Rethinking Privacy Preserving Deep Learning: How to Evaluate and Thwart Privacy Attacks

Lixin Fan[*] WeBank AI Lab
Kam Woh Ng[*] WeBank AI Lab
Ce Ju[*] WeBank AI Lab
Tianyu Zhang WeBank AI Lab
Chang Liu WeBank AI Lab
Chee Seng Chan University of Malaya
Qiang Yang Hong Kong Univ. of Science and Tech.

Abstract

This paper investigates capabilities of Privacy-Preserving Deep Learning (PPDL) mechanisms against various forms of privacy attacks. First, we propose to quantitatively measure the trade-off between model accuracy and privacy losses incurred by *reconstruction, tracing and membership attacks*. Second, we formulate reconstruction attacks as solving a noisy system of linear equations, and prove that attacks are guaranteed to be defeated if condition (2) is unfulfilled. Third, based on theoretical analysis, a novel Secret Polarization Network (SPN) is proposed to thwart privacy attacks, which pose serious challenges to existing PPDL methods. Extensive experiments showed that model accuracies are improved on average by 5-20% compared with baseline mechanisms, in regimes where data privacy are satisfactorily protected.

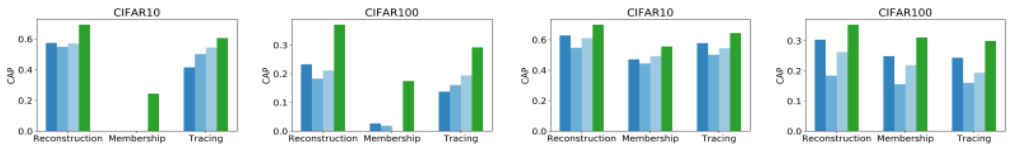
1 Introduction

Privacy-preserving deep learning (PPDL) aims to collaboratively train and share a deep neural network model among multiple participants, without exposing to each other information about their private training data. This typical *federated learning* setting is particularly attractive to business scenarios in which raw data e.g. medical records or bank transactions are too sensitive and valuable to be disclosed to other parties [11, 20]. While *differential privacy* based approaches e.g. [1, 15] attract much attentions due to its theoretical guarantee of privacy protection and low computational complexity [4, 5], there is a fundamental trade-off between *privacy guarantee* vs *utility* of learned models, i.e. overly conservative privacy protections often significantly deteriorate model utilities (*accuracies* for classification models). Existing solutions e.g. [1, 15] are unsatisfactory in our view — low ϵ privacy budget value does not necessarily lead to desired levels of privacy protection. For instance, the leakage of shared gradients may admit complete reconstruction of training data under certain circumstances [21, 18, 19, 8], even though substantial fraction of gradients elements are truncated [15] or large random noise are added [1].

In order to make critical analysis and fair evaluations of different PPDL algorithms, we argue that one must employ an *objective* evaluation protocol to quantitatively measure privacy preserving capabilities against various forms of privacy attacks. Following a privacy adversary approach [6, 12], we propose to evaluate the admitted privacy loss by three objective measures i.e. *reconstruction*, *tracing* and *membership* losses, with respect to the accuracies of protected models. To this end, Privacy-Preserving Characteristic (PPC) curves are used to delineate the trade-off, with Calibrated Averaged Performance (CAP) faithfully quantifying a given PPC curve. These empirical measures

[*] equal contribution.

Preprint. Under review.



(a) Attack Batch Size 1 **(b)** Attack Batch Size 8

Figure 1: Comparison of Calibrated Averaged Performances (CAPs) for the proposed SPN, PPDL [15] and DP [1] methods, against reconstruction, membership and tracing attacks (CAP the higher the better, see threat model and evaluation protocol in Sect. 2.1). (a): CIFAR10/100 models attacked with batch size 1; (b): CIFAR10/100 models attacked with batch size 8.

complement the theoretical bound of the privacy loss and constitute the first contribution of our work (see Figure 5 for example PPC).

As demonstrated by experimental results in Sect. 4, the leakage of shared gradients poses serious challenges to existing PPDL methods[1, 15, 21]. Our second contribution, therefore, is a novel *secret polarization network* (SPN) and a polarization loss term, which bring about two advantages in tandem with public backbone networks — first, SPN helps to defeat privacy attacks by adding *secrete* and *element-wise adaptive* gradients to shared gradients; second, the added polarization loss acts as a regularization term to consistently improve the classification accuracies of baseline networks in federated learning settings. This SPN based mechanism has demonstrated strong capability to thwart three types of privacy attacks without significant deterioration of model accuracies. As summarized by CAP values in Fig. 1, SPN compares favorably with existing solutions [15] and [1] with pronounced improvements of performances against reconstruction, membership and tracing attacks.

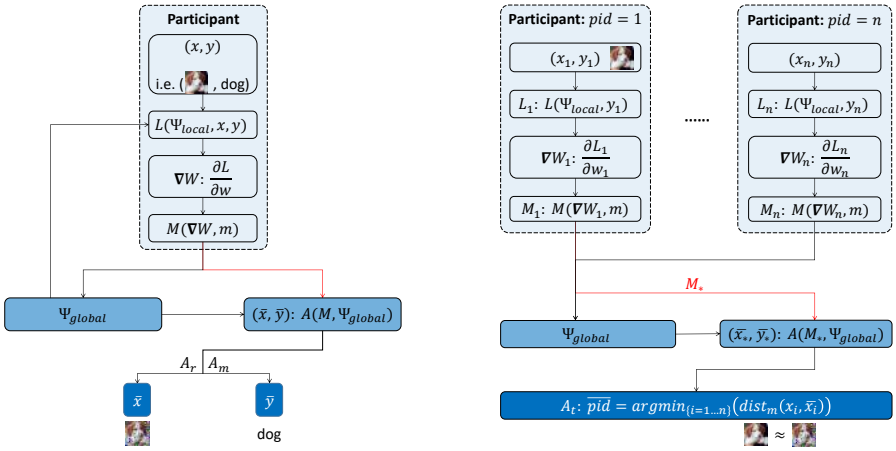
Our third contribution is the formulation of reconstruction attacks as solving a *noisy system of linear equations*, and it is proved that reconstructions are guaranteed to fail if the necessary condition (2) in Theorem 2.2 is purposely invalidated. This theoretical analysis sheds new light on the effectiveness of DP based privacy preserving mechanisms.

1.1 Related Work

[1] demonstrated how to maintain data privacy by adding Gaussian noise to shared gradients during the training of deep neural networks. [15] proposed to randomly select and share a small fraction of gradient elements (those with large magnitudes) to reduce privacy loss. Although both methods [1, 15] offered strong *differential privacy* (DP) guarantees [4, 5], as shown by [12, 21] and our empirical studies, pixel-level reconstructions of training data and disclosing of membership information raise serious concerns about potential privacy loss.

Dwork et.al. [6] have formulated privacy attacks towards a database, as a series of queries maliciously chosen according to an attack strategy designed to compromise privacy. Among three privacy attacks i.e. *reconstruction*,*tracing* and *re-identification* discussed in [6], the detrimental reconstruction attack is formulated as solving a noisy system of linear equations, and reconstruction errors are essentially bounded by the worst-case accuracies of query answers (Theorem 1 in [6]). However, this formulation is not directly applicable to deep learning, since queries about private training data are not explicitly answered during the training or inferencing of DNNs.

In the context of deep learning, membership attacks was investigated in [16] while [7] demonstrated that recognizable face images can be recovered from confidence values revealed along with predictions. [12] demonstrated with both CNNs and RNNs that periodical gradient updates during training leaked information about *training data, features* as well as class *memberships*. Possible defences such as selective gradient sharing, reducing dimensionality, and dropout were proved to be ineffective or had a negative impact on the quality of the collaboratively trained model. Based on the assumption that activation functions are twice-differentiable, recent attacks were proposed to reconstruct training data with pixel-level accuracies [21, 18, 19, 8]. These recent reconstruction attacks were adopted in the present work to evaluate capabilities of privacy-preserving strategies proposed in [1, 15, 12, 21], with extensive experiments conducted over different networks and datasets (see Sect. 4 and supplementary material).



(a) Reconstruction attacks $\mathcal{A}_r$, with relative MSE between reconstructed and original data $\frac{\|\bar{x}-x\|}{\|x\|}$. **Membership attacks** $\mathcal{A}_m$, with categorical distance between reconstructed and original labels $dist_m(\bar{y}, y)$.

(b) Tracing attacks $\mathcal{A}_t$, with categorical distance between recovered and actual participant IDs $dist_m(\overline{pid}, p)$.

Figure 2: Three privacy attacks considered in this work (see text in Sect. 2).

Homomorphic-Encryption (HE) based [9, 10, 2] and Secure Multi-Party Computation (MPC) based privacy-preserving approaches [14, 13] demonstrated strong privacy protection via encryption, but often incur significantly more demanding computational and communication costs. For instance, [2] reported 2-3 times communication overheads and [13, 3] had to speed up highly-intensive computation with efficient implementations. In this paper our work is only compared with Differential Privacy based mechanisms [15, 1], and we refer readers to [20, 17] for thorough reviews of HE and MPC based privacy-preserving methods therein.

2 Privacy Attacks on Training Data

In this work we consider a distributed learning scenario, in which $K (K \geq 2)$ participants collaboratively learn a multi-layered deep learning model without exposing their private training data (this setting is also known as *federated learning* [11, 20]). We assume one participant is the *honest-but-curious adversary*. The adversary is honest in the sense that he/she faithfully follows the collaborative learning protocol and does not submit any malformed messages, but he/she may launch privacy attacks on the *training data* of other participants, by analyzing periodic updates to the joint model (e.g. *gradients*) during training.

