) with linear (left) and logarithmic (right) x-axis.

Additional Results on SUSY In Fig. 7 we repeat the experiments from Sec. 5 where we compare FEDDC using the iterated Radon point with standard federated learning on the SUSY binary classification dataset [3] and FEDAVG, training a linear model on 441 clients with only 2 samples per

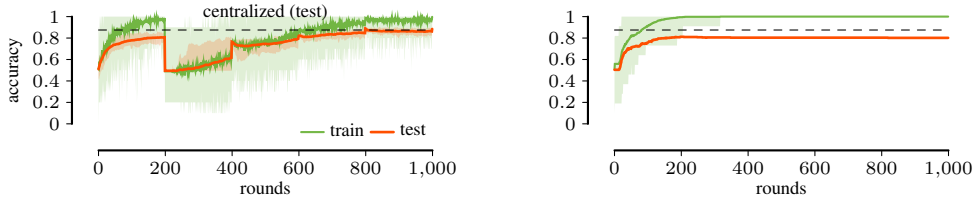


(a) FEDDC with Radon point with $d = 1$, $b = 50$.

(b) Federated learning with Radon point with $b = 1$.

(c) Federated learning with averaging with $b = 1$.

Figure 7: *Results on SUSY.* We visualize results in terms of train (green) and test error (orange) for FEDDC (a) and standard federated learning (b), both using Radon points for aggregation, and FEDAVG (c). The network has 441 clients with 2 data points per client. The performance of a central model trained on all data is indicated by the dashed line.



(a) FEDDC with $d = 1$, $b = 200$.

(b) FEDAVG with $b = 1$.

Figure 8: *Synthetic data results.* Comparison of FEDDC (d=1,b=200) (left) and FEDAVG (b=1) (right) for training fully connected neural networks on a synthetic dataset. Mean and confidence test accuracy of each client reported in orange, the test accuracy of centralized training is indicated by the dashed black line.

client. For this experiment, we set the aggregation period to $b = 1$ for federated learning with the Radon point and FEDAVG, so that all method have the same amount of communication. The results with $b = 1$ both for federated learning with the Radon point (ACC=66.7) and FEDAVG (ACC=65.4) are virtually the same as for $b = 50$. Since we measure the performance of the aggregate model, if a model aggregation took place, the overfitting of local models is not visible in the plot.

Additional Results on Synthetic Data For completeness, we include results for FEDAVG (b=1) for the synthetic data in Fig. 8. The higher amount of communication improves the accuracy of FEDAVG from 76.0 for $b = 200$ to 80.2 for $b = 1$, but it is still outperformed substantially by FEDDC with an accuracy of 88.5.

A.2 Realistic Dataset Size Distribution

In medical applications, a common scenario is that some hospitals, e.g., university clinics, hold larger datasets, while small clinics, or local doctor's offices only hold very small datasets. To simulate such a scenario, we draw local dataset sizes for a dataset of size n so that a fraction c of the clients hold only a minimum number of samples n_{min} (the local doctor's offices), and the other clients have an increasing local dataset size starting from n_{min} until all data is distributed. That is, for clients $i = [1, \dots, m - \lfloor cm \rfloor]$ the dataset sizes are given by $n_{min} + ai$ with

$$a = \frac{2(n - (\lfloor cm \rfloor) n_{min})}{\lfloor (1-c)m \rfloor (\lfloor (1-c)m \rfloor - 1)}$$

.

We have chosen $c = 0.3$ and $n_{min} = 2$.

Comparison with Centralized training and mini-batch SGD on CIFAR10 The centralized baseline on CIFAR10 reported in Tab. 1 is run with the same parameters as the federated variants, i.e., an

	MRI
FEDDC (d=1)	**74.7** ±0.61
FEDDC (d=2)	**74.4** ±1.15
FEDDC (d=4)	**80.6** ±0.66
FEDDC (d=5)	**77.8** ±1.42
DC (baseline)	53.9 ±0.25
FEDAVG (b=1)	70.1 ±2.59
FEDAVG (b=10)	75.3 ±2.37
central	79.9 ±6.23

Table 4: Results for realistic dataset size distribution on MRI, reported is the average test accuracy of the final model over three runs (± denotes maximum deviation from the average).

