Fighting Against Attacks on Decentralized Learning Systems with Dynamic Topologies

Filip Carevic
Student
filip.carevic@epfl.ch

Rishi Sharma
Mentor
rishi.sharma@epfl.ch

Rafael Pires
Mentor
rafael.pires@epfl.ch

Abstract

Decentralized machine learning gained popularity as a promising solution to privacy and data management challenges introduced by increased data generation in recent years. In order to bring this paradigm closer to practical use, this field needs a lot of research in conditions similar to real-world scenarios. To help in the better understanding of privacy guarantees in such a condition, in this paper, we perform, to the best of our knowledge, the first privacy evaluation of decentralized machine learning in a non-IID setting. In the assumed passive threat model, we evaluate the effectiveness of membership inference and gradient inversion attacks. Finally, we disclose how dynamic topologies could be used to mitigate the risks and enhance privacy guarantees offered by decentralized machine learning.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

Keywords: privacy evaluation, dynamic topologies, non-IID

ACM Reference Format:
Filip Carevic, Rishi Sharma, and Rafael Pires. 2018. Fighting Against Attacks on Decentralized Learning Systems with Dynamic Topologies. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

An enormous increase in the amount of data produced by mobile and IoT devices introduced new issues in data gathering for the purpose of traditional centralized machine learning: (i) vast amounts of data has to be moved and stored in a centralized location and (ii) collection of sensitive data has to comply with legislation (e.g., GDPR [30]). To resolve the aforementioned challenges, the paradigms of collaborative learning emerged, with the main idea of performing the learning process while keeping sensitive user data locally on personal devices.

The first paradigm to gain traction is **Federated ML** [23], where model parameters (i.e., *FedAvg*) or model updates (i.e., *FedSGD*) are shared with the central server. In particular, each data owner performs local training and at regular time intervals sends model weights/updates to the central server where the global model is built by averaging received model updates. However, due to the assumed centrality, this paradigm has disadvantages related to (i) communication efficiency [16, 26, 27], where the centralized server quickly becomes a bottleneck, and (ii) privacy and security risks [3, 9, 10, 14, 19, 24, 25, 31], where the centralized server is the single point of trust and the single point of failure. In addition, this paradigm is extremely difficult to implement unless the underlying network topology supports centralized communication.

To address the drawbacks of Federated ML, the second paradigm of collaborative learning, **Decentralized ML** [21], has been introduced. In Decentralized ML, instead of sending data to the central aggregator, the participants share and aggregate model parameters only with their neighbors. The global model is obtained by achieving consensus in the system. In other words, Decentralized ML leverages peer-to-peer communication in order to resolve issues created by implicit centrality in Federated ML. As such, Decentralized ML became an area of active research. Dozens of theoretical and empirical studies have been performed in order to explain and relax conditions for the system convergence [21, 22, 28], reduce communication overhead [7, 29] and investigate privacy or security risks [25]. However, most of the studies assume an IID environment [21, 22, 25] where every local dataset is a random uniform subsample of a single global distribution. In real-world scenarios, data is produced in diverse conditions, thus the distributions of local datasets are more likely to be distant. For example, the amount of noise in data collected for sound processing heavily depends on the quality of recording devices. Recent works focus on understanding the learning process in this **Non-IID** environment [11], and designing algorithms robust to high data variance [28]. However, to the best of our knowledge, no work has performed privacy evaluation in this heterogeneous environment.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

In this paper, we perform, to the best of our knowledge, the first privacy evaluation of decentralized learning in a Non-IID setting. We assess privacy guarantees against passive (i.e, honest-but-curious) adversaries. In particular, we evaluate the vulnerability towards the gradient inversion and membership inference attacks and demonstrate how something as simple as randomness could be used to mitigate the effectiveness of the aforementioned attacks.

Organization. Section 2 introduces fundamental concepts required to understand the topic. We discuss the privacy threats and describe how to leverage the dynamic topologies as defense mechanisms against membership inference attacks in Section 3, and gradient inversion attacks in Section 4. We highlight related work in Section 5, disclose future work in Section 6 and draw relevant conclusions in Section 7.

2 Background

In this section, we provide the background information necessary to comprehend the key components of the study.

Decentralized ML. Decentralized ML enables collaborative model training without sharing the sensitive raw data. The main goal of Decentralized ML is to combine local learnings into a single machine learning model. Instead of relying on a central aggregator, the nodes employ peer-to-peer synchronous [21] or asynchronous [22] communication, thus alleviating the problems induced by centralized topology.

Formally, the problem of finding the best global parameters θ (with respect to the unobserved union of local data distributions D_i) that decentralized learning tries to solve is defined as:

$$\underset{\theta}{argmin} \frac{1}{n} \sum_{i=1}^{n} E_{(x_i, y_i) \sim D_i} [L_i(x_i, y_i, \theta)] \tag{1}$$

where L_i is the loss function calculated on i^{th} node, and n is the total number of nodes in the system.

In this study, we focus on a family of learning algorithms based on the distributed consensus averaging, with Distributed-Parallel SGD (i.e., D-PSGD) [21] as representative. In D-PSGD, all users in the system agree on the same hyperparameters, model architecture and weights initialization. The communication is peer-to-peer with the topology modeled as undirected graph $G = (N, E)$, where $\{i, j\} \in E$ represents a communication channel (i.e., an edge) between nodes i and j. The pseudocode of this algorithm is given in *Algorithm* 1. In every communication round, each node performs: (i) stochastic gradient descent (SGD) step on the local data and (ii) sharing and aggregation of model parameters with a full set of neighbors. One execution of this algorithm is called the *communication step/round*. The algorithm terminates when the system reaches the consensus state.

Intuitively, the first step performs learning on the local data, while the second step transfers knowledge and capabilities across neighboring models. From a statistical point of view, the local training step maximizes the likelihood function by selecting the parameter values of local models that make the observed local data most probable, whereas the aggregation step finds the model parameters that maximize the joint likelihood of posterior distributions of neighboring models.