Fig. 2 illustrates three privacy attacks considered in this work. The goal of *reconstruction attack* is to recover original training data x as accurate as possible by analyzing the publicly shared gradients, which might be perturbed by privacy-preserving mechanisms. Subsequent *membership attack* and *tracing attack* are based on reconstruction attacks — for the former, membership labels are derived either directly during the reconstruction stage or by classifying reconstructed data; for the latter, the goal is to determine whether a given training data item belongs to certain participant, by comparing it against reconstructed data[2].

2.1 Evaluation of Trade-off by Privacy Preserving Mechanism

We assume there is a Privacy-Preserving Mechanism (PPM)[3] $\mathcal{M}$ that aims to defeat the privacy attacks $\mathcal{A}$ by modifying the public information from $\mathcal{G}$ to $\bar{\mathcal{G}}_m = \mathcal{M}(\mathcal{G}, m)$, that is exchanged during the learning stage and m is the controlling parameter of the amount of changes exerted on $\mathcal{G}$. This modification protects the private information x from being disclosed to the adversary, who can only make an estimation based on public information i.e. $\bar{x}_m = \mathcal{A}(\bar{\mathcal{G}}_m)$. Needless to

[2]Note that membership inference in [12] is the tracing attack considered in our work.

[3]We do not restrict ourselves to privacy mechanisms considered by differential privacy[4, 5, 15, 1].

say, a PPM can defeat any adversaries by introducing exorbitant modification so that $dist(\bar{x}, x)$ *is as large as possible*, where $dist()$ is a properly defined distance measure such as MSE. The modification of public information, however, inevitably deteriorates the performances of global models i.e. $Acc(\bar{\mathcal{G}}_m) \leq Acc(\mathcal{G}_m)$, where $Acc()$ denotes model performances such as accuracies or any other metrics that is relevant to the model task in question. A well-designed PPM is expected to have $Acc(\bar{\mathcal{G}}_m)$ *as high as possible*.

We propose to plot Privacy Preserving Characteristic (PPC) to illustrate the trade-off between two opposing goals i.e. to maintain high model accuracies and low privacy losses as follows,

Definition 1 (Privacy Preserving Characteristic)**.** *For a given Privacy-Preserving Mechanism $\mathcal{M}$, its privacy loss and performance trade-off is delineated by a set of calibrated performances i.e. $\{Acc(\bar{\mathcal{G}}_m) \cdot dist(\bar{x}_m, x) | m \in \{m_1, \cdots, m_n\}\}$, where $Acc()$ is the model performance, $dist()$ a distance measure, $\bar{\mathcal{G}}_m = \mathcal{M}(\mathcal{G}, m)$ is the modified public information, x is the private data, $\bar{x}_m = \mathcal{A}(\bar{\mathcal{G}}_m)$ is the estimation of private data by the attack and m the controlling parameter of the mechanism.*

Moreover, Calibrated Averaged Performance (CAP) for a given PPC is defined as follows,

$$CAP(\mathcal{M}, \mathcal{A}) = \frac{1}{n} \sum_{m=m_1}^{m_n} Acc(\bar{\mathcal{G}}_m) \cdot dist(\bar{x}_m, x). \tag{1}$$

Fig. 5 illustrates example PPCs of different mechanisms against privacy attacks. One may also quantitatively summarize PPCs with CAP — *the higher the CAP value is, the better the mechanism is at preserving privacy without compromising the model performances* (see Table 1).

2.2 Formulation of Reconstruction Attack

Consider a neural network $\Psi(x; w, b) : \mathcal{X} \to \mathbb{R}^C$, where $x \in \mathcal{X}$, w and b are the weights and biases of neural networks, and C is the output dimension. In a machine learning task, we optimize the parameters w and b of neural network Ψ with a loss function $\mathcal{L}\big(\Psi(x; w, b), y\big)$, where x is the input data and y is the ground truth labels. We denote the superscript $w^{[i]}$ and $b^{[i]}$ as the i-th layer weights and biases. The following theorem proves that the reconstruction of input x exists under certain conditions (proofs are given in Appendix A, in supplementary material due to the limited space).

Theorem 2.1. *Suppose a multilayer neural network $\Psi := \Psi^{[L-1]} \circ \Psi^{[L-2]} \circ \cdots \circ \Psi^{[0]}(\,\cdot\,; w, b)$ is $\mathcal{C}^1$, where the i-th layer $\Psi^{[i]}$ is a fully-connected layer[4] Then,* ***initial input*** *x^** ***of*** *Ψ* ***exists****, provided that: if there is an i $(1 \leq i \leq L)$ such that*

1. *Jacobian matrix $D_x\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)$ around x is full-rank;*
2. *Partial derivative $\nabla_{b^{[i]}} \mathcal{L}\big(\Psi(x; w, b), y\big)$[5] is nonsingular.*

If assumptions in Theorem 2.1 are met, we can pick an index set I from row index set of $\nabla_{w^{[i]}, b^{[i]}} \mathcal{L}\big(\Psi(x; w, b), y\big)$ such that the following linear equation is well-posed,

$$B_I \cdot x = W_I,$$

where $B_I := \nabla^I_{b^{[i]}} \mathcal{L}\big(\Psi(x; w, b), y\big)$ and $W_I := \nabla^I_{w^{[i]}} \mathcal{L}\big(\Psi(x; w, b), y\big)$. According to Theorem 2.1, the initial input x^* is $\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)^{-1}(x)$.

The linear system can be composed from any subsets of observed gradients elements, and the reconstruction solution exists as long as the condition of full rank matrix is fulfilled. For common privacy-preserving strategies adopted in a distributed learning scenario such as *sharing fewer gradients* or *adding noisy to shared gradients* [15, 1, 12], the following theorem proves that input x can be reconstructed from such a noisy linear system, if condition (2) is fulfilled.

[4] Any convolution layers can be converted into a fully-connected layer by simply stacking together spatially shifted convolution kernels (see proofs in supplementary material).

[5] We write the partial derivative as a diagonal matrix that each adjacent diagonal entries in an order are copies of each entry in $\nabla_{b^{[i]}} \mathcal{L}\big(\Psi(x; w, b), y\big)$, see proofs in Appendix for details.

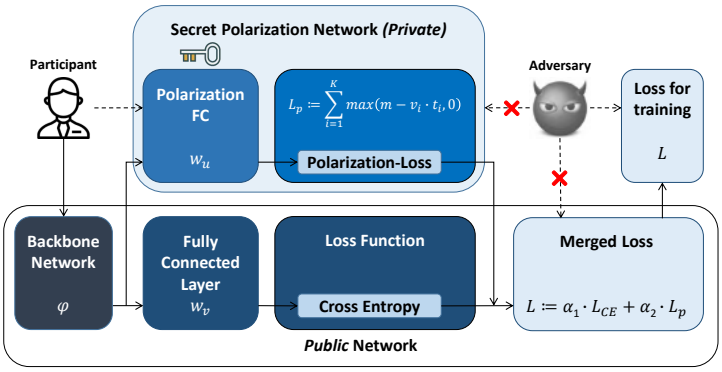


Figure 3: Our proposed SPN architecture that consists of a public and a private network (see text in Sect. 3).

Theorem 2.2. *Suppose there are perturbations E_B, E_W added on B_I, W_I, respectively, such that observed measurements $\bar{B}_I = B_I + E_B, \bar{W}_I = W_I + E_W$. Then, the* ***reconstruction*** *x^** ***of the initial input*** *x* ***can be determined*** *by solving a noisy linear system $\bar{B}_I \cdot x^* = \bar{W}_I$, provided that*

$$\|B_I^{-1} \cdot E_B\| < 1; \tag{2}$$

Moreover, the relative error is bounded,

$$\frac{\|x^* - x\|}{\|x\|} \leq \frac{\kappa(B_I)}{1 - \|B_I^{-1} \cdot E_B\|} \Big(\frac{\|E_B\|}{\|B_I\|} + \frac{\|E_W\|}{\|W_I\|} \Big), \tag{3}$$

in which B_I^{-1} is the inverse of B_I, where $\kappa(B_I)$ is the conditional number of B_I.

In the deep leakage approach [21], the recovery of initial image requires model parameters $\mathcal{W}$ and the corresponding gradients $\nabla\mathcal{W}$ such that a minimization of gradient differences $E_p := ||\nabla\mathcal{W}' - \nabla\mathcal{W}||$ yields a recovery $\bar{x}$ of initial image. The minimizing error E_p introduces more errors to the noisy linear system. Therefore, for any iterative reconstruction algorithms like [21] to be successful, condition $\|B_I^{-1} \cdot E_B\| < 1$ is *necessary*. In other words, a sufficiently large perturbation $\|E_B\| > \|B_I\|$ such as Gaussian noise is *guaranteed* to defeat reconstruction attacks. To our best knowledge, (2) is the first analysis that elucidates a theoretical guarantee for thwarting reconstruction attacks like [21]. Nevertheless, existing mechanisms [15, 1] have to put up with significant drops in model accuracy incurred by high levels of added noise (see Sect. 4.2).

3 Privacy Preserving with Secret Polarization Network

In Sect. 2 we have proved that the necessary condition of successful reconstruction attack is unfulfilled if sufficiently large perturbations are added. We illustrate in this section a novel multi-task dual-headed networks, which leverages private network parameters and element-wise adaptive gradient perturbations to defeat reconstruction attacks and, simultaneously, maintain high model accuracies.

3.1 Secret Perturbation of Gradients via Polarization Loss

Fig. 3 illustrates a Secret Polarization Network (SPN), in which fully connected polarization layers are kept private with its *parameters not shared* during the distributed learning process. Appendix shows the pseudo codes of the proposed method.