SGD with learning rate 0.01, mini-batch size $B = 64$ and a learning rate scheduler with a decay rate of 0.5, to ensure the results are comparable. This achieves an accuracy of 61.7 ± 1.17. An optimal choice of parameters for the centralized baseline, i.e., a learning rate of 0.2 and similar mini-batch size and decay rate, improve the accuracy to 64.2 ± 0.54, slightly outperforming FEDDC.

We also compare to to mini-batch SGD, i.e., central updates where gradients are computed distributedly. We use the same setup, i.e., $m = 150$ clients, a learning rate of $\lambda = 0.01$ and a mini-batch size of $B = 64$, so that the effective mini-batch size for each update is $mB = 9600$. Here, mini-batch SGD achieves a test accuracy of 19.47 ± 0.68, outperforming centralized training with $B = 9600$ and the small learning rate of $\lambda = 0.000067$, due to a more favorable learning rate. If we instead use $B = 1$ and thus get an effective mini-batch size of $B = 150$, the results are substantially improved to an accuracy of 50.14 ± 0.63, underlining the negative effect of the large batch size in line with the theoretical analysis of Shamir and Srebro [39]. Running it 64 times the number of rounds that FedDC uses improves the accuracy just slightly to 54.3. Thus, even with optimal $B = 1$ and a 64-times slower convergence, mini-batch SGD is outperformed by both FedAvg and FedDC.

Communication efficiency of FEDDC Although communication is not a concern in cross-silo applications, such as healthcare, the communication efficiency of FEDDC is important in classical federated learning applications. We therefore compare FEDDC with varying amounts of communication to FEDAVG on CIFAR10 in Tab. 5. The results show that FEDDC (d=2) outperforms FEDAVG (b=1) with half the amount of communication, and FEDDC (d=5) performs similar to FEDAVG (b=1) with five times less communication. FEDDC with $d = 10$ and $b = 10$ significantly outperforms FEDAVG (b=10) while requiring the same amount of communication.

	CIFAR10
FEDDC (d=1,b=10)	**62.9** ±0.02
FEDDC (d=2,b=10)	**60.8** ±0.65
FEDDC (d=5,b=10)	**55.4** ±0.11
FEDDC (d=10,b=20)	**53.8** ±0.47
FEDAVG (b=1)	55.8 ±0.78
FEDAVG (b=10)	51.1 ±0.87
central	61.7 ±1.17

Table 5: Communication efficiency of FEDDC compared to FEDAVG.

A.3 Additional Results on MNIST

In order to demonstrate the efficiency of FEDDC on clients that achieve state-of-the-art performance we perform experiments on the MNIST [21] dataset. We use a CNN with two convolutional layers with max-pooling, followed by two linear layers with 1024, resp. 100 neurons. Centralized training on all $60\,000$ training samples of MNIST achieves a test accuracy of 99.4 which is similar to the

state-of-the-art. The results for $m = 50$ clients in Tab. 6 show that FEDDC outperforms FEDAVG both with the same amount of communication, i.e., FEDAVG ($b = 1$) and FEDAVG ($b = 10$). In line with the results on CIFAR10 (cf. Fig. 6), the advantage of FEDDC shrinks with increasing local dataset size. Using $n = 1200$, i.e., the full training set distributed over $m = 50$ clients, results in virtually the same performance of FEDDC and FEDAVG, both reaching a test accuracy of 94.

	FEDDC (d=1,b=10)	**FEDAVG (b=1)**	**FEDAVG (b=10)**
$n = 8$	87.6	84.4	84.9
$n = 1200$	96.7	96.3	96.5

Table 6: Performance of FEDDC and FEDAVG on MNIST for varying local dataset sizes n.

A.4 Proof of convergence

Corollary. *Let the empirical risks $\mathcal{E}^i_{emp}(h) = \sum_{(x,y)\in D^i} \ell(h_i(x), y)$ at each client be L-smooth with σ^2-bounded gradient variance and G^2-bounded second moments, then* FEDDC *with averaging and SGD as learning algorithm has a convergence rate of $\mathcal{O}(1/\sqrt{mT})$, where $T \in \mathbb{N}$ is the number of local updates.*