Algorithm 1 D-PSGD

```
Set of nodes $N$, Hastings weight-matrix $W$, learning rate $\eta$, local dataset $D_i$ of $i^{th}$ node , total number of communication rounds $T$, local model weights $x_{t,i}$ of $i^{th}$ node in $t^{th}$ communication round, set of neighbours $\tilde{N}(i)$ for $i^{th}$ node.
Require: $\forall i \in |N|. \ x_{0,i} = x_0$

1: for $t = 1, 2, \dots, T$ do
                                              ▷ LocalSGD_Step
2:     $\psi_i \sim D_i$
3:     $x_{i,t+\frac{1}{2}} = x_{i,t} - \eta * \nabla_x L(x_{i,t}, \psi_i)$
                                              ▷ Averaging_Step
4:     $\forall n \in \tilde{N}(i)$ send $x_{i,t+\frac{1}{2}}$ to $n$
5:     $\forall n \in \tilde{N}(i)$ receive $x_{n,t+\frac{1}{2}}$ from $n$
6:     $x_{i,t+1} = \sum_{j=1}^{N} W_{i,j} * x_{j,t+\frac{1}{2}}$
7: end for
```

In our work, we run experiments in static and dynamic graph topologies. In order to ensure that, in both of these settings, models converge on average to a stationary point of the optimization problem defined in 1, we employ weighted averaging as the aggregation function, with weight calculation adopted from Metropolis-Hastings algorithm [32] [1]:

$$W_{ij} = \begin{cases} 1/(1 + max\{d_i(t), d_j(t)\}) & \{i, j\} \in E(t) \\ 1 - \sum_{k \in \tilde{N}_i(t)} W_{ik}(t) & i = j \\ 0 & otherwise \end{cases} \tag{2}$$

where $d_i(t)$ is the degree of i^{th} node, $E(t)$ set of edges, and $\tilde{N}(i)$ set of neighbours for i^{th} node at time-step t.

Non-IID. In practice, the data is generated in different contexts and diverse conditions. The devices placed in the same geolocation are plausible to produce correlated measurements (e.g., nearby camera devices capture similar image backgrounds, while microphones record similar background noise). On the other hand, if devices are placed in different surroundings, it is natural to assume variations in label distributions (e.g., terrestrial cameras cannot capture images of deep-sea animals). Henceforth, the assumptions of identical and independent data (i.e., *IID*) in real-world scenarios do not hold. In other words, in a *non-IID* environment, local data distributions $P_i(x, y)$, where x is a feature and y a label, for each of the node i vary significantly.

[1]The proof of convergence with time-varying Metropolis weights is shown in the original work [32].

Following the classification from [34], non-IID setups distinguish **attribute**, **label**, and **quantity** skewness.[2]. Attribute skewness implies setting where marginal feature distribution $P_i(x)$ across attributes vary for each node, quantity skewness assumes differences in amount of data stored at each node, while label skewness suggests dissimilarities in marginal label distribution $P_i(y)$ across nodes. Recent work concludes detrimental effects of data-skewness on model quality in decentralized learning [11]. Moreover, the non-IID environment became a fundamental and ubiquitous problem of collaborative learning in general [11, 20, 27, 34].

3 Membership inference attack

Firstly, we discuss the privacy guarantees offered by decentralized machine learning against membership inference attacks. In particular, we provide analysis of privacy threats in static graph topologies and describe how dynamic graph topologies could be used to reduce the risks.

3.1 Definition

The goal of membership inference attacks (MIA) is to infer whether a particular data sample was part of the training dataset. This presents a major privacy threat in cases where the datasets contain sensitive data such as medical or financial records.

Membership inference attacks exploit differences in behavior when models are presented with unseen samples and samples seen during the training [12]. In this paper, we focus on the membership inference attack known as the *threshold attack* [33].

The *Threshold attack* aims to exploit dissimilarities in the train and test loss distributions. The attack works as follows: Given a threshold t, the sample is classified as part of the training dataset if the corresponding loss is smaller than t.

$$ThresholdAttack(x) = \begin{cases} 1, & L(x) \leq t \\ 0, & L(x) > t \end{cases} \quad , \exists\, t \in R \tag{3}$$

The metric used to evaluate the effectiveness of this attack is **Attacker advantage**. It is defined as

$$Attacker\_advantage = Pr(success) - Pr(random\_guess)$$

where $Pr(success)$ is the probability that the sample is correctly classified as part of the training dataset, while the $Pr(random\_guess)$ is probability of a random guess (in binary classification equal to 0.5). Threshold t is chosen such that the *Attacker advantage* metric is maximized. In this study, the metric is scaled to interval [0,1].

It is important to note that the threshold attack provides the lower bound vulnerability of the model towards the whole family of MIA. Consequently, we consider the threshold attack as a vulnerability assessment method for membership inference attacks, rather than the actual attack.

[2]Note that in the original paper [34] authors introduced **temporal** skew as an additional non-IID category. However, it has been neglected since it is irrelevant to our study

3.2 Threat model

We evaluate privacy against a passive (i.e, honest-but-curious) adversary. The attacker, as part of the system, wants to infer information about sensitive datasets of other nodes to the greatest extent, while not being allowed to deviate from the protocol (i.e., the attacker is not allowed to intentionally share malicious weights, tamper with the model architecture, loss function, hyperparameters or local datasets). In this study, we are interested in examining the privacy guarantees of the system as a whole. Thus, in our experiments, every node mounts an attack on its own local model, i.e., every node performs self-evaluation.

3.3 Experimental setup

We evaluate the privacy guarantees offered by the Decentralized ML in a label-skew non-IID environment. More precisely, every node performs self-assessment twice per communication round: firstly, after the training on the local dataset, and secondly, after the aggregation phase has been completed. In our experiments, we use the following setup:

Topology. We evaluate privacy in both, static and dynamic graph topologies. We utilize regular graph topology with 96 nodes and node-degree set to 4. Moreover, the same topology is used in static and dynamic settings, with the main difference being that in a dynamic setting, we randomly choose the new set of neighbors for each node at **every** communication round. Lastly, as the baseline, we leverage the fully-connected graph with 96 nodes.

Architecture and hyperparameters. The model used in our experiments is based on the LeNet [18] model architecture. To train the model, we use the vanilla SGD optimizer (i.e., without momentum or adaptive learning rate) with learning rate set to 0.01. In every communication round, each node performs 3 mini-batch stochastic gradient descent steps with mini-batch size equal to 8. Lastly, in the averaging step, nodes perform full-model sharing.

Datasets. In our experiments, we use the CIFAR10 [17] dataset consisting of 10 evenly balanced classes. It comprises 60 000 data points, separated in global training (50 000) and global testing (10 000) sets. To establish the label-skew environment, we employ a heterogeneous partitioning scheme proposed in [23]. In particular, the training samples are sorted by class and split into shards of equal sizes. Each node is given two random shards. This partitioning ensures that most of the nodes will have local datasets containing samples from exactly 2 classes.

Metrics. As described in section 3.1, the metric we adopt is the *Attacker advantage*. For the purpose of self-assessment, each node i crafts a new dataset D_i consisting of the entire local training dataset (denoting class *member*), and a random subsample of global test dataset (denoting class *non-member*), where the size of the subsample is equal to the size of local training dataset. In other words, the dataset D_i is balanced with respect to the membership distribution.

