Decentralized Federated Learning via Third-Party Bridge Collaboration

Tianning Chai[1], **Mang Ye**[1], **Wenke Huang**[1], **Chun-mei Feng**[2]

[1]Wuhan University
[2]A*STAR, Singapore

Abstract

Decentralized Federated Learning (DFL) emerges as a solution to enhance system robustness by eliminating central server in Federated Learning (FL). However, the DFL solution increases privacy risk since without a central server, a client directly exchanges with its peer and thus exposes sensitive information to the peer. Using logits and their corresponding public data shared in mutual learning process, inversion models have been developed to successfully infer private data. Privacy threats to DFL becomes more severe. We develop a novel Bridge Federated Learning (BFL) that enhances privacy preservation without sacrificing robustness of DFL. In BFL, none of the parties involved can obtain the entire necessary information to infer private data. Our BFL system delegates the logits exchange and loss calculation between the two mutual learning peers to a third-party Bridge, which is randomly chosen among the neighboring clients. We recognize that during mutual learning process, the key information needed for an inversion model to infer private data are logits and their corresponding relationship with the public data. Through introducing the Bridge, none of the parties involved have both the logits and the corresponding relationship with the public data. Our BFL achieves privacy protection without sacrificing performance or adding a significant computational burden since no extra noise is added and no additional encryption is needed. We also add a measure to improve performance in data heterogeneous scenario by utilizing client's confidence. We name this approach the Confidence Aware Mutual Learning (CAML). In our experiments, BFL method outperforms other methods by having none of the private data breached by the state-of-the-art attacks. The decentralized CAML performs competitively against FL with central server.

Introduction

Federated Learning (FL) is an important paradigm that enables collaborative learning across clients without sharing private data (Mohassel and Zhang 2017; Yang et al. 2019a; Konečnỳ et al. 2016). Centralized Federated Learning (CFL) is the most widely used form for Federated Learning. In CFL, every client is connected only to a central server. The server collects trained models from all the clients to produce a global model. The global model is then redistributed back to the clients for further training. The process continues until convergence. However, if the central sever fails, the entire system paralyzes. This critical vulnerability of the CFL is depicted as the single point of failure (Wang and Hu 2021; Qu et al. 2021).

To address this vulnerability, Decentralized Federated Learning (DFL) has emerged (Roy et al. 2019; He, Bian, and Jaggi 2018; Warnat-Herresthal et al. 2021; Wang et al. 2021; Savazzi et al. 2021). In DFL, no central server is established. The clients exchange their knowledge in a peer-to-peer manner (Beltrán et al. 2023; Chen et al. 2021). Initial implementation of a server-less DFL system utilized concise peer-to-peer networks (Roy et al. 2019; He, Bian, and Jaggi 2018). Subsequent developments have integrated blockchain technology (Warnat-Herresthal et al. 2021; Li et al. 2020; Qu et al. 2020) to increase reliability in data exchange process. However, a critical vulnerability remains. The requirement for clients to share model weights or gradients exposes the system to inversion attacks (Mothukuri et al. 2021; Yin et al. 2021; Zhu, Liu, and Han 2019). One effective solution is FedMD (Li and Wang 2019). Using logits to preform knowledge distillation (Hinton, Vinyals, and Dean 2015; Gou et al. 2021), FedMD requires only to share logits and their corresponding public data. FedMD has been considered safe (Kumar et al. 2021; Momcheva 2020; Hu et al. 2022) with low communication cost (Lyu et al. 2022). FedMD has been quickly applied to various fields (Long et al. 2021; Cheng et al. 2021; Zhou, Wang, and Wang 2022). However, recently FedMD has also been found to be susceptible to private data leakage (Takahashi, Liu, and Liu 2023). Inversion models can be trained by feeding logits and their corresponding public data. Once training is completed, inversion models can reconstruct private data from shared logits on public data only. Although differential privacy methods (Dwork 2006; Geyer, Klein, and Nabi 2017; Kalra et al. 2023; Qi, Wang, and Huang 2024), which add noise to logits, can mitigate the privacy leakage risk, they tend to compromise system performance.

We propose the novel Bridge Federated Learning (BFL) which utilizes a mechanism to cut the correlation between logits and their corresponding public data, thus makes inversion model training extremely difficult. The BFL evolves from DFL with no central server and global model, therefore is not subject to central server vulnerability. BFL is essentially a novel form of DFL. Our approach is a two-step mechanism that involves a new entity, termed "Bridge". The

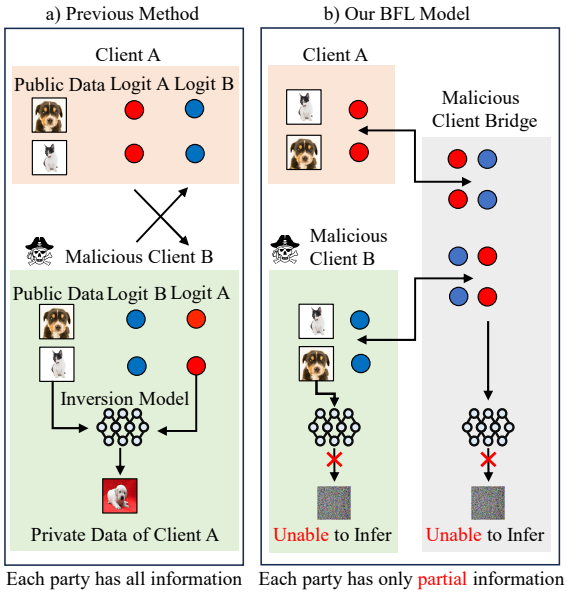

Figure 1: Illustration of the key issues in privacy protection for Federated Learning. In previous DFL systems, the two mutual learning peers directly exchange logits calculated using public data. Upon receiving logits from the other, a peer can associate the logits with their corresponding public data, and train an inversion model to infer private data of the other. In our new approach, all three parties involved have only partial information, and therefore are extremely difficult to train inversion models to infer private data of the others. We achieve this goal by introducing a third-party Bridge into the system. The Bridge serves as a hub for logits exchange and loss calculation.