Proof. We assume that each client $i \in [m]$ uses SGD as learning algorithm. In each iteration $t \in [T]$, a client i computes the gradient $\mathbf{G}^i_t = \nabla\ell(h^i_t(x), y)$ with $x, y \in D^i$ drawn randomly and updates the local model $h^i_{t+1} = h^i_t - \gamma G^i_t$, where $\gamma > 0$ denotes the learning rate. FEDDC with daisy-chaining period d and averaging period b, run for T local iterations, computes T/b local gradients at each of the m clients before averaging. Each local gradient is computed on an iid sample from $\mathcal{D}$, independent of whether local models are permuted. Therefore, FEDDC with averaging and SGD is equivalent to parallel restarted SGD (PR-SGD) [50] with b/d times larger local datasets. Yu et al. [50] analyze the convergence of PR-SGD with respect to the average $\overline{h}_t$ of local models in round t. Since $\mathcal{E}^i_{emp}(h) = \sum_{(x,y)\in D^i} \ell(h_i(x), y)$ at each client be L-smooth with σ^2-bounded gradient variance and G^2-bounded second moments, Theorem 1 in Yu et al. [50] is applicable. It then follows from Corollary 1 [50] that for $\gamma = \sqrt{m}/(L\sqrt{T})$ and $b \leq T^{\frac{1}{4}}/m^{\frac{3}{4}}$ it holds that

$$\begin{aligned}\frac{1}{T}\,\mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{E}^i_{emp}\left(\overline{h}_t(x), y\right)\right]\right] \leq & \frac{2L}{\sqrt{mT}}\left(\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{E}^i_{emp}\left(\overline{h}_0(x), y\right)\right]\right. \\ & \left. - \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{E}^i_{emp}\left(\overline{h}^*(x), y\right)\right]\right) \\ & + \frac{1}{\sqrt{mT}}\left(4G^2 + \sigma^2\right) \in \mathcal{O}\left(\frac{1}{\sqrt{mT}}\right) \quad .\end{aligned}$$

Here, the first expectation is over the draw of local datasets and $\overline{h}^*$ is given by

$$\overline{h}^* = \arg\min_{h\in\mathcal{H}} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{E}_{emp}(\overline{h}(x), y)\right].$$

Thus, FEDDC with averaging and SGD converges in $\mathcal{O}(1/\sqrt{mT})$. □

A.5 Proof of Lemma 4

Lemma. *Given $\delta \in (0, 1]$, $m \in \mathbb{N}$ clients, and $k \in [m]$, if Algorithm 1 with daisy chaining period $d \in \mathbb{N}$ is run for $T \in \mathbb{N}$ rounds with*

$$T \geq d\frac{\ln\delta}{\ln\left(\frac{m-1}{m}\right)(m-k+1)m}$$

then each local model has seen at least k distinct datasets with probability $1 - \gamma$.

Proof. For m clients with m local datasets, the chance of a client i to not see dataset j after τ many permutations is $\left(\frac{m-1}{m}\right)^\tau$. The probability that each of the m clients is not seeing $m-k+1$ other datasets is hence

$$\prod_{j=1}^{m-k+1}\left(\frac{m-1}{m}\right)^\tau = \left(\frac{m-1}{m}\right)^{\tau(m-k+1)} ,$$

and corresponds to the probability of each client seeing less than k distinct other datasets. The probability of all clients seeing at least k distinct datasets is hence at least

$$1-\left(\frac{m-1}{m}\right)^{\tau(m-k+1)m} \overset{!}{\geq} 1-\gamma$$
$$\Leftrightarrow \left(\frac{m-1}{m}\right)^{\tau(m-k+1)m} \overset{!}{\leq} \gamma \ .$$

Taking the logarithm on both sides with base $(m-1)/m < 1$ yields

$$\tau(m-k+1)m \geq \frac{\ln\gamma}{\ln\frac{m-1}{m}} \ .$$

Multiplying with $m-k+1$ and observing that τ many daisy-chaining rounds with period d require $T=\tau d$ total rounds yields the result. □

A.6 Proof of model quality improvement by FEDDC

Proposition. *Let $\mathcal{H}$ be a model space with Radon number $r\in\mathbb{N}$, ε a convex risk , and $\mathcal{A}$ a learning algorithm with sample size $n_0(\epsilon,\delta)$. Given $\epsilon>0$, $\delta\in(0,r^{-1})$ and any $h\in\mathbb{N}$, and local datasets $D_1,\ldots,D_m$ of size $n\in\mathbb{N}$ with $m\geq r^h$, then Alg. 1 using the Radon point with aggr. period*

$$b \geq d\frac{\ln\sqrt{\delta}}{\ln\left(\frac{m-1}{m}\right)\left(m-\left\lceil\frac{n_0\left(\epsilon,\sqrt{\delta}\right)}{n}\right\rceil+1\right)m} \tag{4}$$

improves model quality in terms of (ϵ,δ)-guarantees.