3.4 Results and discussion

To evaluate the privacy guarantees, we observe how the average, minimum and maximum level of vulnerability (i.e., attacker advantage) in the system vary across the communication rounds (i.e., iterations). The results are depicted in the figures 2. 3 and 4, respectively. We observe:

- **The level of vulnerability in the system decreases over time.**

In general, during the classification-task training, models learn class-specific features in order to establish suitable classification boundaries. In the label-skew non-IID environment, during the local SGD phase, models are only able to learn the characteristic features for the limited number of classes. In such a setting, models are prone to overfitting the local training dataset, which amplifies the differences in behaviours under the seen and unseen samples in the initial rounds of training. In order to be able to generalize on the global data distribution, models rely on the process of transfer learning in the averaging phase of D-PSGD. Thus, as the learning progresses and the system gets closer to reaching the consensus state, the discrepancies in loss distributions are reduced.

It is important to note that the complete opposite behaviour is exhibited in the IID setting [25], where the vulnerability increases during the training. This shows that the conclusions from experiments in IID environments may be misleading for real-world scenarios.

- **Dynamic topologies reduce the vulnerability towards the family of membership inference attacks**

From figures 2, 3 and 4, we observe that the attackers in dynamic topology achieve lower average, minimum and maximum level of advantage compared to the attackers in static topologies. If we recall that threshold attacks assess lower-bound vulnerability towards the whole family of membership inference attacks, we conclude that the dynamic topologies reduce the lower-bound vulnerability. We explain this with two reasons:

If in d-Regular graph, we pick one node i, we observe that features learned by its local model are transferred to direct neighbours with factor $\frac{1}{d}$, to first indirect neighbours with factor $\frac{1}{d^2}$, etc. In other words, the influence of model w_i on model w_j reduces exponentially with increase in the number of nodes in shortest path between nodes i and j. Thus, in static topologies, if the closest dataset containing samples from class x is k steps away from node i, then the model w_i, in each communication round, learns features of class x with factor $\frac{1}{d^k}$. On the other hand, in dynamic topologies, the value of k varies in each round. This is illustrated in Figure 1, where we observe enormous differences in accuracies on the classes {1, 2, 6} and classes {3,4, 5} obtained by the model in static graph topology, whereas the dynamic topology mitigates the differences among all classes. In other words, in static graph topologies, in the averaging step, models with the greatest factors transfer learning only for fixed set of classes that exist in the local datasets of direct neighbours, while in dynamic topologies, factors for all classes constantly change. This results in different behaviours of models trained in static and dynamic topologies when presented with unseen random samples from the global test distribution.

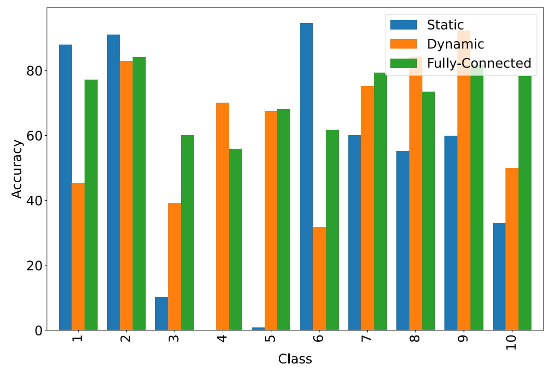


Figure 1. Class accuracies of the final models in static 4-Regular, dynamic 4-Regular and fully-connected graph topologies with 96 nodes.

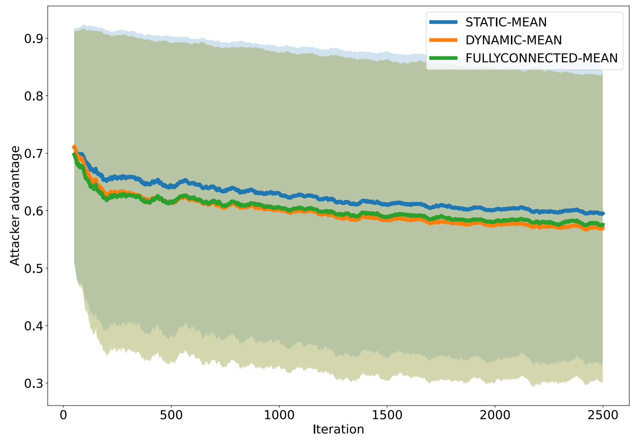
In static graphs, as a consequence of the model sharing, the influence of local model weights of node i from time step t will be distributed back by its neighbours in time step $t + 1$. This additionally increases the overfit to local dataset. On the other hand, the probability of model being distributed back in the following communication round is significantly lowered in dynamic topologies.

4 Gradient inversion attack

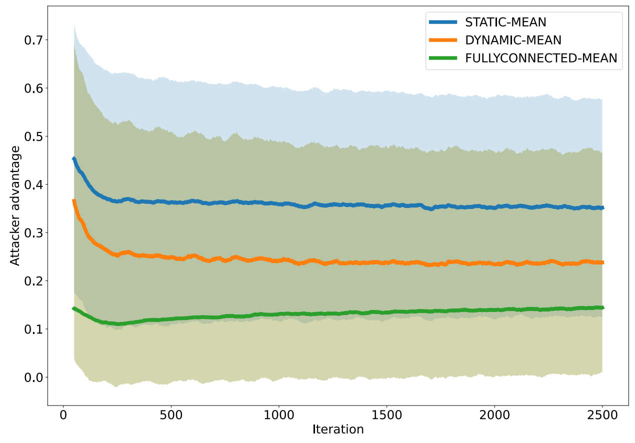
The second attack we analysed in this study is the gradient inversion attack. As in section 3, we highlight privacy guarantees offered by decentralized machine learning in static graph topologies and describe how dynamic graph topologies could be leveraged to mitigate the risks.