Bridge is randomly chosen from the neighboring clients of the mutual learning peers. Step one, peers engaged in mutual learning (e.g. Peer A and Peer B) confidentially agree upon a random shuffled public data index to go through the public dataset. The index is only known to the mutual learning peers. Step two, Peer A and Peer B transmit their logits to the Bridge. The Bridge calculates the loss on these logit pairs and sends information back to the peers. Throughout the process, Peer A and Peer B can't obtain information on each other's specific logits, as they only receive loss information. The Bridge also remains uninformed about the correlation between logits and their public data as it doesn't have the information on the confidential index used for logit alignment.

In this approach, the three entities involved, Peer A, Peer B, and the Bridge, are either unaware of logits, or unaware of correlation between logits and their public data alignment, therefore inversion model can't be effectively trained anywhere among the three involved entities. By breaking the correlation between logits and their public data, our mechanism effectively impedes the training of inversion models.

It is necessary to point out that "Bridge" is only a temporary role. Once the learning process for an epoch is completed, the role is released. Calculating only loss for the two peers, the Bridge bears small computation burden which any average client can bear. The BFL doesn't have the computational and memory burden introduced by differential privacy or blockchains. The BFL is a robust and light weight system, and the Bridge role significantly differs from the traditional role of a central server. The BFL system as a whole does not suffer from a single point of failure such as in CFL. If a Bridge fails, it only affects a pair of peers temporarily.

In BFL, we propose a measure to ensure the integrity of a Bridge. The measure allows Peer A and Peer B to validate the loss computation done by the Bridge. Peer A and Peer B pre-agree on a set of pseudo logits and their corresponding loss. The pseudo logits are sent to the Bridge for calculation of the loss and the results are returned for validation. The validation serves as a safeguard against potential errors or malicious activities by the Bridge.

Evolved from DFL, BFL also has the challenge of data heterogeneity. Data heterogeneity is a significant challenge for FL (Kairouz et al. 2021; Huang, Ye, and Du 2022; Li et al. 2022; Onoszko et al. 2021). The performance of FL suffers greatly from the non-IID data (Zhao et al. 2018; Zhu et al. 2021; Tan et al. 2023; Ye et al. 2023). To address the issue, we propose a novel approach named Confidence Aware Mutual Learning (CAML). Mutual learning refers to a two-way knowledge distillation between a pair of peers (Zhang et al. 2018; Shen et al. 2020). In CAML, the novelty is multiplying Kullback-Leibler divergence by the confidence level of the counterpart. The confidence level is the batch average of the largest value in the counterpart logits. We assume that a better confidence level implies a more stable model against heterogeneity. By multiplying the confidence, we give more weight to more stable models, leading the system to better convergence in data heterogeneous scenarios. Empirical results prove the validness of our assumption.

To summarize, our contribution addresses three key challenges in Federated Learning:

- Enhanced System Robustness: Evolved from DFL, BFL has no central sever and no global model, thus eliminates the vulnerability associated with the central server's single point of failure.
- Advanced Privacy Protection: Correlation between logit and public data is key to train inversion models, BFL is designed to cut this correlation and make the training extremely hard. BFL effectively protects clients' private data without significantly adding capacity requirement.
- Heterogeneity Mitigation: We assume that a better confidence level implies a more stable model against heterogeneity. By utilizing confidence level in the Kullback-Leibler divergence calculation, a more stable model is given more weight. Our solution improves accuracy by more than five percent.

Related Works

Centralized and Decentralized Federated Learning

Centralized Federated Learning (CFL) is a machine learning approach that allows private data to remain on local clients,

while training a global model on a central server. The most widely used method in CFL is FedAvg (McMahan et al. 2017). At each epoch of FedAvg, clients upload their models trained on local data to a central sever. The central server then performs a weighted average to produce a global model, which is dispatched to the clients for further optimization. Based on FedAvg, a series of derivatives have been developed to address issues such as heterogeneity mitigation and privacy protection (Kalra et al. 2023; Huang, Ye, and Du 2022; Li and Wang 2019; Zhao et al. 2018). Notably, Federated Learning with Model Distillation (FedMD) has been developed to address heterogeneity mitigation (Li and Wang 2019). In FedMD, each client uses both private and public data for local training and sends their logits on public data to a central server to aggregate. The aggregated logits are sent back to the client for further distillation. Another notable derivative in CFL is to improve differential privacy (Qi, Wang, and Huang 2024; Kalra et al. 2023; Liu et al. 2021; Geyer, Klein, and Nabi 2017), by adding noise on information shared.

Despite CFL's innovative approach, a crucial vulnerability remains. If a central server fails, the whole system paralyzes (Wang and Hu 2021; Qu et al. 2021). To cope with this vulnerability, researchers have developed Decentralized Federated Learning (DFL). In DFL, no central server is established. Instead, clients exchange knowledge in a peer-to-peer manner (Roy et al. 2019; He, Bian, and Jaggi 2018). The primary advantages for peer-to-peer approaches are robustness, flexibility and scalability. However, shifting communication method from client-to-sever to client-to-client raises privacy concerns, as sensitive information is exposed to a larger group of entities. Some researchers have proposed to incorporate blockchains to DFL systems in order to improve security in data exchange process (Warnat-Herresthal et al. 2021; Li et al. 2020; Qu et al. 2020). Blockchain, in its essence, are distributed ledgers designed to record data securely and permanently. Many efforts have been done in applying blockchain technology in FL system, notably Swarm Learning (Warnat-Herresthal et al. 2021), a DFL scheme that unites edge computing, blockchain-based peer-to-peer networking and coordination. Researchers have also proposed ProxyFL (Kalra et al. 2023), which is to combine mutual learning with differential privacy and proxy model sharing for the purpose of enhancing privacy protection.