Proof. The number of daisy-chaining rounds before computing a Radon point ensure that with probability $1-\sqrt{\delta}$ all local models are trained on at least kn samples with $k=\lceil n_0(\epsilon,\sqrt{\delta})/n\rceil$, i.e., each model is trained on at least $n_0(\epsilon,\sqrt{\delta})$ samples and thus an $(\epsilon,\sqrt{\delta})$-guarantee holds for each model with probability $1-\sqrt{\delta}$. It follows from Eq. 2 that the probability that the risk is higher than ϵ is

$$\mathbb{P}\left(\varepsilon(\mathfrak{r}_h)>\epsilon\right) < \left(r\sqrt{\delta}\sqrt{\delta}\right)^{2^h} = \left(r\delta\right)^{2^h} \ .$$

The result follows from $\delta<r^{-1}$ and Eq. (2). □

A.7 Note on Communication Complexity

In our analysis of the communication complexity, we assume the total amount of communication to be linear in the number of communication rounds. For some communication systems the aggregated model can be broadcasted to individual clients which is not possible in daisy-chaining rounds, reducing the communication complexity in FedAvg. In most scenarios, fiber or GSM networks are used where each model has to be sent individually, so there is no substantial difference between broadcasting a model to all clients and sending an individual model to each client. Therefore, also in this case the amount of communication rounds determines the overall amount of communication.

A.8 Extended Discussion on Privacy

Federated learning only exchanges model parameters, and no local data. It is, however, possible to infer upon local data given the model parameters [27]. In classical federated learning there are two types of attacks that would allow such inference: (i) an attacker intercepting the communication

of a client with the coordinator obtaining model updates to infer upon the clients data, and (ii) a malicious coordinator obtaining models to infer upon the data of each client. A malicious client cannot learn about other clients data, since it only obtains the average of all local models. In federated daisy-chaining there is a third possible attack: (iii) a malicious client obtaining model updates from another client to infer upon its data.

In the following, we discuss potential defenses against these three types of attacks in more detail. Note that we limit the discussion on attacks that aim at inferring upon local data, thus breaching data privacy. Poisoning [4] and backdoor [42] attacks are an additional threat in federated learning, but are of less importance for our main setting in healthcare: there is no obvious incentive for a hospital to poison a prediction. It is possible that FEDDC presents a novel risk surface for those attacks, but such attack strategies are non-obvious. Robust aggregation, such as the Radon point, are suitable defenses against such attacks [26]. Moreover, the standard mechanisms that guarantee differential privacy also defend against backdoor and poising attacks [42].

A general and wide-spread approach to tackle all three possible attack types is to add noise to the model parameters before sending. Using appropriate clipping and noise, this guarantees ϵ, δ-differential privacy for local data [48] at the cost of a slight-to-moderate loss in model quality.

Another approach to tackle an attack on communication (i) is to use encrypted communication. One can furthermore protect against a malicious coordinator (ii) by using homomorphic encryption that allows the coordinator to average models without decrypting them [52]. This, however, only works for particular aggregation operators and does not allow to perform daisy-chaining. Secure daisy-chaining in the presence of a malicious coordinator (ii) can, however, be performed using asymmetric encryption. Assume each client creates a public-private key pair and shares the public key with the coordinator. To avoid the malicious coordinator to send clients its own public key and act as a man in the middle, public keys have to be announced (e.g., by broadcast). While this allows sending clients to identify the recipient of their model, no receiving client can identify the sender. Thus, inference on the origin of a model remains impossible. For a daisy-chaining round the coordinator sends the public key of the receiving client to the sending client, the sending client checks the validity of the key and sends an encrypted model to the coordinator which forwards it to the receiving client. Since only the receiving client can decrypt the model, the communication is secure.