4.1 Definition

The goal of gradient inversion attack is to reconstruct the input data given the gradient information. Previous works [10, 13] formulated reconstruction as an optimization problem: Given model parameters θ, and the gradient $\nabla_\theta L_\theta(x^*, y^*)$

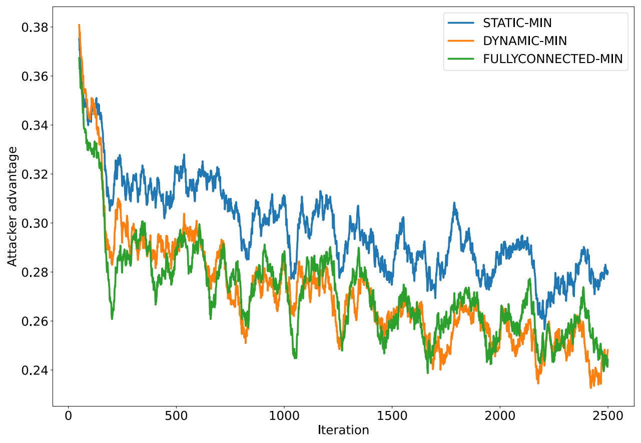


(a) Average-case model vulnerability after the *local training* phase

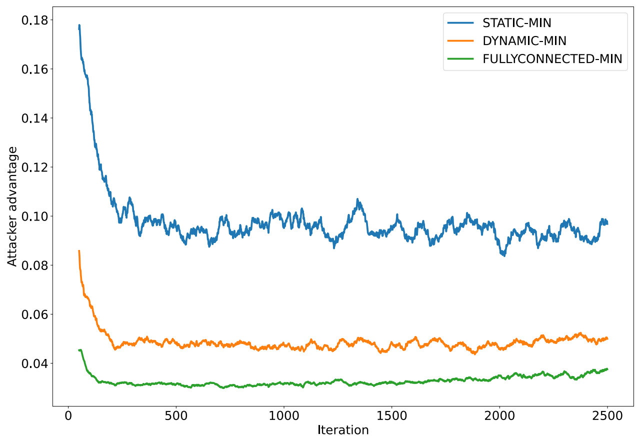


(b) Average-case model vulnerability after the *averaging* phase

Figure 2. Comparison of the *attacker advantage* metric for static 4-Regular, dynamic 4-Regular and fully-connected graph topologies with 96 nodes after the : (a) *local training* and (b) *averaging* phases of D-PSGD algorithm. Figure depicts the moving-average for the *attacker advantage* metric with window size equal to 20.



(a) Best-case model vulnerability after the *local training* phase



(b) Best-case model vulnerability after the *averaging* step

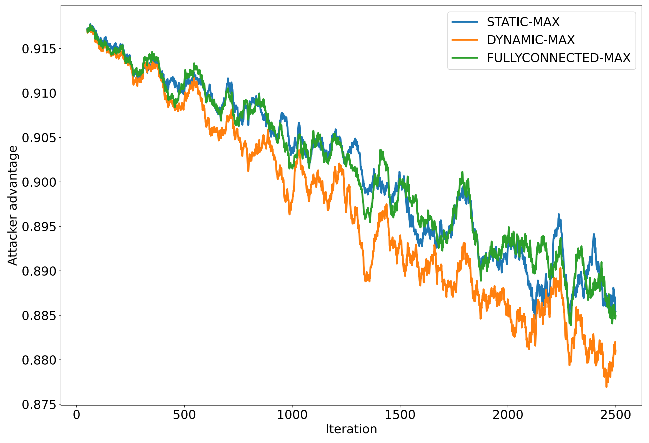
Figure 3. Comparison of the *attacker advantage* metric for static 4-Regular, dynamic 4-Regular and fully-connected graph topologies with 96 nodes after the : (a) *local training* and (b) *averaging* phases of D-PSGD algorithm. Figure depicts the moving-minimum for PSNR metric with window size equal to 20.

computed on a mini-batch of sensitive data $(x^*, y^*) \in R^{b \times d} \times R^b$ (b, d being the batch size, image size), the approximation of x^*, referred to as $x \in R^{b \times d}$:

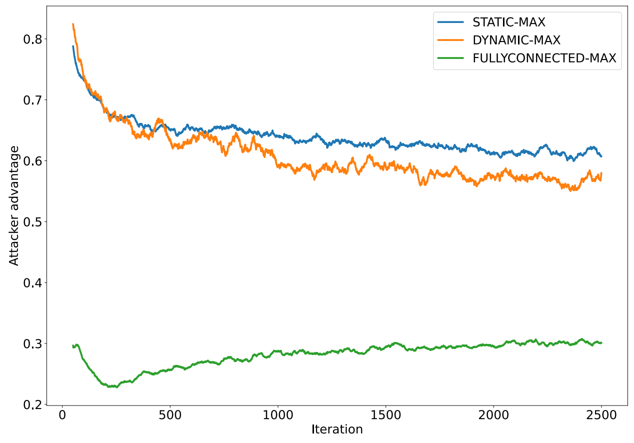
$$\underset{x}{argmin}\ L_{grad}(x; \theta, \nabla_\theta L_\theta(x^*, y^*)) + \alpha R_{aux}(x) \tag{4}$$

where $L_{grad}(x; \nabla_\theta L_\theta(x^*, y^*))$ is the cost function measuring the similarity between the ground-truth gradients $\nabla_\theta L_\theta(x^*, y^*)$ and the gradient of recovered batch x, while $R_{aux}(x)$ enforces regularization based on the image prior(s).

Gradients carry information in their magnitude and direction components. The magnitude component captures information about the training state. It is low in the areas of local minima, valleys, or stable points. On the other hand, the direction component captures the change in the prediction at a



(a) Worst-case model vulnerability after the *local training* phase



(b) Worst-case model vulnerability after the *averaging* phase

Figure 4. Comparison of the *Attacker advantage* metric for static 4-Regular, dynamic 4-Regular and fully-connected graph topologies with 96 nodes after the (a) *local training* and (b) *averaging* phases of D-PSGD algorithm. Figure depicts the moving-maximum for PSNR metric with window size equal to 20.

data point x_i, when taking the gradient step calculated using the data point x_j. This makes the direction component more suitable for exploitation in high-dimensional settings [10]. Following the study from [10], we define the optimization function 4 as:

$$\underset{x}{argmin}\ 1 - \frac{\langle \nabla_\theta L_\theta(x,y), \nabla_\theta L_\theta(x^*,y)\rangle}{\|\nabla_\theta L_\theta(x,y)\|\|\nabla_\theta L_\theta(x^*,y)\|} + \alpha TV(x) \quad (5)$$

where $\langle \cdot, \cdot \rangle$ is the inner-product of two vectors, and $TV(\cdot)$ is the total variation of images. Intuitively, this objective function finds approximations that lead to a similar change in model prediction as the unobserved ground truth.

4.2 Threat model

Similar to Section 3.2, we assume the existence of an honest-but-curious adversary. The adversary cannot deviate from the protocol, craft malicious model updates or tamper with the model architecture. However, the adversary is allowed to accumulate previously received victim's model weights in order to increase the effectiveness of the attack (more details in the section 4.3).