Our approach Bridge Federated Learning (BFL), is a novel Decentralized Federated Learning approach. We aim to enhance privacy protection and mitigate heterogeneity in the system. In BFL, during mutual learning process, we assign a temporary "Bridge" role to a common neighboring client of the two learning peers. The Bridge serves as a hub for logits exchange and loss calculation for the two peers. The role is released when the mutual learning is completed. Since BFL doesn't have a central server, thereby doesn't have the vulnerability issues created by a central server. BFL is essentially a DFL.

Deep Mutual Learning

Deep Mutual Learning (DML) is an extension of knowledge distillation (Zhang et al. 2018). Knowledge distillation (Hinton, Vinyals, and Dean 2015) is a technique to transfer knowledge, typically aligning the logits output of a small student model to a big teacher model. DML shifts from a one-way teacher to student knowledge transfer to a two-way peer to peer knowledge transfer since the models align their logits with each other.

Inversion Attacks in Federated Learning

In Centralized or Decentralized Federated Learning, whether between server and client or between client and client, information must be shared to allow mutual learning to proceed. On a client, it typically resides public data, as well as private data which is only visible to the individual client. Even though a client never shares its private data, it still needs to train its model on both public and private data. Inversion attacks can reconstruct a client's private data from the information shared. Initial developments in the inversion attacks focused on reconstructing private data from shared gradients or parameters (Mothukuri et al. 2021; Yin et al. 2021; Zhu, Liu, and Han 2019). Therefore, researchers have developed new mechanisms where only output logits and their corresponding public data are shared for distillation, notably FedMD, FedGEMS (Cheng et al. 2021) and DS-FL (Itahara et al. 2021). Even though sharing logits on public data is safer than directly sharing gradients or parameters, there still exists substantial risk of private data leakage, as the shared logits are generated by the clients' neural network trained with both public and private data. Malicious attacks have been developed to use inversion neural networks to dig up private knowledge contained in the logits on public data, notably the PLI (Takahashi, Liu, and Liu 2023) and TBI (Yang et al. 2019b). We find out that in order to effectively train inversion networks, there must present simultaneously two pieces of information: logit and logit-data corresponding relationship. Therefore, we design our BFL mechanism in a decentralized setting and no client, either peers or Bridges, in our system can simultaneously have two pieces of information, thereby training an inversion model is extremely difficult. Our experiment validates our findings.

Methodology

Bridge for Mutual Learning

Typical mutual learning process requires the participants to share logits with each other to align their models. However, directly sharing logits can create opportunities for inversion attacks, since a malicious client can associate the shared logits with their specific public data (Takahashi, Liu, and Liu 2023). We delegate the calculation of logits' KL divergence to a third-party Bridge, therefore the peers don't need to exchange logits directly. Crucially, the Bridge doesn't know the correspondence between logits and their public data.

In our method, Peer A and Peer B traverse the samples in public dataset $\mathcal{D}$, by a randomly shuffled sequence π, represented as $\mathcal{D}_\pi = \{x_{\pi(i)} | x_i \in \mathcal{D}\}$. The randomly shuffled sequence must be deranged, meaning that no element in the sequence remains in its original position after the shuffle. The default sequence π may be shuffled by either Peer A or Peer B, and the result is known only by the two of them. A

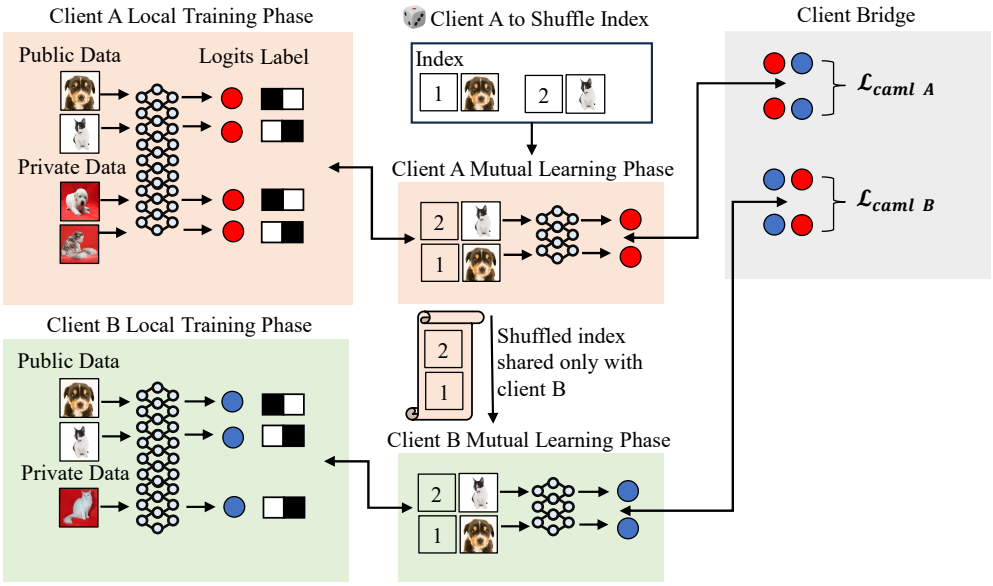

Figure 2: Illustration of our novel Bridge mechanism. In our new mechanism, the three entities involved in mutual learning Peer A, Peer B and Client Bridge each only has partial information, thus an inversion model can't be trained effectively on any one of them. The mutual learning peers Peer A and Peer B first train locally on both public and private data. Peer A shuffles the public data index and shares only with Peer B. Peer A and Peer B generate logits through the public data with the confidential index and send only the logits to the Bridge. The Bridge calculates and returns the CAML loss for Peer A and Peer B $\mathcal{L}_{CAML_A}$ to A, $\mathcal{L}_{CAML_B}$ to B (see Eq.(6)). The Bridge is a temporary role and can simultaneously perform mutual learning with its own peer. Our Decentralized Federated Learning system allows all peers to conduct mutual learning at the same time.