In standard federated learning, a malicious client cannot infer specifically upon the data of another client from model updates, since it only receives the aggregate of all local models. In federated daisy-chaining, it receives the model from a random, unknown client in each daisy-chaining round. Now, the malicious client can infer upon the membership of a particular data point in the local dataset of the client the model originated from, i.e., through a membership inference attack [40]. Similarly, the malicious client can infer upon the presence of data points with certain attributes in the dataset [2]. The malicious client, however, does not know the client the model was trained on, i.e., it does not know the origin of the dataset. Using a random scheduling of daisy-chaining and aggregation rounds at the coordinator, the malicious client cannot even distinguish between a model from another client or the average of all models. Nonetheless, daisy-chaining opens up new potential attack vectors (e.g., clustering received models to potentially determine their origins). These potential attack vectors can be tackled in the same way as in standard federated learning, i.e., by adding noise to model parameters as discussed above, since "[d]ifferentially private models are, by construction, secure against membership inference attacks" [40].

A.9 Details on Experiments Setup

In this section we provide all details to reproduce the empirical results presented in this paper. Furthermore, the implementation provided at `https://anonymous.4open.science/r/FedDC-19E0` allows to directly reproduce the result. Experiments were conducted on an NVIDIA DGX with 6 A6000 GPUs.

A.9.1 Network architectures

Here, we detail network architectures considered in our empirical evaluation All code is publicly available to ensure reproducability.

MLP for Synthetic Data A standard MLP with ReLU activations and three linear layers of size 100,50,20.

Averaging round experiment For this set of experiments we use smaller versions of ResNet architectures with 3 blocks, where the blocks use $16, 32, 64$ filters, respectively. In essence, these are smaller versions of the original ResNet18 to keep training of 250 networks feasible.

CIFAR10 & Pneumonia For CIFAR10, we consider a standard ResNet18 architecture, where weights are initialized by a Kaiming Normal and bias are zero-initialized. Each client constructs and initializes a ResNet network separately. For pneumonia, X-ray images are resized to be of size (224,224).

MRI For the MRI scan data, we train a small convolutional network of type Conv(32)-Batchnorm-ReLU-MaxPool-Conv(64)-Batchnorm-ReLU-MaxPool-Linear, where Conv(x) are convolutional layers with x filters of kernel size 3. The pooling layer uses a stride of 2 and kernel size of 2. The Linear layer is of size 2 matching the number of output classes. All scan images are resized to be of size (150,150).

A.9.2 Training setup

In this section, we give additional information for the training setup for individual experiments in our empirical evaluation.

SUSY experiments SUSY is a binary classification dataset with 18 features. We train linear models with stochastic gradient descent (learning rate 0.0001, found by grid-search on an independent part of the dataset) on 441 clients. We aggregate every 50 rounds. Aggregation is performed via the iterated Radon point [15] with $h = 2$ iterations. FEDDC performs daisy-chaining with period $d = 1$. The test accuracy is evaluated on a test set with $1\,000\,000$ samples drawn iid at randomly.

Synthetic Data The synthetic binary classification dataset is generated by the sklearn [31] `make_classification` function with 100 features of which 20 are informative, 60 are redundant, and 5 are repeated. We generate 3 clusters per class with a class separation of 1.0, a shift of 1.0 and a scale of 3.0. Class labels are randomly flipped with probability 0.02.

Averaging rounds parameter optimization To find a suitable number when averaging should be carried out, we explore $b \in \{1, 10, 20, 50, 100, 200, 500, \infty\}$ on CIFAR10 using 250 clients each equipped with a small ResNet. We assign 64 samples to each client drawn at random (without replacement) from the CIFAR10 training data and use a batch size of 64. For each parameter, we train for $10k$ rounds with SGD using cross entropy loss and initial learning rate of 0.1, multiplying the rate by a factor of $.5$ every 2500 rounds.

FedAdam, FedAdagrad, and FedYogi We use the standard values for β_1 and β_2, i.e., $\beta_1 = 0.9$, $\beta_2 = 0.999$, as suggested in Reddi et al. [35]. We optimized learning rate η_l and global learning rate η from the set $\{0.001, 0.01, 0.1, 1.0, 2.0\}$ with optimal parameters $\eta_l = 0.1$ and $\eta = 1.0$.

CIFAR10 differentail privacy and main experiments We keep the same experimental setup as for hyperparameter finding, but now use 100 clients each equipped with a ResNet18. For the differential privacy experiment