Formally, the proposed dual-headed network consists of a public and a private SPN network based on a backbone network: $\Psi\big(\varphi(\,\cdot\,; w, b); w_u, b_u\big) \oplus \Phi\big(\varphi(\,\cdot\,; w, b); w_v, b_v\big) : \mathcal{X} \to [0,1]^C \oplus \mathbb{R}^K$, i.e. $u \oplus v = \Psi\big(\varphi(x; w, b); w_u, b_u\big) \oplus \Phi\big(\varphi(x; w, b); w_v, b_v\big) \in [0,1]^C \oplus \mathbb{R}^K$, where $\varphi(\,\cdot\,; w, b)$ is the backbone network. The multi-task composite loss is as follows,

$$\mathcal{L}\big(\Psi \oplus \Phi, y \oplus t\big) := \alpha_1 \cdot \mathcal{L}_{CE}(u, y) + \alpha_2 \cdot \mathcal{L}_P(v, t) \tag{4}$$

$$= \underbrace{\alpha_1 \cdot \sum_{c=1}^{C} -y_c \cdot \log(u_c)}_{\text{CE loss}} + \underbrace{\alpha_2 \cdot \sum_{c=1}^{C} \sum_{k=1}^{K} \max(m - v_k \cdot t_c^k, 0)}_{\text{polarization loss}}, \tag{5}$$

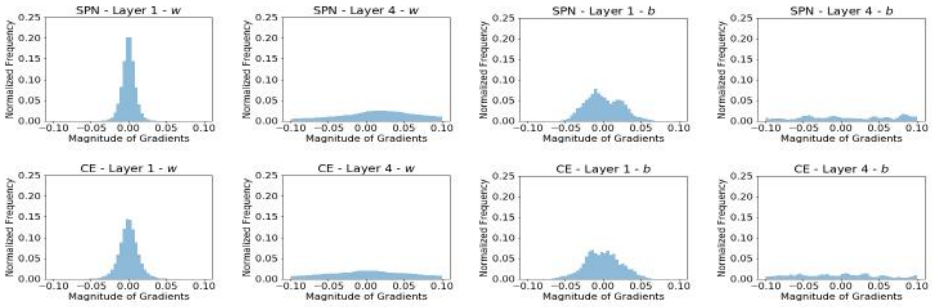


Figure 4: Distributions of gradients at each layer. **Left**: distributions of gradients w.r.t. *weights, w*; **Right**: distributions of gradients w.r.t. *biases, b*; **Top**: gradients by polarization loss; **Bottom**: gradients by CE loss. Cosine similarities between gradients by polarization and CE losses are (from left to right): -0.0033, 0.1760, 0.0243, and 0.1861 respectively.

where α_1 and α_2 are hyper-parameters with $\alpha_1 + \alpha_2 = 1$. y_c is an one-hot representation of labels for class c, and $t_c \in \{-1, +1\}^K$ is the target K-bits binary codes randomly assigned to each class c for $c = 1, \cdots, C$. Note that by minimizing the polarization loss, Hamming distances between thresholded outputs $Bin(v_k)$ of *intra-class* data items are minimized and, at the same time, Hamming distances are maximized for *inter-class* data items (where $Bin(v_k) \in \{-1, +1\}$, see proofs in Appendix). The polarization loss therefore joints forces with the CE loss to improve the model accuracies.

At each step of the optimization, the gradient of the loss $\bigtriangledown_{w,b}\mathcal{L}\big(\Psi \oplus \Phi, y \oplus t\big)$ is a linear combination of gradient of CE loss and polarization loss as follows,

$$\bigtriangledown_{w,b}\mathcal{L} = \alpha_1 \cdot \sum_{c=1}^{C}(y_c - u_c) \cdot \frac{\partial u_c}{\partial w, b} + \underbrace{\alpha_2 \cdot \sum_{c=1}^{C}\sum_{k \in \mathcal{I}^c}(-t_c^k) \cdot \frac{\partial v_k}{\partial w, b}}_{\text{secret perturbation}}, \tag{6}$$

where $\mathcal{I}^c := \Big\{k \in \{1, \cdots, K\} \Big| m - v_k \cdot t_c^k > 0\Big\}$.

Note that w_v is kept secret from other participants including the adversary. The summand due to the polarization loss in (6) is therefore unknown to the adversaries, and acts as perturbations to gradients ascribed to the CE loss. Perturbations introduced by polarization loss, on the one hand, protect training data with α_2 controlling the protection levels. On the other hand, SPN gradients back-propagated to the backbone network layers exhibit strong correlations with CE gradients (see distributions and cosine similarities between gradients by polarization and CE losses in Fig. 4). We ascribe improvements of the model accuracies brought by SPN to element-wise adaptive perturbations introduced by polarization loss.

4 Experimental Results

4.1 Experiment Setup and Evaluation Metrics

Dataset. Popular image datasets MNIST and CIFAR10/100 are used in our experiments. Implementation of **DP** [1] method from Facebook Research Team [6] is used. Implementation[7] of **PPDL** [15] method from Torch/Lua are re-implemented in PyTorch/Python. PPDL is similar to gradient pruning which is one of the suggested protections in [21]. We only show in this paper results with 5% and 30% of selected gradients, named respectively, as **PPDL-0.05** and **PPDL-0.3**. We refer reviewers to more results in the supplementary material. Implementation of **Deep Leakage** attack [21], network architecture and default setting from the official released source code[8] are used in all experiments with training batch size set as $\{1, 4, 8\}$ respectively. Following analysis in [19], we adopt *pattern-initialization* for higher reconstruction successful rates.

[6] `https://github.com/facebookresearch/pytorch-dp`

[7] `https://www.comp.nus.edu.sg/~reza/files/PPDL.zip`

[8] `https://github.com/mit-han-lab/dlg`

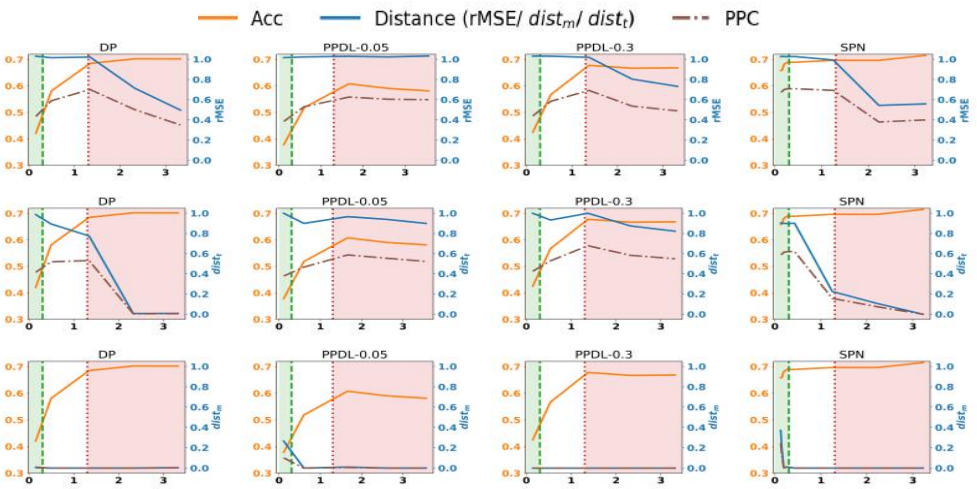
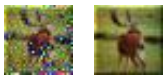


Figure 5: Privacy-Preserving Characteristics (PPC) of different mechanisms (dash-dotted PPC curves); orange curves and y-axis (left): Acc of models; blue curves and y-axis (right): distances for attacks; x-axis: controlling param. $log_{10}(\frac{||B_I||}{||E_B||} + 1)$. **Left to Right**: DP, PPDL-0.05, PPDL-0.3 and SPN (Ours). **Top**: Reconstruction Attack; **Middle**: Tracing Attack; **Bottom**: Membership Attack. See Fig. 6 for example reconstruction images.

(a) Left:1.04 (0.43); **Right**:1.13 (0.54)

(b) Left:1.01 (18.05); **Right**:1.00 (21.91)



(c) Left:0.86 (43.67); **Right**:0.02 (2186.8)

Figure 6: Reconstructed images from different region in Fig. 5. **(a)** Green region **(b)** White region **(c)** Red region. Values inside bracket are $\frac{||B_I||}{||E_B||}$ and values outside are rMSE of reconstructed w.r.t. original images.

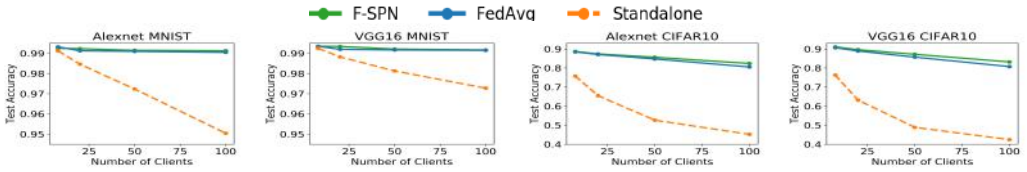
Relative Mean Square Error (rMSE) (= $\frac{||x^* - x||}{||x||}$) is used to measure the distances between reconstructed and original data. **Membership Distance** ($dist_m(y^*, y)$) is the *averaged categorical distances* between recovered data labels and original labels. **Tracing Distance** ($dist_t(x)$) is the *averaged categorical distances* between recovered participant IDs and original IDs, to which the given data x belongs.

4.2 Comparison of Privacy Preserving Mechanisms

Fig. 5 illustrates example Privacy-Preserving Characteristic (PPC) of different mechanisms against *reconstruction, membership* and *tracing* attacks, in which the controlling parameter along x-axis is the ratio m of gradient magnitudes $\|B_I\|$ with respect to magnitudes of added perturbations$\|E_B\|$. It is shown that privacy attacks pose serious challenges to differential privacy based methods **DP** and **PPDL**.