4.3 Experimental setup

We run our experiments in a label-skew non-IID environment. We pick one random node to be the attacker and its random neighbor as the target. The attack is performed at each communication round. In particular, we use the following setup:

Topology. We evaluate privacy in both, static and dynamic graph topologies. We utilize 4-regular (i.e., node degree set to 4) and fully-connected graph topology with 36 nodes. Moreover, the same topology is used for static and dynamic settings. However, in a dynamic setting, we randomly choose the new set of neighbors for each node at every communication round but keep the same set of cardinality.[3]

Architecture and hyperparameters. We use similar architecture and hypeparameters described in 3.3. The main differences are: the learning rate is set to 0.02, in each communication round, every node performs one mini-batch stochastic gradient descent step and the batch size is equal to 4.

Datasets. We use the FashionMNIST dataset consisting of ten evenly balanced classes. It comprises 70 000 data points, separated into global training (60 000) and global testing (10 000) sets. To establish the label-skew environment, we employ the same partitioning scheme described in section 3.3, with the only difference being that each node is given 4 shards of data.

Metrics. In order to assess the quality of reconstructed images, we leverage **Peak signal-to-noise ratio (PSNR)** metric, defined as:

$$PSNR(x, x^*) = 10 \log_{10} \frac{MAX_I^2}{MSE(x, x^*)} \quad (6)$$

where x and x^* are the reconstructed and ground-truth images, MAX_I is the maximum possible pixel value, and $MSE(\cdot, \cdot)$ is the *mean squared error.*

Note that the PSNR metric is inversely proportional to MSE, hence higher values denote better reconstruction and greater privacy vulnerability.

[3]Note here that sets of neighbours in static and dynamic settings for fully-connected graph topologies are always the same.

Gradient approximation. Recall that in the averaging step of D-PSGD (algorithm 1), nodes share model weights, not model updates (i.e., gradients). Hence, the gradients have to be approximated.

Remark: It is important to note that due to data heterogeneity, during the learning process, local models become more and more distant in space [11]. In particular, gradient descent updates are calculated using the skewed local data distribution, thus the steps may lead to a different direction for each node. If the learning algorithm, instead of model parameters, shares model updates (i.e., gradients) in the averaging step, these updates may cancel each other out. Thus, in a non-IID setting, the decentralized learning algorithm should not share model updates.

The update step of the vanilla SGD optimizer [4] is performed as follows:

$$\theta_{t+1} = \theta_t - \eta * \nabla_\theta L_\theta(x, y) \tag{7}$$

where θ_i are the model parameters at time-step i, and $\nabla_\theta L_\theta(x, y)$ are the gradients calculated on a mini-batch of samples (x,y). Later, the gradients are extracted as:

$$\nabla_\theta L_\theta(x, y) = \frac{\theta_t - \theta_{t+1}}{\eta} \tag{8}$$

It is trivial to obtain the model parameters θ_{t+1} since the target node shares its model weights in the averaging step. The tricky part is to obtain model weights θ_t. We tried two approaches. The first, and the most intuitive approach (referred to as **COMMUNICATION**) is to preserve the model weights shared by the victim in the t^{th} communication round. However, the weight-sharing step is the first part of the averaging phase in D-PSGD. In the second part, the actual averaging is performed, and the resulting model is used as the initial point for the next iteration. Thus, the gradients $\nabla_\theta L_\theta(x, y)$ are calculated with respect to the model obtained after the averaging. To address this condition, in the second approach (referred to as **POSTSTEP**), as θ_t, we use the parameters of the *local attacker's model* after the averaging phase. The motivation for this approach is that after the averaging, the influence of local overfit is reduced. In other words, neigbouring models in the averaging phase move in space towards the same point (i.e., the joint-likelihood maximization of posterior distributions), instead of moving to different directions during the local training phase (i.e., maximization of likelihood with respect to local training dataset). Additionally, if the victim and the attacker have similar sets of neighbors, we expect that the local models obtained after the averaging step are close enough for this attack to be successful.

[4]The simplicity of the update step is the main reason for choosing the vanilla SGD optimizer. It is important to note that with optimizers including momentum or adaptive learning rates (e.g., ADAM [15], AdaGRAD [5]), the approximation of gradients is much more difficult but still feasible. However, we wanted to marginalize this dimension of the problem, since it is irrelevant in our study.

Remark: Another benefit of the POSTSTEP approach is that it gives us insights in effectiveness of the *secure aggregation* [2] against gradient inversion attacks.

4.4 Static topology - Results and Discussion

Firstly, we perform experiments in the static graph topology. The results are depicted in Figure 5. We conclude that:

- **The *POSTSTEP* approach provides better gradient approximation than the *COMMUNICATION* approach.**

These results act in accordance with the motivation described in section 4.3.

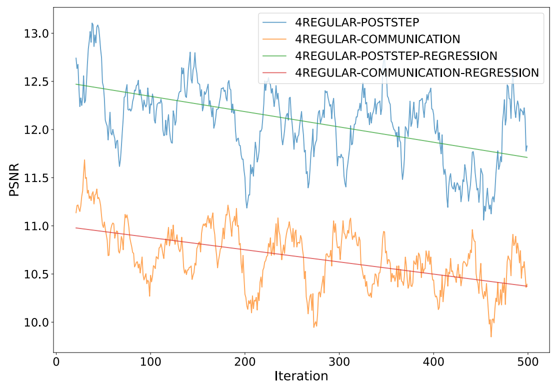
- **The average-case vulnerability decreases over time.**

Before the training begins, nodes have the same initial parameters of local models (recall from algorithm 1). This results in better gradient approximation, thus better sample reconstructions and greater vulnerability in the earliest stages of decentralized learning, with perfectly reconstructed gradients in the first communication round. The study [25], argues that the gradients are perfectly approximated whenever the consensus distance (i.e., the average, pairwise discrepancy among local models at the observed time-step) of the system is equal to 0, meaning that all nodes have the same local models. During decentralized learning, in an IID environment, the system reaches a state with consensus distance equal to 0 twice: (i) in the beginning, when nodes have the same initial model weights, and (ii) at the end of the training, when the consensus is reached. In the non-IID setting, as shown in Figure 6 and table 1, the consensus distance has not reached 0 for the second time in 4-Regular and 10-Regular graph topologies.