malicious client can't infer private data from only logits. It must simultaneously have the information on logits and logits' corresponding public data. With the logits batches sending in randomly shuffled sequence to the Bridge, the Bridge can't associate logits with their corresponding public data. At the same time, Peer A and Peer B don't have information on each other's logits. Thereby, none of the parties involved in mutual learning can have the entire information on logits and their corresponding public data, thus inversion models are difficult to train.

As KL divergence is asymmetric between Peer A and Peer B, the Bridge calculates KL loss separately for Peer A and Peer B. Upon receiving the computed KL divergence from the Bridge, Peer A and Peer B calculate their total loss including their own cross-entropy loss with the labels:

$$\mathcal{L}_A = \mathcal{L}_{CE_A} + \alpha \sum_{i=1}^{M} P_A(i) \log \left(\frac{P_A(i)}{P_B(i)} \right), \quad (1)$$

$$\mathcal{L}_B = \mathcal{L}_{CE_B} + \alpha \sum_{i=1}^{M} P_B(i) \log \left(\frac{P_B(i)}{P_A(i)} \right). \quad (2)$$

In later sections, we will introduce how we multiply the confidence factor with the KL divergence to mitigate heterogeneity. Back propagation is then performed by Peer A and Peer B to update their model parameters, using gradient descent:

$$\theta_A \leftarrow \theta_A - \eta \cdot \nabla_{\theta_A} \mathcal{L}_A, \quad (3)$$

$$\theta_B \leftarrow \theta_B - \eta \cdot \nabla_{\theta_B} \mathcal{L}_B. \quad (4)$$

At this point, Peer A and Peer B don't have information on each other's logits, and the Bridge can't link the logits to their corresponding public data. Therefore, inversion models are difficult to train by any party involved in mutual learning and our BFL system achieves a high level of confidentiality.

Our method strikes a balance between efficient knowledge sharing and privacy preservation, and it is a robust solution for practical environments.

Pseudo Logits Batches to Detect Malicious Bridge

To enhance the security of our Bridge Protocol, we introduce a mechanism to detect malicious behavior of a Bridge. With 10 random numbers in the range of 1 to 20, pseudo logits are generated and normalized by either Peer A or Peer B. These pseudo logits are paired and shared between Peer A and Peer B. The loss of each pair is calculated and stored on both Peer A and Peer B. The pseudo logits are mixed with the regular ones and sent to the Bridge altogether. The Bridge returns calculated loss of all logits. The loss of the pseudo logits calculated by the Bridge are compared with those stored previously on Peer A and Peer B, any significant deviation is an indicator of potential malicious behaviors by the Bridge. The loss calculated from the pseudo logit pairs are not used in the backward propagation of the model.

We can calculate the probability of guessing the confidential index correctly with pseudo logits added. Given N, the number of samples in public dataset, and γ, the

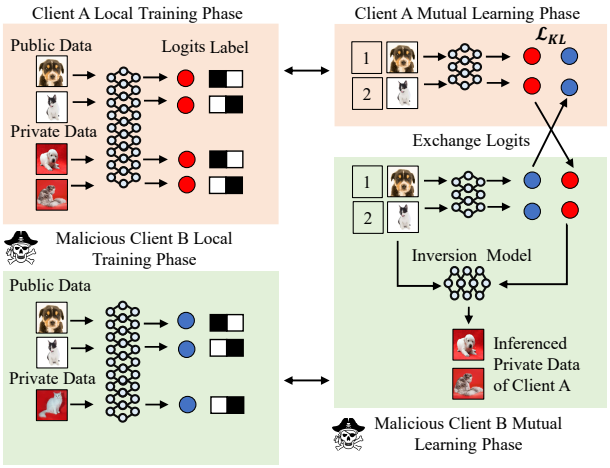

Figure 3: Illustration of previous mutual learning system in FL. In previous system, peers exchange information on logits and their corresponding public data directly. Using exchanged information, a malicious peer can train its inversion model and reconstruct private data of the other.

pseudo logits generation rate, the total number of logits is $N_{\text{total}} = N + \gamma \cdot N$. The number of ways to choose N logits from N_{total} is $\binom{N+\gamma \cdot N}{N}$, and there are $N!$ (N factorial) ways to arrange these N chosen logits. Therefore, the total number of possible arrangements is $\binom{N+\gamma \cdot N}{N} \cdot N!$, and the probability of guessing the correct arrangement randomly is the reciprocal of this number:

$$P_{correct} = \frac{1}{\binom{N+\gamma \cdot N}{N} \cdot N!}. \tag{5}$$

Measures on Selecting Bridge and Peers

Bridge selection is random for security purpose. Peer A selects randomly from its non-malicious neighbors which are identified through the earlier pseudo logit tests.

Peer selection is also random. When the performance levels of clients are at the similar level, random selection is advantageous as it reduces the risk of engaging with hostile peers. Previous research, notably the PENS (Onoszko et al. 2021), suggests that selecting high-performance peers can enhance the system's overall performance. However, PENS requires potential peers to send their models to the other client to test performance on the other client's private data, posing substantial privacy risks and leading to efficiency degradation. Our framework adopts a random selection approach for peer assignment. Random peer selection is a widely accepted practice in decentralized scenarios (Beltrán et al. 2023).