Reconstruction attacks (top row): when the ratio ranges between tens to thousands in red regions, errors decrease rapidly and *pixel-level information* about original training data are almost completely disclosed (see Fig. 6c). In the white regions, increased magnitudes of perturbations lead to large reconstruction errors (rMSE $\approx$ 1.0) with noticeable artifacts and random noisy dots in Fig. 6b. However, model accuracies for DP and PPDL methods also decrease dramatically. Pronounced drops in accuracies (with more than 20% for CIFAR10 and 5% for MNIST) are observed when added perturbations exceed magnitudes of original gradients (in green regions), beyond which condition (2) of reconstruction attacks is no longer fulfilled and attacks are guaranteed to be defeated (see Theorem 2.2 and Fig. 6a).

Tracing attacks (middle row): similar trends were observed for distances of tracing attacks. In addition, the distance increases as the number of participants increases. We refer reviewers to ablation studies in supplementary material due to the limited space of this submission.



(a) MNIST **(b)** CIFAR10

Figure 7: Comparison of accuracies for *standalone* local models, *FedAvg* global model and *Federated SPN* model. Improvements over standalone models increase with the number of clients.

	CIFAR10									CIFAR100								
	Reconstruction			Membership			Tracing			Reconstruction			Membership			Tracing		
BS	1	4	8	1	4	8	1	4	8	1	4	8	1	4	8	1	4	8
[1]	0.57	0.63	0.63	0.00	0.45	0.47	0.42	0.57	0.58	0.23	0.31	0.30	0.01	0.22	0.25	0.14	0.24	0.24
[15]∗	0.55	0.55	0.55	0.00	0.37	0.44	0.50	0.50	0.50	0.18	0.18	0.18	0.02	0.13	0.16	0.16	0.16	0.16
[15]⋆	0.57	0.61	0.61	0.00	0.43	0.49	0.54	0.54	0.54	0.21	0.26	0.26	0.00	0.19	0.22	0.19	0.19	0.19
SPN	**0.69**	**0.70**	**0.70**	**0.24**	**0.50**	**0.55**	**0.60**	**0.62**	**0.64**	**0.35**	**0.35**	**0.36**	**0.17**	**0.28**	**0.31**	**0.29**	**0.30**	**0.30**

Table 1: CAP performance with different batch size and dataset for reconstruction, membership and tracing attack. Higher better. BS = Attack Batch Size, [1] = DP, [15]∗ = PPDL-0.05, [15]⋆ = PPDL-0.3

Membership attacks (bottom row): the disclosing of memberships is more detrimental, with distances between reconstructed memberships and ground truth labels almost being zero, except for PPDL-0.05 in the green region. With the increase of the number of classes (for CIFAR100) and the training batch size (8), success rates of membership attacks dropped and the distances increased. One may mitigate membership attacks by using even larger batch sizes, as suggested in [21, 19].

In a sharp contrast, Secret Polarization Network (SPN) based mechanism maintains consistent model accuracies, even though gradient magnitudes due to polarization loss exceed gradient magnitudes of original CE loss. Superior performances of SPN mechanism in this green region provide *theoretically guaranteed privacy-preserving capabilities*, and at the same time, maintain decent model accuracies to be useful in practice. This superiority is ascribed to the adaptive element-wise gradient perturbations introduced by polarization loss (see discussions near Eq. (6)).

4.3 SPN Polarization Network for Federated Learning

The dual-headed Secret Polarization Network (SPN) brought improvements in model accuracies in a federated learning setting, in which MNIST and CIFAR10 datasets are evenly distributed among all clients, resulting in small local training datasets on each client (for instance, there are only 500 CIFAR10 training data when the number of clients is 100). Substantial performances deterioration were observed for local standalone models with large numbers of e.g. 100 clients (see Fig. 7). Since local training data are *i.i.d.*, the FedAvg algorithm [11] effectively improved the global model accuracies about 2-4% for MNIST and 10-40% for CIFAR10. The proposed SPN, once integrated with the FedAvg algorithm, consistently improved further model accuracies ranging between 2-3% for CIFAR10 dataset and about 0.2% for MNIST (see more results in supplementary material). The improvements are ascribed to element-wise gradients introduced by polarization losses (see discussion in Sect. 3), which in our view advocate the adoption of SPN in practical applications.

5 Discussion and Conclusion

The crux of differential-privacy based approaches is a trade-off between privacy vs accuracy [15, 1]. As shown in [12] and our experiments, existing defenses such as *sharing fewer gradients* and *adding Gaussian or Laplacian noise* are vulnerable to aggressive reconstruction attacks, despite the theoretical privacy guarantee. We extricated from the dilemma by hiding a fraction of network parameters and gradients from the adversary. To this end, we proposed to employ a dual-headed network architecture i.e. Secret Polarization Network (SPN), which on the one hand exerts secret gradient perturbations to original gradients under attack, and on the other hand, maintains performances of the global shared model by jointing forces with the backbone network. This secret-public network configuration provides a theoretically guaranteed privacy protection mechanism without compromising model accuracies, and does not incur significant computational and communication overheads which HE/SMPC based approaches have to put up with. We find that the combination of secret-public networks

provides a preferable alternative to DP-based mechanisms in application scenarios, whereas large computational and communication overheads are unaffordable e.g. with mobile or IOT devices. As for future work, the adversarial learning nature of SPN also makes it an effective defense mechanism against adversarial example attacks. To formulate both privacy and adversarial attacks in a unified framework is one of our future directions.

Broader Impact

Our benchmark is likely to increase progress of federated learning and encourage more companies and people to share their data. While there will be immediate benefits resulted from the use of SPN in general, here we also advocate the impact of using our measurement tool to evaluate and thwart privacy attacks. Benefits of using such a tool include increasing transparency in federated learning applications, and mitigating data safety risks in distributed machine learning - see introduction of the paper for more details.

The *sharing of local model updates* in distributed learning scenarios, concomitantly disclose *privacy of local data* if no protection measures are taken. Our investigations about the trade-off between *data privacy* protection and *model utilities* for differential-privacy (DP) based approaches, therefore, is of interest to people who concern about the risks of reverse engineering and/or stealing of valuable private data. Moreover, the theoretical guarantee (2) for the first time lays the foundation for a series of protection mechanisms, one of which is instantiated by a secret polarization network (SPN) that thwarts privacy attacks and maintains high model utilities at the same time. The proposed secret-public network configuration, on its own, also paves the way for a novel research direction in our view. Finally, source codes of this work will be made publicly available for people to reproduce and follow up.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2017.

[3] Ahmad Al Badawi, Jin Chao, Jie Lin, Chan Fook Mun, Sim Jun Jie, Benjamin Hong Meng Tan, Xiao Nan, Khin Mi Mi Aung, and Vijay Ramaseshan Chandrasekhar. The alexnet moment for homomorphic encryption: Hcnn, the first homomorphic CNN on encrypted data with gpus. *CoRR*, abs/1811.00778, 2018.

[4] Cynthia Dwork. Differential privacy. *Automata, languages and programming*, pages 1–12, 2006.

[5] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284, 2006.

[6] Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. Exposed! a survey of attacks on private data. *Annual Review of Statistics and Its Application*, 4:61–84, 2017.

[7] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[8] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients–how easy is it to break privacy in federated learning? *arXiv preprint arXiv:2003.14053*, 2020.

[9] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*, pages 201–210, 2016.

[10] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*, 2017.

[11] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[12] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE, 2019.

[13] Payman Mohassel and Peter Rindal. Aby3: A mixed protocol framework for machine learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 35–52, 2018.

[14] Bita Darvish Rouhani, M Sadegh Riazi, and Farinaz Koushanfar. Deepsecure: Scalable provably-secure deep learning. In *Proceedings of the 55th Annual Design Automation Conference*, pages 1–6, 2018.

[15] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

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

[17] Harry Chandra Tanuwidjaja, Rakyong Choi, and Kwangjo Kim. A survey on deep learning techniques for privacy-preserving. In *International Conference on Machine Learning for Cyber Security*, pages 29–46, 2019.

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

[19] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating gradient leakage attacks in federated learning. *arXiv preprint arXiv:2004.10397*, 2020.

[20] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):12, 2019.

[21] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *NeurIPS*, pages 14747–14756, 2019.

Appendix A: Proofs of Reconstruction Attacks

Consider a neural network $\Psi(x; w, b) : \mathcal{X} \to \mathbb{R}^C$, where $x \in \mathcal{X}$, w and b are the weights and biases of neural networks, and C is the output dimension. In a machine learning task, we optimize the parameters w and b of neural network Ψ with a loss function $\mathcal{L}\big(\Psi(x; w, b), y\big)$, where x is the input data and y is the ground truth labels. We abbreviate loss function as $\mathcal{L}$ and denote the superscript $w^{[i]}$ and $b^{[i]}$ as the i-th layer weights and biases.

Suppose a multilayer neural network $\Psi := \Psi^{[L-1]} \circ \Psi^{[L-2]} \circ \cdots \circ \Psi^{[0]}(\,\cdot\,; w, b)$ is $\mathcal{C}^1$, where the i-th layer $\Psi^{[i]}$ is a fully-connected layer with the step forward propagation as follows,

$$o^{[i+1]} = a\big(w^{[i]} \cdot o^{[i]} + b^{[i]}\big),$$

where $o^{[i]}$, $o^{[i+1]}$, $w^{[i]}$ and $b^{[i]}$ are an input vector, an output vector, a weight matrix and a bias vector respectively, and a is the activation function in the i-th layer.