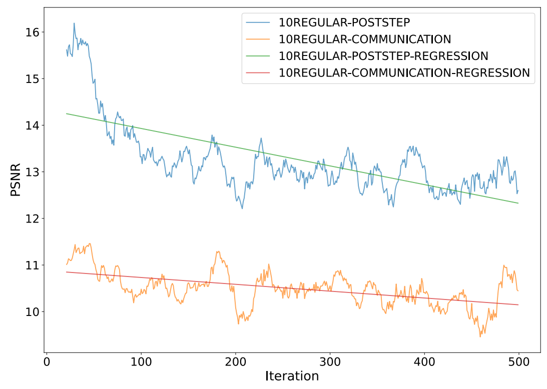
Howver, even if the system reaches the consensus state for the second time (as illustrated for the fully-connected graph topolgies in Figure 6 and table 1), we found that the training state, as a completely orthogonal dimension, influences the PNSR metric. More precisely, during the training, the characteristic features of specific classes are being emphasized in the reconstructed samples as the learning progresses. In addition, the features are observed in the reconstructions even if they were not present in the ground truth samples. This behaviour is illustrated in A.1. This has detrimental effects on PSNR metric since it is calculated using the MSE function where the difference in every pixel matters. This behavior is aligned with findings in the original paper [10]. However, in this case, the reduced PNSR metric does not imply that privacy is not breached. In order to better understand this phenomenon, more research needs to be done.

- **Dense graph topologies amplify the average-case vulnerability.**

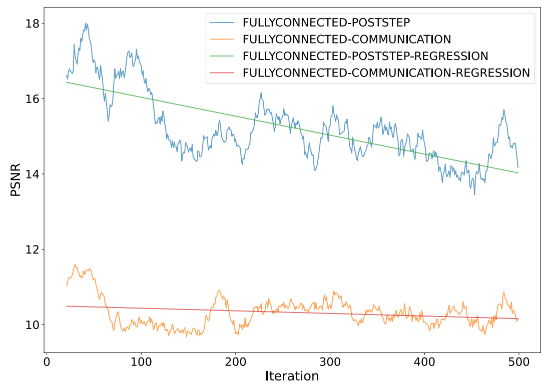
We observed that (Figure 7) with the increase in node degree of regular graph topologies, the quality of reconstructed images increases as well. More precisely, the node degree



(a) Average-case vulnerability in static 4-Regular graph



(b) Average-case vulnerability in static 10-Regular graph



(c) Average-case vulnerability in fully-connected graph

Figure 5. Comparison of the POSTSTEP and COMMUNICATION approaches in **static** graph topologies with 36 nodes. Here, we present the moving average for PSNR metric with window size equal to 10.

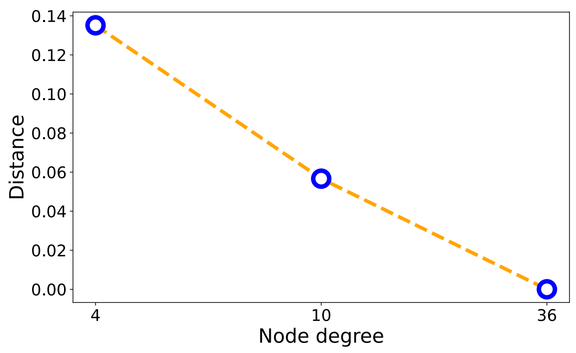


Figure 6. Euclidean distances between final models of the attacker and victim in **static** *d*-Regular graphs with 36 nodes. The node degree equal to 36 indicates the fully-connected graph.

specifies how many neighbouring models are going to be included in the averaging phase. If the aggregation is performed among more models, we expect that the consensus distance in the system is going to be smaller, i.e., models in the system are expected to be closer in space. Similar behaviour is illustrated in Figure 6, which depicts the decrease in Euclidian distance between the final attacker's and victim's models (i.e., models obtained after the completion of the training) with the increase in node degree of regular graph topologies. Henceforth, in systems with higher node degree, the quality of approximated gradients is better and average-case vulnerability increased.

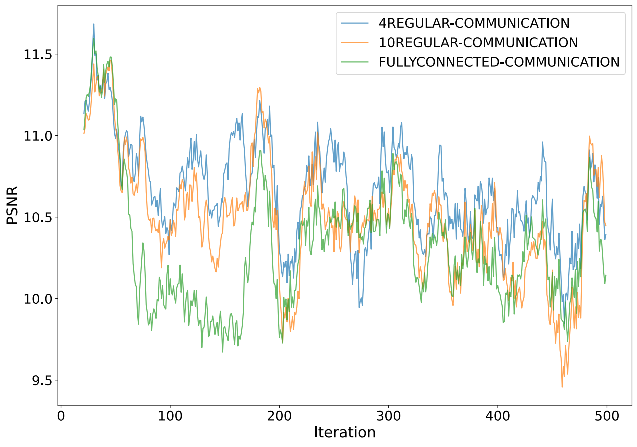
4.5 Dynamic topology - Results and Discussion

In order to examine whether randomness could mitigate this type of privacy threat, we compared the maximum levels of vulnerability across several consecutive iterations in static and dynamic topology experiments.

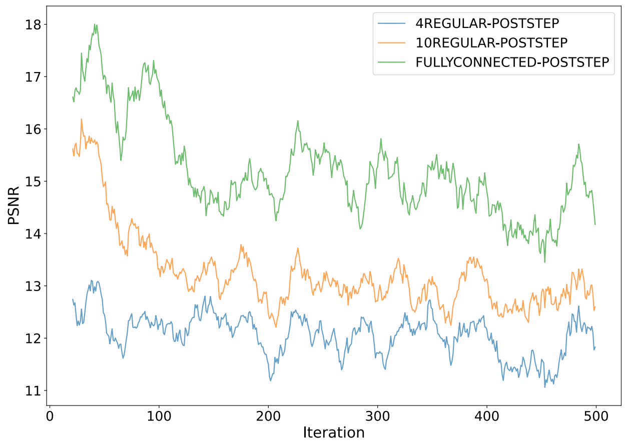
Table 1. Euclidian distances of the attacker's and victim's final local models for a 4-regular graph with 36 nodes (Static vs. Dynamic topologies).