Confidence Aware Mutual Learning

In Federated Learning, a pivotal challenge arises from data heterogeneity across the distributed network of clients. The heterogeneity stems from the variations in data distribution among different participants, and can markedly impair the effectiveness of mutual learning strategies. More precisely, in a standard federated learning setup with K participants (indexed by i), each participant possesses a local model θ_i and a private dataset $D_i = \{(X_i, Y_i) | X_i \in R^{N_i \times D}, Y_i \in R^{N_i \times C}\}$, where N_i denotes the number of data samples, D is the input size, and C represents the number of classes for classification tasks. The private data distribution for each participant is denoted as $P_i(X, Y)$ and can be decomposed as $P_i(X|Y)P_i(Y)$.

Crucially, in heterogeneous federated learning environments, data heterogeneity is defined by the condition $P_i(X|Y) \neq P_j(X|Y)$ for any pair of participants i and j, which implies that the training data of the clients is non-independently and identically distributed, even if the marginal distribution $P(Y)$ is consistent across participants, and the conditional distribution $P(X|Y)$ varies, signifying that the same label Y may correspond to different features X for different clients (Huang, Ye, and Du 2022). Such non-IID nature of data across clients introduces significant complexities to the learning process, demanding advanced strategies to ensure effective mutual learning under the challenge of disparities.

To address the challenge of data heterogeneity, we propose the novel Confidence Aware Mutual Learning (CAML) and integrate it into our BFL. CAML leverages the confidence levels in the predictions of each client's model to modulate their influence in the learning process. Specifically, the confidence is quantified as the maximum logit value output by a model for a given sample. The rationale behind this approach is that a model's high confidence in its prediction suggests a better alignment of its training data with the given sample. Therefore, weighting the contributions based on confidence can mitigate the effects of data heterogeneity, allowing for more effective knowledge sharing and model generalization.

Consider a collaborative learning scenario between Peer A and Peer B, Peer A's loss, denoted as $\mathcal{L}_{CAMLA}$, is determined by the Kullback-Leibler divergence between the softmax probabilities of Logits A and Logits B. The loss is further weighted by a scaling factor α and the batch-wise average of the maximum elements within the logits of Peer B. The formula is expressed as:

$$\mathcal{L}_{CAML_A} = \alpha \sum_{i=1}^{M} P_A(i) \log \left(\frac{P_A(i)}{P_B(i)} \right) \cdot \frac{1}{N} \sum_{i=1}^{N} \max(L_B^{(i)}). \tag{6}$$

$P_A(i)$ and $P_B(i)$ represent the softmax probabilities of the i-th element in the Logits of Peer A and Peer B, respectively. The term $\max(L_B^{(i)})$ denotes the maximum element in the logits of Peer B for the i-th instance in the batch. M corresponds to the total number of classes in the classification problem and N corresponds to the batch size. Our novelty is to multiply the confidence factor:

$$\frac{1}{N} \sum_{i=1}^{N} \max(L_B^{(i)}). \tag{7}$$

We then take $\mathcal{L}_{CAML}$ as part of our total loss $\mathcal{L}_A$ for the

Peer A, adding the cross entropy loss aligning the logits A and the labels:

$$\mathcal{L}_A = \mathcal{L}_{CE_A} + \alpha \sum_{i=1}^{M} P_A(i) \log \left(\frac{P_A(i)}{P_B(i)} \right) \cdot \frac{1}{N} \sum_{i=1}^{N} \max(L_B^{(i)}), \tag{8}$$

and we take the symmetric version for the total loss $\mathcal{L}_B$ for the Peer B:

$$\mathcal{L}_B = \mathcal{L}_{CE_B} + \alpha \sum_{i=1}^{M} P_B(i) \log \left(\frac{P_B(i)}{P_A(i)} \right) \cdot \frac{1}{N} \sum_{i=1}^{N} \max(L_A^{(i)}). \tag{9}$$

By integrating CAML into the system, we ensure that our BFL is efficient and effective in environments characterized by significant data heterogeneity.

Experiments

Experiments on Inversion Attacks

Datasets: We use the Large-Age-Gap dataset (Bianco 2017) and the Masked LFW dataset (Phan and Nguyen 2022). The LAG dataset contains pairs of youth and adult images of the same celebrities, highlighting the age gap. The Masked LFW dataset is a modified version of the popular Labeled Faces in the Wild dataset (Huang and Learned-Miller 2014) where face images are augmented with synthetic masks.

Breaching by the State-of-the-Art Attacks: We test the validity of our method with the state-of-the-art Paired Logits Inversion attack (Takahashi, Liu, and Liu 2023) and the Training Based Inversion attack (Yang et al. 2019b). The Paired Logits Inversion attack leverages the distribution gap between the logits from the targeted client and those from the malicious server. The targeted client's logits reflect knowledge from both public and private data, as its model is trained with both. In contrast, the server's model is trained with only public data. During training, the inversion neural network trained by the server is optimized to reconstruct the public data that corresponds to the given logits, exploiting the distribution gap. Once trained, when provided with prior estimations, the inversion neural network is capable of reconstructing the private data. In DFL, where logits are shared peer-to-peer, any client can act malicious. TBI is another gradient-free attack, it doesn't require the targeted client's gradients to reconstruct private data. The inversion network of TBI is trained with an auxiliary dataset. We compare our method with FedMD (Li and Wang 2019), FedGEMS (Cheng et al. 2021) and DS-FL (Itahara et al. 2021), which are important Federated Learning paradigms that utilizes knowledge distillation.