By the backpropagation, we have the matrix derivatives on $\Psi^{[i]}$ as follows,

$$\nabla_{w^{[i]}}\mathcal{L} = \nabla_{o^{[i+1]}}\mathcal{L} \cdot a'\big(w^{[i]} \cdot o^{[i]} + b^{[i]}\big) \cdot {o^{[i]}}^T$$
$$\nabla_{b^{[i]}}\mathcal{L} = \nabla_{o^{[i+1]}}\mathcal{L} \cdot a'\big(w^{[i]} \cdot o^{[i]} + b^{[i]}\big) \cdot I,$$

which yield the following output equations:

$$\nabla_{w^{[i]}}\mathcal{L} = \nabla_{b^{[i]}}\mathcal{L} \cdot {o^{[i]}}^T, \tag{7}$$

where gradients $\nabla_{w^{[i]}}\mathcal{L}$ and $\nabla_{b^{[i]}}\mathcal{L}$ are supposed to be shared in a distributed learning setting, and known to honest-and-curious adversaries who may launch reconstruction attacks on observed gradients.

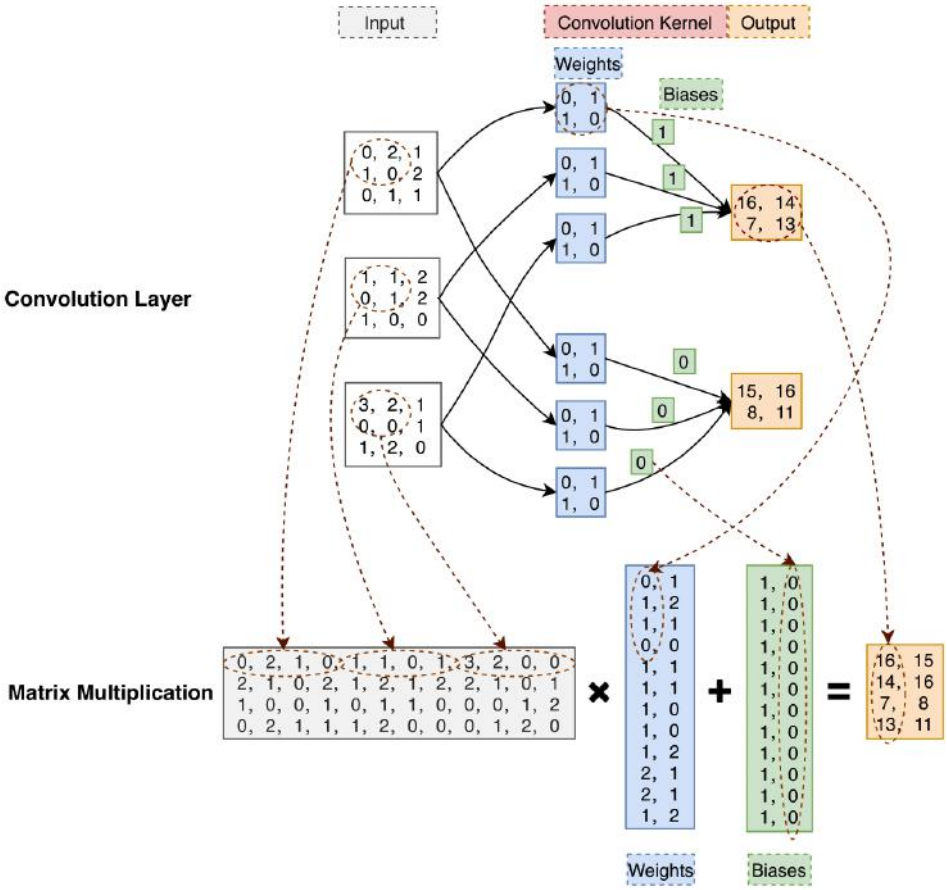


Figure 8: A pictorial example illustrating how to switch a convolution operator to a matrix multiplication.

Remark. *Any convolution layers can be converted into a fully-connected layer by simply stacking together spatially shifted convolution kernels, as noted in Footnote 3. A simple illustration refers to Figure 8 and detailed algorithm refers to a technical report* [9].

Remark. *Suppose* $\nabla_{w^{[i]}}\mathcal{L} \in \mathbb{R}^{M\cdot N}$, $\nabla_{b^{[i]}}\mathcal{L} \in \mathbb{R}^{M}$ *and* $o^{[i]} \in \mathbb{R}^{N}$, *we write*

$$\nabla_{w^{[i]}}\mathcal{L} := \Big(\frac{\partial \mathcal{L}}{\partial w_{mn}^{[i]}}\Big)_{\substack{1\le m\le M;\\ 1\le n\le N.}}, \quad \nabla_{b^{[i]}}\mathcal{L} := \Big(\frac{\partial \mathcal{L}}{\partial b_{1}^{[i]}}, \cdots, \frac{\partial \mathcal{L}}{\partial b_{M}^{[i]}}\Big)^{T}, \text{ and } o^{[i]} := \Big(o_1^{[i]}, \ldots, o_N^{[i]}\Big)^{T}.$$

By the piecewise matrix multiplication, Equation 7 becomes as a linear system in a formal convention as follows,

$$\frac{\partial \mathcal{L}}{\partial w_{mn}^{[i]}} = \frac{\partial \mathcal{L}}{\partial b_{m}^{[i]}} \cdot o_n^{[i]}, \quad \textit{for } 1 \le m \le M \textit{ and } 1 \le n \le N.$$

Hence, we can write the partial derivative $\nabla_{b^{[i]}}\mathcal{L}$ *as an* $mn \times mn$ *diagonal matrix that each n adjacent diagonal entries in an order are copies of each entry, and partial derivative* $\nabla_{w^{[i]}}\mathcal{L}$ *as an* mn*-dimensional vector.*

In the following paragraph, we always abbreviate equation coefficients $\nabla_{w^{[i]}}\mathcal{L}$ and $\nabla_{b^{[i]}}\mathcal{L}$ to $W^{[i]}$ and $B^{[i]}$ respectively.

Lemma 5.1. *Suppose* $d^{[0]}$ *and* $d^{[1]}, \cdots, d^{[L]}$ *are dimensions of input image* x *and output vectors* $o^{[1]}, \cdots, o^{[L]}$ *respectively.* x *and* $o^{[i]}$ *can be estimated by solving the following* $d^{[i]} \cdot d^{[i+1]}$*-dimensional linear system if it is well-posed,*

$$W^{[0]} = B^{[i]} \cdot x \tag{8}$$

$$or \;\; W^{[i]} = B^{[i]} \cdot {o^{[i]}}^{T}, \;\; \textit{for } i = 1, \cdots, L-1. \tag{9}$$

Remark. *Output vectors* $o^{[1]}, \cdots, o^{[L]}$ *are outputs of neural networks* $\Psi(\,\cdot\,; w, b)$ *on input image* x*. However, solving Linear System (9) are always numerically unstable in that minor numerical perturbation of* $B^{[i]}$ *around 0 would yield the infinity solution even if it is a well-posed problem. Hence, it is not typically to directly recover input image* x *and output vectors* $o^{[1]}, \cdots, o^{[L]}$ *by simple matrix computations in practice.*

Lemma 5.2. *Assume the linear system* $B \cdot x = W$ *is corrupted in coefficients written as* $\bar{B} \cdot \bar{x} = \bar{W}$*. If* B *is nonsingular, we have the following inequality,*

$$||x - \bar{x}|| \le ||B^{-1}|| \cdot \big(||W - \bar{W}|| + ||B - \bar{B}|| \cdot ||\bar{x}||\big).$$

Proof. Obviously, we have

$$B \cdot (x - \bar{x}) = (W - \bar{W}) + (B - \bar{B}) \cdot \bar{x}, \tag{10}$$

which yields this lemma if B is nonsingular. □

According to Lemma 5.1 and Lemma 5.2, we have the following existing theorem.

Theorem 5.3. *Suppose a multilayer neural network* $\Psi := \Psi^{[L-1]} \circ \Psi^{[L-2]} \circ \cdots \circ \Psi^{[0]}(\,\cdot\,; w, b)$ *is* $\mathcal{C}^1$*, where the* i*-th layer* $\Psi^{[i]}$ *is a fully-connected layer. Then,* ***initial input*** x^* ***of*** Ψ ***exists****, provided that: if there is an* i $(1 \le i \le L)$ *such that*

1. *Jacobian matrix* $D_x\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)$ *around* x *is full-rank;*
2. *Partial derivative* $\nabla_{b^{[i]}}\mathcal{L}\big(\Psi(x; w, b), y\big)$ *is nonsingular.*

Moreover, we have the following inequality around x^**,*

$$||x - x^*|| \le M \cdot ||\nabla_{w^{[i]}, b^{[i]}}\mathcal{L}\big(\Psi(x; w, b), y\big) - \nabla_{w^{[i]}, b^{[i]}}\mathcal{L}\big(\Psi(x^*; w, b), y\big)||. \tag{11}$$

Proof. WLOG, we suppose i yields that Jacobian matrix $D_x\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)$ around x is full-rank. By the implicit function theorem, there exists a bounded inverse function $\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)^{-1}(\,\cdot\,; w, b)$ around x, s.t.

$$\big|\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)^{-1}(\,\cdot\,; w, b)\big| \le M^{[i]}. \tag{12}$$

Since partial derivative $\nabla_{b^{[i]}}\mathcal{L}$ is nonsingular, vector $o^{[i]}$ is solved by matrix computations in Lemma 5.1, and thus the initial image $x^* := \big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)^{-1}(o^{[i]})$.