Topology	Distance
Fully-Connected	1e-6
Static	0.13
Dynamic	**0.21**

The results are portrayed in Figure 8. Note that the peaks on the plot indicate more successful attacks. We observe significant differences in the distributions of peaks in static and dynamic cases. More precisely, recall that, in order to mount the gradient inversion attack, we had to approximate the gradients (section 4.3). The COMMUNICATION approach

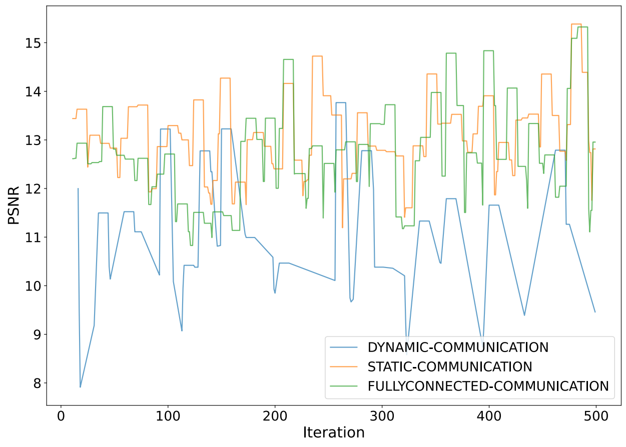


(a) COMMUNICATION approach **average-case** vulnerability

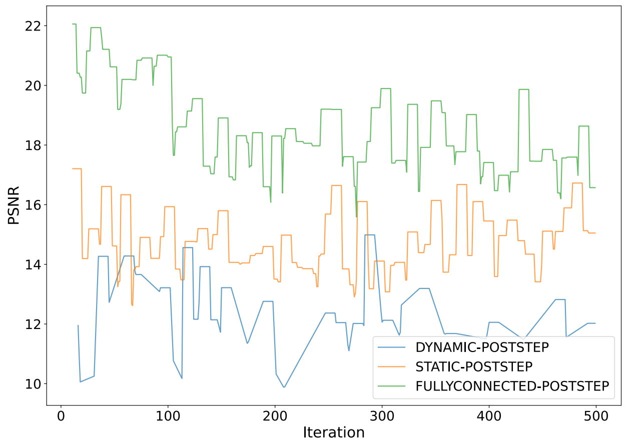


(b) POSTSTEP approach **average-case** vulnerability

Figure 7. Average-case PSNR metric comparison for **static** 4-Regular, 10-Regular and fully-connected graphs in: (a) COMMUNICATION and (b) POSTSTEP approach. Figure depicts the moving-average for PSNR metric with window size equal to 10.



(a) COMMUNICATION approach **worst-case** vulnerability



(b) POSTSTEP approach **worst-case** vulnerability

Figure 8. Worst-case PSNR metric comparison for static 4-Regular, dynamic 4-Regular and fully-connected graph topologies in: (a) COMMUNICATION and (b) POSTSTEP approaches. Figure portrays the moving-maximum for PSNR metric with window size equal to 10.

relies on the fact that in two consecutive iterations, the attacker and the victim are neighbors. However, if the set of neighbors for each node is changed at each communication round, the probability of the attacker and victim being the neighbors in two consecutive (or nearby) iterations is small. On the other hand, the POSTSTEP approach assumes that the local models of the attacker and victim, as a consequence of joint likelihood maximization, have moved towards the same point in the space. In static topology, this is always the case, since the averaging step always performs the aggregation of the same set of models. However, in dynamic topology, the models are averaged with different sets of neighbouring models in each round, thus move in different directions in space. The distances of the attacker's and victim's models obtained after the completion of the training process (table 1) show that the dynamic topology models tend to be more distant in space. This reduces the quality of approximated gradients. **Hence, we conclude that the randomness could be used to mitigate the effectiveness of the gradient inversion attacks.**

5 Related Work

In this section, we disclose related work on the topics of privacy evaluation and data heterogeneity in decentralized learning.

Privacy evaluation. The privacy guarantees are one of the biggest reasons why decentralized learning (and collaborative learning in general) is gaining importance. A large body of empirical studies has been performed to evaluate the privacy guarantees of collaborative learning. However, the vast majority of the studies is based on the topic of Federated ML [3, 9, 10, 14, 19, 24, 31].

Initial steps towards addressing the privacy issues in Decentralized ML were made by Cheng et al. [4]. In their work, Cheng et al. proposed the Leader-Follower Elastic Averaging Stochastic Gradient Descent (LEASGD) algorithm, a novel solution for achieving differential privacy [6] in a fully decentralized environment while maintaining good convergence rate and low communication cost. Unfortunately, this work does not provide actual privacy evaluation.

The first in-depth evaluation of privacy in decentralized machine learning was performed by Pasquini et al. [25]. In their work, Pasquini et al. showed that the Decentralized ML grants more capabilities for (active and passive) adversaries than the Federated ML. They showed that in assumed passive threat model exist an intrinsic trade-off that limits the achievable level of privacy in Decentralized ML. In particular, they argued that in order to mitigate the vulnerability towards the membership inference attacks (i.e., to mitigate the harmful effect of *local generalization*), the nodes in the system should increase the number of neighbours. On the other hand, increase in the number of neighbours maximizes the chances of efficient execution of gradient inversion attack (i.e., amplifies the risks caused by the *system knowledge* capability). Although provides the first comprehensive understanding of privacy issues in decentralized machine learning setups, this study assumed only the IID environment. Moreover, in section 3.4, we saw that this may not address the privacy issues in real-world scenarios.

Data heterogeneity. It has been shown that the data heterogeneity has detrimental effects on trained model quality [11], thus it became a fundamental problem of a decentralized learning. Recent works aim to address these issues by relaxing bounded variance conditions for convergence of learning algorithm [28], cross-gradient aggregations [8] or designing sparse topologies [1]. However, to the best of our knowledge, no work has been performed to evaluate the impact of heterogeneity on users privacy in Decentralized ML.

In summary, it is not well understood what are the privacy guarantees of decentralized machine learning in non-IID environments. In this paper we have performed the analysis of passive attacks in such a setting, and shown how the risks could be mitigated by employing dynamic graph topologies. However, this is a relatively new and unexplored area that necessitates a lot of further research.

6 Future Work

During our study, we encountered several questions that require further research.

The first question arose in the study of gradient inversion attacks (section 4), where we encountered the influence of the training state on PSNR metric. Thus, future work should investigate *how the current training state correlates with the level of privacy breach caused by the gradient inversion attack.*

Second question arose from the nature of our experiments. To be more precise, in our experiments, we used regular graphs with fixed sizes (i.e., number of nodes) and node degrees. Henceforth, the question that naturally arose should investigate *how privacy guarantees of the non-IID system change when we vary the total number of nodes and node degrees* in regular graphs, or use different graph topologies (e.g.,, small-world, ring).

Finally, we observed that a subset of nodes extremely fast overfit to local dataset in a label-skewed environment, which directly enhances the effectiveness of membership inference attacks. The last question should address this issue and perform more research on *how to mitigate the overfit in non-IID setting.* One approach could be to adjust the hyperparameters for each node to fit the properties of the local datasets. However, it is not clear how it would affect the system convergence.