We compare the number of successful private data breaches across different FL paradigms. The definition of a successful private data breach is:

$$\text{SSIM}(\hat{x}_j, x_j) > \max(\text{SSIM}(\hat{x}_j, x_{-j}), \text{SSIM}(\hat{x}_j, x_p)). \tag{10}$$

$\hat{x}_j$ is the reconstructed private image of label j by the inversion network, x_j is the averaged private image of label j, x_{-j} is the averaged private image of each label except j and x_p is the averaged public image of any label. SSIM evaluates the similarity between images (Wang et al. 2004). This definition of successful breaches is the same as in the PLI attack study (Takahashi, Liu, and Liu 2023).

Implementation Details: The experiments are run on 8 NVIDIA GeForce RTX 4090 GPUs, each GPU has 24564 MB of memory. We use Adam (Kingma and Ba 2014) as the optimizer to train the inversion network, with a momentum of 0.1, a learning rate of 0.001 and a batch size of 8. Epochs of inversion model training for LAG is 3 and for LFW is 1. Images are resized to 64×64 and normalized to $[-1, 1]$. The random seed for LAG is 42 and for LFW is 1211.

Algorithm 1: FL with Bridge Protocol and CAML

Input: Public Dataset $\mathcal{D}$, number of clients N, number of local epochs E, communication rounds T, Learning rate η.
Output: θ_A

for each round $t = 1$ to T **do**
  for each client A $= 1$ to N in parallel **do**
    A picks the Bridge and Peer B.
    A generates new index π for $\mathcal{D}$ and sends it to B.
    for each local epoch $e = 1$ to E **do**
      A and B send logits indexed by π to Bridge.
      Bridge computes $\mathcal{L}_{CAML_A}$, $\mathcal{L}_{CAML_B}$ in Eq.(6).
      Bridge sends $\mathcal{L}_{CAML_A}$ to A.
      Bridge sends $\mathcal{L}_{CAML_B}$ to B.
      A computes $\mathcal{L}_{CE_A}$ and gets $\mathcal{L}_A$ in Eq.(8).
      B computes $\mathcal{L}_{CE_B}$ and gets $\mathcal{L}_B$ in Eq.(9).
      $\theta_A \leftarrow \theta_A - \eta \cdot \nabla_{\theta_A} \mathcal{L}_A$ in Eq.(3).
      $\theta_B \leftarrow \theta_B - \eta \cdot \nabla_{\theta_B} \mathcal{L}_B$ in Eq.(4).
    end for
  end for
  Adjust η throughout epochs.
end for
return θ_A

Experiments on Data Heterogeneity

Datasets: We use Dirichlet distribution with different concentration parameter β to simulate data heterogeneity (Lin et al. 2020; Wang et al. 2020a,b; Li, He, and Song 2021). The following datasets are used: Fashion-MNIST (Xiao, Rasul, and Vollgraf 2017), SVHN (Netzer et al. 2011), CIFAR-10 (Krizhevsky, Hinton et al. 2009) and MNIST (LeCun 1998).

Comparison with State-of-the-Art Methods: We compare our results with various methods. FedAvg (McMahan et al. 2017) is a Centralized Federated Learning method where all clients train models on public data and private data, and a central server collects and aggregates model parameters from all clients to form a global model. ProxyFL (Kalra et al. 2023) is a Decentralized Federated Learning method with differential privacy and mutual learning on proxy models.

Implementation Details: Our method is trained using pytorch on 4 NVIDIA GeForce RTX 3090 GPUs, each GPU has 24576 MB of memory. We use Adam as the optimizer, with a momentum of 0.3 and a batch size of 32. The initial learning rate is 0.01, which is reduced by a factor of 0.1 every 15 epochs. Images are resized to 32×32. Our work is done in the assumption that the network is fully connected, meaning that a client can reach any other client in the network. The random seed is 1211.

	CIFAR-10			Fashion-MNIST			MNIST			SVHN		
	$\beta = 1$	$\beta = 5$	$\beta = 10$	$\beta = 1$	$\beta = 5$	$\beta = 10$	$\beta = 1$	$\beta = 5$	$\beta = 10$	$\beta = 1$	$\beta = 5$	$\beta = 10$
FedAvg	0.5536	0.5558	0.5547	0.8604	0.8602	0.8606	0.9750	0.9780	0.9772	0.8454	0.8430	0.8454
FedMD	0.5372	0.5421	0.5384	0.8575	0.8588	0.8595	0.9757	0.9768	0.9761	0.8418	0.8444	0.8435
ProxyFL	0.5230	0.5278	0.5353	0.8417	0.8465	0.8464	0.9705	0.9716	0.9732	0.8172	0.8264	0.8256
Ours	0.5269	0.5325	0.5397	0.8570	**0.8607**	0.8596	**0.9759**	0.9753	0.9760	0.8425	**0.8475**	0.8452

Table 1: Comparison with the stat-of-the-art methods in data heterogeneous settings on the average accuracy of all clients. β stands for the concentration parameter of Dirichlet distribution. FedAvg and FedMD are centralized methods that depend on a central server. Ours and ProxyFL are decentralized methods. Typically, it is difficult for DFL methods to surpass CFL methods in performance as CFL methods have servers that aggregate knowledge from all of the clients while DFL methods are only peer to peer. However, our method still manages to surpass both CFL and other DFL methods in some tasks.