[9] Wei Ma, Jun Lu: An Equivalence of Fully Connected Layer and Convolutional Layer. `https://arxiv.org/pdf/1712.01252.pdf`

By Lemma 5.2 and Inequality (12), in an open neighborhood of x^*, we have

$$\begin{aligned}||x - x^*|| =& ||\big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)^{-1}(o^{[i]}; w, b) - \big(\Psi^{[i-1]} \circ \Psi^{[i-1]} \circ \cdots \circ \Psi^{[0]}\big)^{-1}(o^{[i]^*}; w, b)|| \\ \leq& M^{[i]} \cdot ||o^{[i]} - o^{[i]*}|| \\ \leq& M^{[i]} \cdot ||\nabla_{b^{[i]}}\mathcal{L}^{-1}|| \cdot \big(||\nabla_{w^{[i]}}\mathcal{L}(\Psi(x; w, b), y) - \nabla_{w^{[i]}}\mathcal{L}(\Psi(x^*; w, b), y)|| \\ &+ ||x^*|| \cdot ||\nabla_{b^{[i]}}\mathcal{L}(\Psi(x; w, b), y) - \nabla_{b^{[i]}}\mathcal{L}(\Psi(x^*; w, b), y)||\big) \\ \leq& M \cdot ||\nabla_{w^{[i]}, b^{[i]}}\mathcal{L}\big(\Psi(x; w, b), y\big) - \nabla_{w^{[i]}, b^{[i]}}\mathcal{L}\big(\Psi(x^*; w, b), y\big)||,\end{aligned}$$

where we pick enough big number $M := M^{[i]} \cdot ||x^*|| \cdot ||\nabla_{b^{[i]}}\mathcal{L}^{-1}|| + 1$. □

Remark. *1) In the deep leakage approach [21], the recovery of initial image requires model parameters $\mathcal{W}$ and the corresponding gradients $\nabla\mathcal{W}$ such that a minimization of gradient differences $||\nabla\mathcal{W}' - \nabla\mathcal{W}||$ yields a recovery of initial image if the initial image exists. Our theorem provides sufficient conditions of the initial image existence, and Inequality (11) confirms the effectiveness of the deep leakage approach.*

2) Essentially, deep leakage approach is a trade-off computational technique for the matrix approach in the meaning that a loss in accuracy is trade-off with the existence of approximate solution by the optimization approach. Both approaches require model parameters $\mathcal{W}$ and the corresponding gradients $\nabla\mathcal{W}$.

3) If Jacobian matrix is not full-rank or $\nabla_{b^{[i]}}\mathcal{L}$ is singular, the inverse problem is ill-posed and a minimization of gradient differences might yield multiple solutions or an infeasibility which is observed as noisy images.

If assumptions in Theorem 5.3 are met, we pick an index set I from row index set of $B^{[i]}$ and $W^{[i]}$ such that the following linear equation is well-posed,

$$B_I \cdot x = W_I,$$

where $B_I := B_I^{[i]}$ and $W_I := W_I^{[i]}$.

Theorem 5.4. *Suppose there are perturbations E_B, E_W added on B_I, W_I, respectively, such that observed measurements $\bar{B}_I = B_I + E_B, \bar{W}_I = W_I + E_W$. Then, the* ***reconstruction*** *x^** ***of the initial input*** *x* ***can be determined*** *by solving a noisy linear system $\bar{B}_I \cdot x^* = \bar{W}_I$, provided that*

$$\|B_I^{-1} \cdot E_B\| < 1; \tag{13}$$

Moreover, the relative error is bounded,

$$\frac{\|x^* - x\|}{\|x\|} \leq \frac{\kappa(B_I)}{1 - \|B_I^{-1} \cdot E_B\|}\Big(\frac{\|E_B\|}{\|B_I\|} + \frac{\|E_W\|}{\|W_I\|}\Big), \tag{14}$$

in which B_I^{-1} is the inverse of B_I, where $\kappa(B_I)$ is the conditional number of B_I.

Proof. According to the construction, we have

$$(\bar{B}_I - B_I) \cdot x^* + B_I \cdot (x^* - x) = \bar{W}_I - W_I,$$

which yields

$$x^* - x = {B_I}^{-1} \cdot \big(\bar{W}_I - W_I - (\bar{B}_I - B_I) \cdot x^*\big). \tag{15}$$

Consider the relative error: since $||W_I|| \leq ||B_I|| \cdot ||x||$, Equation (15) becomes

$$\frac{||x^* - x||}{||x||} \leq \kappa(B_I) \cdot \Big(\frac{||E_B||}{||B_I||} \cdot \frac{||x^*||}{||x||} + \frac{||E_W||}{||W_I||}\Big), \tag{16}$$

where condition number $\kappa(B_I) := ||B_I|| \cdot ||{B_I}^{-1}||$.

Moreover, according to Lemma 5.2, we have

$$B_I \cdot (x - x^*) = E_B \cdot x^* - E_W.$$

A simplification of the above equation, we have

$$x + ({B_I}^{-1} \cdot E_B - I) \cdot x^* = -{B_I}^{-1} \cdot E_W.$$

Take a norm on both sides, we have

$$\|x\| + \|{B_I}^{-1} \cdot E_B - I\| \cdot \|x^*\| \geq 0.$$

Since $\|B_I^{-1} \cdot E_B\| < 1$, we have

$$\frac{\|x^*\|}{\|x\|} \leq \frac{1}{1 - \|{B_I}^{-1} \cdot E_B\|}. \tag{17}$$

Combine Equation (16) and Equation (17), we get Equation (14). □

Remark. $\|B_I^{-1} \cdot E_B\| < 1$ *alone is a* necessary *condition for the iterative reconstruction algorithm to converge. In other words, a big perturbation with $||E_B|| > ||B_I||$, such as Gaussian noise with a sufficiently big variance, is guaranteed to defeat reconstruction attacks like [21].*

Appendix B: Polarization Loss

Definition 2 (Polarization loss)**.** *For each data $\boldsymbol{x} \in \mathcal{X}$ and its corresponding output vector $\boldsymbol{v} := \Psi(\boldsymbol{x}; \boldsymbol{w}) \in \mathbb{R}^K$, the polarization loss is defined on the vector $\boldsymbol{v}$ with respect to a pre-set target binary code $\boldsymbol{t} \in \mathcal{H}$ as follows,*

$$\mathcal{L}_P(\boldsymbol{v}, \boldsymbol{t}) := \sum_{i=1}^{K} \max(m - v_i \cdot t_i, 0), \tag{18}$$

where the margin threshold is pre-set, $m \geq 1$, for the bound in Lemma 5.5 to be strict.

Lemma 5.5. *For output vector $\boldsymbol{v} = \Psi(\boldsymbol{x}; \boldsymbol{w})$, the Hamming distance $\mathcal{D}_h(\boldsymbol{b}, \boldsymbol{t}) := \frac{1}{2}(K - \boldsymbol{b} \cdot \boldsymbol{t})$ between K-bits binary hash code $\boldsymbol{b} = \texttt{Bin}(\boldsymbol{v})$ and the corresponding binary vector $\boldsymbol{t}$ is upper bounded by the polarization loss*

$$\mathcal{D}_h(\boldsymbol{b}, \boldsymbol{t}) \leq \mathcal{L}_P(\boldsymbol{v}, \boldsymbol{t}), \tag{19}$$

for any $m \geq 1$ and $\boldsymbol{v} \in \{(v_1, \cdots, v_K) \big| v_k \in \mathbb{R}\}$.

Proof. On one side, there are two cases for each coordinate of Hamming distance $\mathcal{D}_h(\mathbf{b}, \mathbf{t})$,

$$|b_i - t_i| \in \{0, 2\} \quad \text{if } v_i \cdot t_i > 0 \text{ or } v_i \cdot t_i \leq 0;$$

On the other side, both above cases are upper bounded by $\max(m - v_i \cdot t_i, 0)$ provided that if any $m \geq 1$.

Sum up the residues of each coordinate, we get this lemma. □

Proposition 5.1. *Suppose class $\mathcal{C}$ consists of data points $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{|\mathcal{C}|}\}$ associated with a pre-set target $\boldsymbol{t} \in \mathcal{H}$ in Hamming space. The averaged intra-class pairwise Hamming distances among the corresponding binary codes $\{\boldsymbol{b}_1, \cdots, \boldsymbol{b}_{|\mathcal{C}|} | \boldsymbol{b}_i = \boldsymbol{\Phi}(\boldsymbol{x}_i; \boldsymbol{w})\}$ is upper bounded by,*

$$\frac{1}{|\mathcal{C}|^2} \cdot \sum_{1 \leq i,j \leq |\mathcal{C}|} \mathcal{D}_h(\boldsymbol{b}_i, \boldsymbol{b}_j) \leq \frac{2}{|\mathcal{C}|} \cdot \sum_{1 \leq i \leq |\mathcal{C}|} \mathcal{L}_P(\boldsymbol{v}_i, \boldsymbol{t}). \tag{20}$$

Proof. According to Lemma 5.5 and the triangle law, we have

$$\begin{aligned}
\sum_{1 \leq i,j \leq |\mathcal{C}|} \mathcal{D}_h(\mathbf{b}_i, \mathbf{b}_j) &\leq \sum_{1 \leq i,j \leq |\mathcal{C}|} \mathcal{D}_h(\mathbf{b}_i, \mathbf{t}) + \mathcal{D}_h(\mathbf{b}_j, \mathbf{t}) \\
&\leq \sum_{1 \leq i,j \leq |\mathcal{C}|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_i, \mathbf{t}) + \mathcal{L}_{\mathrm{P}}(\mathbf{v}_j, \mathbf{t}) \\
&\leq 2|\mathcal{C}| \cdot \sum_{1 \leq i \leq |\mathcal{C}|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_i, \mathbf{t}).
\end{aligned}$$