7 Conclusion

In this paper, we performed, to the best of our knowledge, the first privacy analysis of decentralized machine learning in non-IID environment, and presented how dynamic topologies could be leveraged in mitigating the privacy risks.

In assumed passive threat model, we evaluated privacy vulnerabilities towards membership inference and gradient inversion attacks. In the first analysis, we observed that, in the label-skewed setting, models overfit extremely fast on the local datasets, which directly influences the vulnerability towards the family of MIA. In particular, we observed that the average MIA vulnerability in the system decreases over time, and argued that this is due to the fact that models generalize to the global data distribution mainly due to transfer learning performed in the averaging steps. Moreover, we have shown that the regular changes in graph topology induce better generalization, thus mitigate the average MIA vulnerability.

Second analysis evaluated the effectiveness of gradient inversion attacks. We explained two approaches for the gradient approximation and concluded that the vulnerability is the greatest in the initial communication rounds of the training. Lastly, we described how the regular changes in sets of neighbors for each node could be leveraged as a defense mechanism to mitigate the impact of the gradient inversion attack.

References

[1] Aurélien Bellet, Anne-Marie Kermarrec, and Erick Lavoie. 2022. D-cliques: Compensating for data heterogeneity with topology in decentralized federated learning. In *41st International Symposium on Reliable Distributed Systems (SRDS)*.

[2] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2016. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482* (2016).

[3] Di Cao, Shan Chang, Zhijian Lin, Guohua Liu, and Donghong Sun. 2019. Understanding distributed poisoning attack in federated learning. In *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 233–239.

[4] Hsin-Pai Cheng, Patrick Yu, Haojing Hu, Syed Zawad, Feng Yan, Shiyu Li, Hai Li, and Yiran Chen. 2019. Towards decentralized deep learning with differential privacy. In *International Conference on Cloud Computing*. Springer, 130–145.

[5] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* 12, 61 (2011), 2121–2159. http://jmlr.org/papers/v12/duchi11a.html

[6] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

[7] Anis Elgabli, Jihong Park, Amrit S Bedi, Mehdi Bennis, and Vaneet Aggarwal. 2020. Communication efficient framework for decentralized machine learning. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 1–5.

[8] Yasaman Esfandiari, Sin Yong Tan, Zhanhong Jiang, Aditya Balu, Ethan Herron, Chinmay Hegde, and Soumik Sarkar. 2021. Cross-gradient aggregation for decentralized learning from non-iid data. In *International Conference on Machine Learning*. PMLR, 3036–3046.

[9] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. 2020. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*. 1605–1622.

[10] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems* 33 (2020), 16937–16947.

[11] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. 2020. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*. PMLR, 4387–4398.

[12] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. 2022. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)* 54, 11s (2022), 1–37.

[13] Yangsibo Huang, Samyak Gupta, Zhao Song, Kai Li, and Sanjeev Arora. 2021. Evaluating gradient inversion attacks and defenses in federated learning. *Advances in Neural Information Processing Systems* 34 (2021), 7232–7241.

[14] Malhar S Jere, Tyler Farnan, and Farinaz Koushanfar. 2020. A taxonomy of attacks on federated learning. *IEEE Security & Privacy* 19, 2 (2020), 20–28.

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).

[17] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324. https://doi.org/10.1109/5.726791

[19] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37, 3 (2020), 50–60.

[20] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2019. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189* (2019).

[21] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems* 30 (2017).

[22] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. 2018. Asynchronous decentralized parallel stochastic gradient descent. In *International Conference on Machine Learning*. PMLR, 3043–3052.

[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[24] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. 2021. A survey on security and privacy of federated learning. *Future Generation Computer Systems* 115 (2021), 619–640.

[25] Dario Pasquini, Mathilde Raynal, and Carmela Troncoso. 2022. On the Privacy of Decentralized Machine Learning. *arXiv preprint arXiv:2205.08443* (2022).

[26] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. 2020. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*. PMLR, 8253–8265.

[27] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. 2019. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems* 31, 9 (2019), 3400–3413.

[28] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. 2018. D^2: Decentralized training over decentralized data. In *International Conference on Machine Learning*. PMLR, 4848–4856.

[29] Zhenheng Tang, Shaohuai Shi, and Xiaowen Chu. 2020. Communication-efficient decentralized learning with sparsification and adaptive peer selection. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 1207–1208.

[30] Paul Voigt and Axel Von dem Bussche. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10, 3152676 (2017), 10–5555.

[31] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2512–2520.

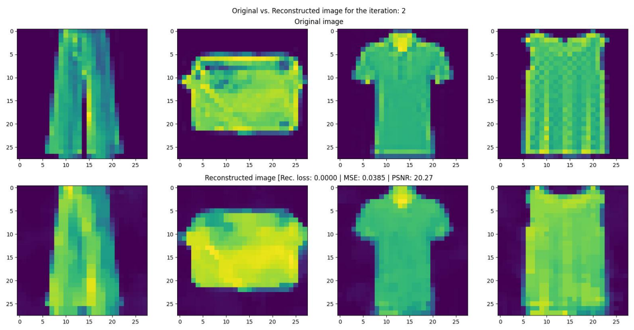
[32] Lin Xiao, Stephen Boyd, and Sanjay Lall. 2006. Distributed average consensus with time-varying metropolis weights. *Automatica* 1 (2006).

[33] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE, 268–282.

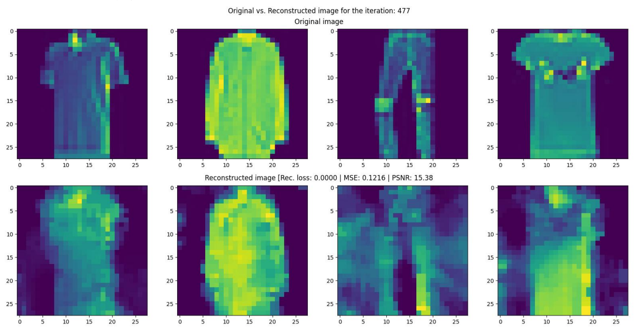
[34] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. 2021. Federated learning on non-IID data: A survey. *Neurocomputing* 465 (2021), 371–390.

A Gradient inversion attack

A.1 Examples of the reconstructed images



(a) Reconstructions from the 2nd iteration.



(b) Reconstruction from the 477th iteration.

Figure 9. The plot depicts image reconstructions from the (a) 2nd and (a) 477thiteration.

The influence of the training state on PSNR is the most evident on the last reconstruction from the 477th iteration, where we observe that the yellow color (as a characteristic feature of the class *shirt*) is highly reflected in the reconstructed, while not being represented in the original image.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009