	LAG		LFW	
	PLI	TBI	PLI	TBI
FedMD	68.5%	0.0%	88.5%	4.5%
FedGEMS	26.5%	0.0%	70.5%	3.5%
DS-FL	22.0%	4.0%	36.0%	16.0%
Ours	**0.0%**	**0.0%**	**0.0%**	**0.0%**

Table 2: Comparison with the state-of-the-art FL methods on successful breaches of private data in percentage (e.g. 68.5% of FedMD's tested private data has been breached by the PLI attack). PLI and TBI are cutting edge attack methods. LAG and LFW are two widely used datasets in the field of privacy protection.

Dataset	$\beta = 1$	$\beta = 5$	$\beta = 10$
CIFAR-10	0.5399	0.5431	0.5381
Fashion-MNIST	0.8568	0.8595	0.8591
MNIST	0.9770	0.9775	0.9758
SVHN	0.8462	0.8458	0.8498

Table 3: Results of Ablation Study. We use the standard KL loss in Eq.(1) as the mutual learning loss instead of our CAML loss.

Ablation Studies

For more rigorous examination, we also conduct ablation studies which are done by employing KL loss instead of CAML loss in BFL Protocol (see Eq.(1)). Ablation is essentially Deep Mutual Learning under BFL Protocol. When realizing our approach, we let the models first train solo to almost reach convergence, then start the BFL Protocol. We take the same approach in Ablation.

Discussion

Our method successfully protects clients' private data from the state-of-the-art inversion attacks. Previous privacy protection methods typically require more computing resources while our method only puts a negligible burden to calculate pseudo logits batches. We accomplish privacy protection without adding noise or encrypting data. Rather, we recognize that two pieces of information logits and logits-public data correlation must simultaneously present in order to effectively train an inversion model. We design a new structure Bridge so that no entity can simultaneously have both information. This is a new idea for privacy protection in the field of machine learning.

In the LAG and LFW dataset, our method outperforms other methods by having none of the private data breached by the two state-of-the-art attacks.

In the settings of data heterogeneity, our method has demonstrated its competitive performance against centralized methods which gathers much more knowledge. Our method also exhibits strong robustness and privacy preservation qualities inherent in its design. The reason why the ablation method has lower performance than BFL is that the optimization directions of different models differ greatly in non-IID scenarios. Without giving KL an adaptive confidence weighting, mutual learning may harm the models by giving them undesirable gradient descent directions. Furthermore, if we set $\beta = 0.1$ and reduce the amount of public data to one-tenth of its original size to test more extreme cases, the standard deviation of performance among the clients in the ablation study is 0.0546, while ours is 0.046. Our method can better reduce performance variability across clients, even in extreme non-IID scenarios.

Conclusion

In our novel BFL scheme, no client can simultaneously have the knowledge on others' logits and logits-public data correlation, thereby inversion models are extremely difficult to train. We also propose Confidence Aware Mutual Learning to make mutual learning heterogeneity tolerant. The effectiveness of our approach is supported by the experiments. BFL represents a transformation in the way Federated Learning systems handle system robustness and data privacy without sacrificing performance or adding significant computational burden. Our work is a creative example of how Federated Learning can evolve to meet the complex demands of system robustness and data privacy in the data-centric era.

References

Beltrán, E. T. M.; Pérez, M. Q.; Sánchez, P. M. S.; Bernal, S. L.; Bovet, G.; Pérez, M. G.; Pérez, G. M.; and Celdrán, A. H. 2023. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials*.

Bianco, S. 2017. Large age-gap face verification by feature injection in deep networks. *Pattern Recognition Letters*, 90: 36–42.

Chen, J.-H.; Chen, M.-R.; Zeng, G.-Q.; and Weng, J.-S. 2021. BDFL: A byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle. *IEEE Transactions on Vehicular Technology*, 70(9): 8639–8652.

Cheng, S.; Wu, J.; Xiao, Y.; and Liu, Y. 2021. Fedgems: Federated learning of larger server models via selective knowledge fusion. *arXiv preprint arXiv:2110.11027*.

Dwork, C. 2006. Differential privacy. In *International colloquium on automata, languages, and programming*, 1–12. Springer.

Geyer, R. C.; Klein, T.; and Nabi, M. 2017. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6): 1789–1819.

He, L.; Bian, A.; and Jaggi, M. 2018. Cola: Decentralized linear learning. *Advances in Neural Information Processing Systems*, 31.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hu, H.; Salcic, Z.; Sun, L.; Dobbie, G.; Yu, P. S.; and Zhang, X. 2022. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s): 1–37.

Huang, G. B.; and Learned-Miller, E. 2014. Labeled faces in the wild: Updates and new reporting procedures. *Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep*, 14(003).

Huang, W.; Ye, M.; and Du, B. 2022. Learn from others and be yourself in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10143–10153.

Itahara, S.; Nishio, T.; Koda, Y.; Morikura, M.; and Yamamoto, K. 2021. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *IEEE Transactions on Mobile Computing*, 22(1): 191–205.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2): 1–210.

Kalra, S.; Wen, J.; Cresswell, J. C.; Volkovs, M.; and Tizhoosh, H. 2023. Decentralized federated learning through proxy model sharing. *Nature communications*, 14(1): 2899.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Konečnỳ, J.; McMahan, H. B.; Ramage, D.; and Richtárik, P. 2016. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kumar, A.; Purohit, V.; Bharti, V.; Singh, R.; and Singh, S. K. 2021. MediSecFed: Private and secure medical image classification in the presence of malicious clients. *IEEE Transactions on Industrial Informatics*, 18(8): 5648–5657.

LeCun, Y. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Li, D.; and Wang, J. 2019. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*.

Li, Q.; Diao, Y.; Chen, Q.; and He, B. 2022. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th international conference on data engineering (ICDE)*, 965–978. IEEE.

Li, Q.; He, B.; and Song, D. 2021. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10713–10722.