Divide $|\mathcal{C}|^2$ on both sides, we get this proposition. □

Proposition 5.2. *Suppose there are L classes in the dataset, i.e. $\mathcal{C}_1, \cdots, \mathcal{C}_L$. For any two classes $\mathcal{C}_x$ and $\mathcal{C}_y$ $(1 \leq x \neq y \leq L)$, respectively, with associated targets binary vectors $\boldsymbol{t}_x$ and $\boldsymbol{t}_y$ and binary hash codes $\boldsymbol{b}_i^x = \boldsymbol{\Phi}(\mathbf{x}_i; \mathbf{w}), i \in \{1, \cdots, |\mathcal{C}_x|\}$, $\boldsymbol{b}_i^y = \boldsymbol{\Phi}(\mathbf{y}_j; \mathbf{w}), j \in \{1, \cdots, |\mathcal{C}_y|\}$, the averaged inter-class pairwise Hamming distances among binary codes $\sum_{\substack{1 \leq i \leq |\mathcal{C}_x|, \\ 1 \leq j \leq |\mathcal{C}_y|}} \mathcal{D}_h(\boldsymbol{b}_i^x, \boldsymbol{b}_j^y)$ is lower bounded by,*

$$\sum_{1 \leq x \neq y \leq L} \Big(\mathcal{D}_h(\boldsymbol{t}_x, \boldsymbol{t}_y) - \frac{1}{|\mathcal{C}_x| \cdot |\mathcal{C}_y|} \cdot \sum_{\substack{1 \leq i \leq |\mathcal{C}_x|, \\ 1 \leq j \leq |\mathcal{C}_y|}} \mathcal{D}_h(\boldsymbol{b}_i^x, \boldsymbol{b}_j^y) \Big) \leq \sum_{1 \leq x \leq L} \frac{2 \cdot (L-1)}{|\mathcal{C}_x|} \cdot \sum_{1 \leq i \leq |\mathcal{C}_x|} \mathcal{L}_P(\boldsymbol{v}_i^x, \boldsymbol{t}_x). \tag{21}$$

Proof. By the triangle law, we have

$$\mathcal{D}_h(\mathbf{t}_x, \mathbf{t}_y) \leq \mathcal{D}_h(\mathbf{t}_x, \mathbf{b}_i^x) + \mathcal{D}_h(\mathbf{b}_i^x, \mathbf{b}_j^y) + \mathcal{D}_h(\mathbf{b}_j^y, \mathbf{t}_y)$$

. Fix x, y and sum over i, j on both sides, we have

$$\begin{aligned}
|\mathcal{C}_x| \cdot |\mathcal{C}_y| \cdot \mathcal{D}_h(\mathbf{t}_x, \mathbf{t}_y) - \sum_{\substack{1 \leq i \leq |\mathcal{C}_x|, \\ 1 \leq j \leq |\mathcal{C}_y|}} \mathcal{D}_h(\mathbf{b}_i^x, \mathbf{b}_j^y) &\leq |\mathcal{C}_y| \cdot \sum_{1 \leq i \leq |\mathcal{C}_x|} \mathcal{D}_h(\mathbf{t}_x, \mathbf{b}_i^x) + |\mathcal{C}_x| \cdot \sum_{1 \leq j \leq |\mathcal{C}_y|} \mathcal{D}_h(\mathbf{b}_j^y, \mathbf{t}_y) \\
&\leq |\mathcal{C}_y| \cdot \sum_{1 \leq i \leq |\mathcal{C}_x|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_i^x, \mathbf{t}_x) + |\mathcal{C}_x| \cdot \sum_{1 \leq j \leq |\mathcal{C}_y|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_j^y, \mathbf{t}_y).
\end{aligned}$$

Divide $|\mathcal{C}_x| \cdot |\mathcal{C}_y|$ and sum over x, y on both sides, we have

$$\begin{aligned}
&\sum_{1\le x\neq y\le L} \Big(\mathcal{D}_h(\mathbf{t}_x, \mathbf{t}_y) - \frac{1}{|\mathcal{C}_x| \cdot |\mathcal{C}_y|} \cdot \sum_{\substack{1\le i\le |\mathcal{C}_x|,\\ 1\le j\le |\mathcal{C}_y|}} \mathcal{D}_h(\mathbf{b}_i^x, \mathbf{b}_j^y)\Big) \\
\le &\sum_{1\le x\neq y\le L} \frac{1}{|\mathcal{C}_x|} \cdot \sum_{1\le i\le |\mathcal{C}_x|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_i^x, \mathbf{t}_x) + \frac{1}{|\mathcal{C}_y|} \cdot \sum_{1\le j\le |\mathcal{C}_y|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_j^y, \mathbf{t}_y) \\
\le &\sum_{1\le x\le L} \frac{2\cdot(L-1)}{|\mathcal{C}_x|} \cdot \sum_{1\le i\le |\mathcal{C}_x|} \mathcal{L}_{\mathrm{P}}(\mathbf{v}_i^x, \mathbf{t}_x).
\end{aligned}$$

□

Proposition 5.3. *The difference between averaged intra-class pairwise Hamming distance and averaged inter-class pairwise Hamming distance is upper bounded, i.e.*

$$\begin{aligned}
&\sum_{1\le x\le L} \frac{1}{|\mathcal{C}_x|^2} \cdot \sum_{1\le i,j\le |\mathcal{C}_x|} \mathcal{D}_h(\boldsymbol{b}_i^x, \boldsymbol{b}_j^x) - \sum_{1\le x\neq y\le L} \frac{1}{|\mathcal{C}_x| \cdot |\mathcal{C}_y|} \sum_{\substack{1\le i\le |\mathcal{C}_x|,\\ 1\le j\le |\mathcal{C}_y|}} \mathcal{D}_h(\boldsymbol{b}_i^x, \boldsymbol{b}_j^y) \\
\le &\sum_{1\le x\le L} \frac{2\cdot L}{|\mathcal{C}_x|} \cdot \sum_{1\le i\le |\mathcal{C}_x|} \mathcal{L}_P(\boldsymbol{v}_i^x, \boldsymbol{t}_x) - \sum_{1\le x\neq y\le L} \mathcal{D}_h(\boldsymbol{t}_x, \boldsymbol{t}_y).
\end{aligned} \tag{22}$$

Proof. By Lemma 5.1 and Lemma 5.2, we directly get this proposition. □

Remark. *1) Inequality in (Eq. 20) shows that the averaged polarization loss is a strict upper-bound of the averaged pairwise Hamming distances between points of the same class. That is to say, minimizing the RHS of (Eq. 20) effectively minimizes the averaged intra-class pairwise Hamming distances.*

2) In terms of the computational complexity, pairwise Hamming distances on the LHS of (Eq. 20) is $O(|\mathcal{C}|^2)$ while the polarization loss on the RHS of (Eq. 20) is $O(|\mathcal{C}|)$ only.

3) Inequality in (Eq. 21) shows that minimizing polarization losses on the RHS of (Eq. 21) effectively maximizes the averaged inter-class pair-wised Hamming distances on LHS.

4) According to Proposition 5.3, the optimization problem of simultaneous minimizing the intra-class and maximizing inter-class Hamming distances, i.e.

$$\min_{\boldsymbol{w}} \sum_{1\le x\le L} \frac{1}{|\mathcal{C}_x|^2} \cdot \sum_{1\le i,j\le |\mathcal{C}_x|} \mathcal{D}_h(\boldsymbol{b}_i^x, \boldsymbol{b}_j^x) - \sum_{1\le x\neq y\le L} \frac{1}{|\mathcal{C}_x| \cdot |\mathcal{C}_y|} \sum_{\substack{1\le i\le |\mathcal{C}_x|,\\ 1\le j\le |\mathcal{C}_y|}} \mathcal{D}_h(\boldsymbol{b}_i^x, \boldsymbol{b}_j^y),$$

is equivalent to the problem of minimizing the averaged polarization loss over the whole data set, i.e.

$$\min_{\boldsymbol{w}} \sum_{1\le x\le L} \frac{1}{|\mathcal{C}_x|} \cdot \sum_{1\le i\le |\mathcal{C}_x|} \mathcal{L}_P(\boldsymbol{v}_i^x, \boldsymbol{t}_x).$$

Appendix C: Experiment Setup

Dataset

In our experiments, we used MNIST, CIFAR10, CIFAR100 and SVHN, which are used in previous PPDL studies.

Network Architecture

In our experiments, we used AlexNet, VGG16 and DLNet (from [21]).

For AlexNet and VGG16, we slightly modified the architecture implementation from *torchvision*[10] package to adapt a 32×32 input. In VGG16, we added Group Normalization after every Convolution layer. (See Table 4 and 5)

For privacy attack analysis, we used network architecture (DLNet) from released code [11].

Privacy-Preserving Mechanisms

For **DP**, we are using implementation from *pytorch-dp* package. Slightly modified to adapt to privacy attack analysis. (We disabled the gradient clipping function.)

For **PPDL**, we reimplemented using reference from author released code [12].

For our **SPN**, we used $\alpha_1 = 1$ in all of our experiments. We random initialize the private target $\boldsymbol{t}$, and using 64-bit in all of our experiments. (See Algorithm 1)

Privacy Attacks

For **reconstruction attacks**, we adopt author released code [11] to reconstruct images. We follow their implementation which we random initialized the model for reconstruction attack. (See Algorithm 5)

For **membership attacks**, we are using same algorithm from reconstruction attacks. (See Algorithm 5)

For **tracing attacks**, first, we perform reconstruction attacks to recovered X number of images, we used $X = 1000$ in our experiments. Then we separated reconstructed images into N partitions simulating N participants, we used $N = 10$ in our experiments. During tracing, we trace the query image from the reconstructed dataset. The query images is the dataset that used for reconstruction attacks. We are using full query dataset (e.g. 50000 images for CIFAR10) for tracing. (See Algorithm 6)

Federated Learning.

The federated learning environment is run with both IID and Non-IID dataset. For IID case, we uniformly split training datasets into N partitions (with same number of data per class), respectively, for N participants, and use all testing datasets for evaluation of the global model performances. For Non-IID dataset, we follows their implementation [13] to separate the dataset into N participants using Dirichlet distribution with $\alpha = 0.9$.

For DLNet, we are using round robin for model aggregation following implementation from [12] (See Algorithm 4). Otherwise, we are using FedAvg algorithm for model aggregation, which is following the implementation in [11] and using source code from this [14] GitHub repository as reference.