Li, Y.; Chen, C.; Liu, N.; Huang, H.; Zheng, Z.; and Yan, Q. 2020. A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network*, 35(1): 234–241.

Lin, T.; Kong, L.; Stich, S. U.; and Jaggi, M. 2020. Ensemble distillation for robust model fusion in federated learning. *Advances in neural information processing systems*, 33: 2351–2363.

Liu, R.; Cao, Y.; Chen, H.; Guo, R.; and Yoshikawa, M. 2021. Flame: Differentially private federated learning in the shuffle model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 8688–8696.

Long, G.; Shen, T.; Tan, Y.; Gerrard, L.; Clarke, A.; and Jiang, J. 2021. Federated learning for privacy-preserving open innovation future on digital health. In *Humanity driven AI: productivity, well-being, sustainability and partnership*, 113–133. Springer.

Lyu, L.; Yu, H.; Ma, X.; Chen, C.; Sun, L.; Zhao, J.; Yang, Q.; and Philip, S. Y. 2022. Privacy and robustness in federated learning: Attacks and defenses. *IEEE transactions on neural networks and learning systems*.

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

Mohassel, P.; and Zhang, Y. 2017. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE symposium on security and privacy (SP)*, 19–38. IEEE.

Momcheva, P. T. G. 2020. Sentiment detection with fedmd: Federated learning via model distillation. *Information Systems and Grid Technologies*, 1.

Mothukuri, V.; Parizi, R. M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; and Srivastava, G. 2021. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115: 619–640.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A. Y.; et al. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, 4. Granada.

Onoszko, N.; Karlsson, G.; Mogren, O.; and Zec, E. L. 2021. Decentralized federated learning of deep neural networks on non-iid data. *arXiv preprint arXiv:2107.08517*.

Phan, H.; and Nguyen, A. 2022. DeepFace-EMD: Re-ranking using patch-wise earth mover's distance improves out-of-distribution face identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20259–20269.

Qi, T.; Wang, H.; and Huang, Y. 2024. Towards the Robustness of Differentially Private Federated Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 19911–19919.

Qu, Y.; Dai, H.; Zhuang, Y.; Chen, J.; Dong, C.; Wu, F.; and Guo, S. 2021. Decentralized federated learning for UAV networks: Architecture, challenges, and opportunities. *IEEE Network*, 35(6): 156–162.

Qu, Y.; Gao, L.; Luan, T. H.; Xiang, Y.; Yu, S.; Li, B.; and Zheng, G. 2020. Decentralized privacy using blockchain-enabled federated learning in fog computing. *IEEE Internet of Things Journal*, 7(6): 5171–5183.

Roy, A. G.; Siddiqui, S.; Pölsterl, S.; Navab, N.; and Wachinger, C. 2019. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv preprint arXiv:1905.06731*.

Savazzi, S.; Nicoli, M.; Bennis, M.; Kianoush, S.; and Barbieri, L. 2021. Opportunities of federated learning in connected, cooperative, and automated industrial systems. *IEEE Communications Magazine*, 59(2): 16–21.

Shen, T.; Zhang, J.; Jia, X.; Zhang, F.; Huang, G.; Zhou, P.; Kuang, K.; Wu, F.; and Wu, C. 2020. Federated mutual learning. *arXiv preprint arXiv:2006.16765*.

Takahashi, H.; Liu, J.; and Liu, Y. 2023. Breaching FedMD: Image Recovery via Paired-Logits Inversion Attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12198–12207.

Tan, Y.; Liu, Y.; Long, G.; Jiang, J.; Lu, Q.; and Zhang, C. 2023. Federated learning on non-iid graphs via structural knowledge sharing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 9953–9961.

Wang, H.; Yurochkin, M.; Sun, Y.; Papailiopoulos, D.; and Khazaeni, Y. 2020a. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*.

Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; and Poor, H. V. 2020b. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33: 7611–7623.

Wang, T.; Liu, Y.; Zheng, X.; Dai, H.-N.; Jia, W.; and Xie, M. 2021. Edge-based communication optimization for distributed federated learning. *IEEE Transactions on Network Science and Engineering*, 9(4): 2015–2024.

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4): 600–612.

Wang, Z.; and Hu, Q. 2021. Blockchain-based federated learning: A comprehensive survey. *arXiv preprint arXiv:2110.02182*.

Warnat-Herresthal, S.; Schultze, H.; Shastry, K. L.; Manamohan, S.; Mukherjee, S.; Garg, V.; Sarveswara, R.; Händler, K.; Pickkers, P.; Aziz, N. A.; et al. 2021. Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862): 265–270.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019a. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2): 1–19.

Yang, Z.; Zhang, J.; Chang, E.-C.; and Liang, Z. 2019b. Neural network inversion in adversarial setting via background knowledge alignment. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 225–240.

Ye, M.; Fang, X.; Du, B.; Yuen, P. C.; and Tao, D. 2023. Heterogeneous federated learning: State-of-the-art and research challenges. *ACM Computing Surveys*, 56(3): 1–44.

Yin, H.; Mallya, A.; Vahdat, A.; Alvarez, J. M.; Kautz, J.; and Molchanov, P. 2021. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 16337–16346.

Zhang, Y.; Xiang, T.; Hospedales, T. M.; and Lu, H. 2018. Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4320–4328.

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

Zhou, Y.; Wang, J.; and Wang, Z. 2022. Bearing faulty prediction method based on federated transfer learning and knowledge distillation. *Machines*, 10(5): 376.

Zhu, H.; Xu, J.; Liu, S.; and Jin, Y. 2021. Federated learning on non-IID data: A survey. *Neurocomputing*, 465: 371–390.

Zhu, L.; Liu, Z.; and Han, S. 2019. Deep leakage from gradients. *Advances in neural information processing systems*, 32.