Table 2 and 3 summarized the hyperparameters we used in this paper.

[10] https://pytorch.org/docs/stable/torchvision/models.html
[11] https://github.com/mit-han-lab/dlg
[12] https://www.comp.nus.edu.sg/~reza/files/PPDL.zip
[13] https://github.com/ebagdasa/backdoor_federated_learning
[14] https://github.com/shaoxiongji/federated-learning

Hyperparameter	Privacy Attack Analysis
Training Hyperparameters	
Dataset	MNIST, CIFAR10, CIFAR100, SVHN
Network Architecture	DLNet [21]
Weight Initialization	$uniform(-0.3, 0.3)$
Optimization method	Adam
Optimizer Hyperparameter	Adam ($\beta_1 = 0.9, \beta_2 = 0.999$)
Learning rate	0.001
Learning rate decay	No decay
Batch size	32
Local Epochs/Global Communication Rounds	1/300
Number of Clients	10
Privacy-Preserving Hyperparameters	
SPN number of bit	64
SPN α_2	0.0001, 0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5
PPDL shared percentage	5%, 30%
DP noise σ	0.0001, 0.001, 0.01, 0.1, 0.5
Deep Leakage Attack Hyperparameters	
Attack Batch Size	1, 4, 8
SPN α_2	0.0001, 0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5
PPDL shared percentage	5%, 30%
DP noise σ	0.0001, 0.001, 0.01, 0.1, 0.5

Table 2: Hyperparameters used in our privacy attack analysis.

Hyperparameter	Federated Learning
Dataset	MNIST, CIFAR10, CIFAR100, SVHN
Network Architecture	AlexNet, VGG16
Weight Initialization	$kaiming\_uniform$
Optimization method	SGD
Optimizer Hyperparameter	Momentum = 0.9
Learning rate	0.01, 0.001
Learning rate decay	Decay by factor of 0.5 at round 100 and 200
Batch size	64
Local Epochs/Global Communication Rounds	1/300
Number of Clients	8, 20, 50, 100
Privacy-Preserving Hyperparameters	
SPN number of bit	64
SPN α_2	0.1
PPDL shared percentage	5%, 30%
DP noise σ	0.1

Table 3: Hyperparameters used in our Federated Learning Task experiments.

layer name	output size	weight shape	padding
Conv1	32×32	$64 \times 3 \times 5 \times 5$	2
MaxPool2d	16×16	2×2	
Conv2	16×16	$192 \times 64 \times 5 \times 5$	2
Maxpool2d	8×8	2×2	
Conv3	8×8	$384 \times 192 \times 3 \times 3$	1
Conv4	8×8	$256 \times 384 \times 3 \times 3$	1
Conv5	8×8	$256 \times 256 \times 3 \times 3$	1
MaxPool2d	4×4	2×2	
Linear	256	256×4096	
Linear	10	10×256	

Table 4: Modified AlexNet

layer name	output size	weight shape	padding
Conv1-GN $\times$ 2	32×32	$64 \times 64 \times 3 \times 3$	1
MaxPool2d	16×16	2×2	
Conv2-GN $\times$ 2	16×16	$128 \times 128 \times 3 \times 3$	1
Maxpool2d	8×8	2×2	
Conv3-GN $\times$ 3	8×8	$256 \times 256 \times 3 \times 3$	1
Maxpool2d	8×8	2×2	
Conv4-GN $\times$ 3	8×8	$512 \times 512 \times 3 \times 3$	1
Maxpool2d	8×8	2×2	
Conv5-GN $\times$ 3	8×8	$512 \times 512 \times 3 \times 3$	1
MaxPool2d	4×4	2×2	
Linear	256	256×4096	
Linear	10	10×256	

Table 5: Modified VGG16

layer name	output size	weight shape	padding	stride
Conv1	16×16	$12 \times 3 \times 5 \times 5$	2	2
Conv2	8×8	$12 \times 12 \times 5 \times 5$	2	2
Conv3	8×8	$12 \times 12 \times 5 \times 5$	2	1
Conv4	8×8	$12 \times 12 \times 5 \times 5$	2	1
Linear	10	10×768		

Table 6: DLNet from [21]

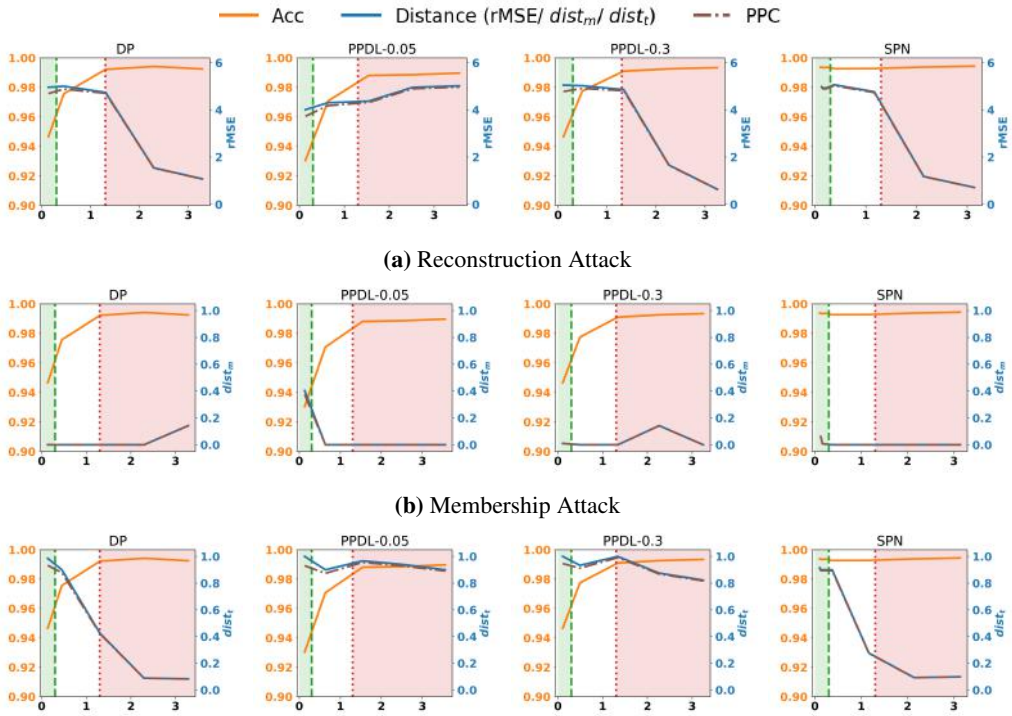
Appendix D: Privacy-Preserving Capability

In this section, we show experiment results of Privacy-Preserving Characteristics (PPC) and Calibrated Averaged Performance (CAP) for different dataset and different attack batch size. During our experiment, we found out that Sigmoid activation layer is having gradient vanishing problem, causing difficulty in training model on SVHN. Therefore, we replace Sigmoid with Tanh activation layer specifically for SVHN to measure PPC and CAP, while MNIST, CIFAR10 and CIFAR100 are measured with Sigmoid.

To measure PPC and CAP, we are using model trained on 10 clients using Federated Averaged algorithm as mentioned in Appendix C with batch size of 32. We are using IID dataset which we uniformly split into 10 clients.

5.1 MNIST

5.1.1 Privacy-Preserving Characteristics (PPC)

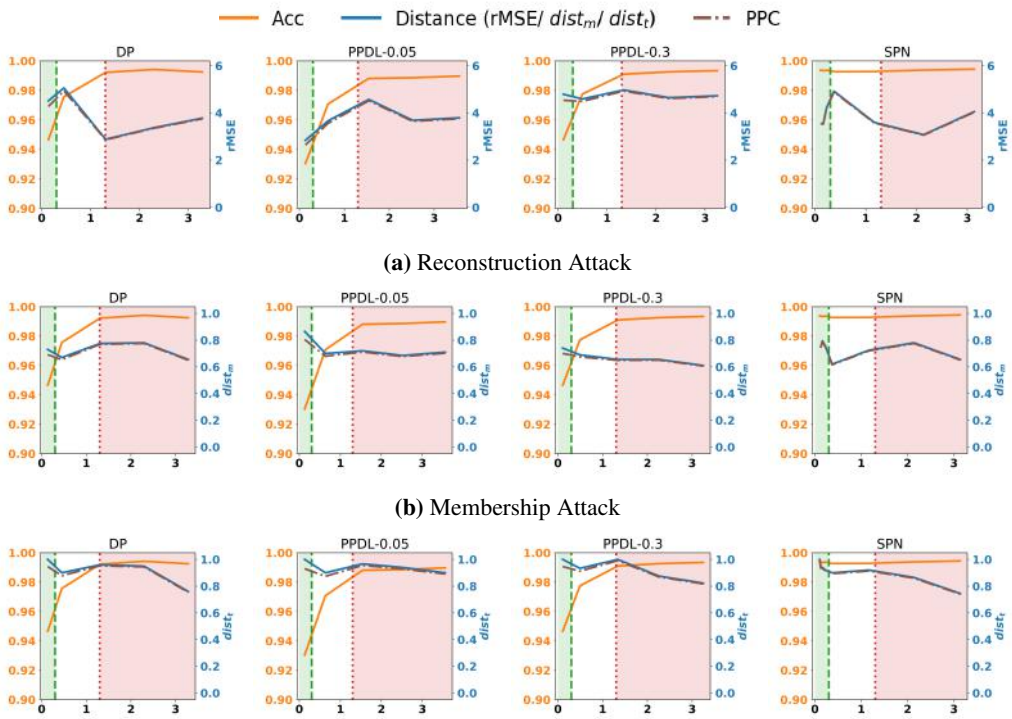


(a) Reconstruction Attack

(b) Membership Attack

(c) Tracing Attack

Figure 9: Attack with Batch Size 1

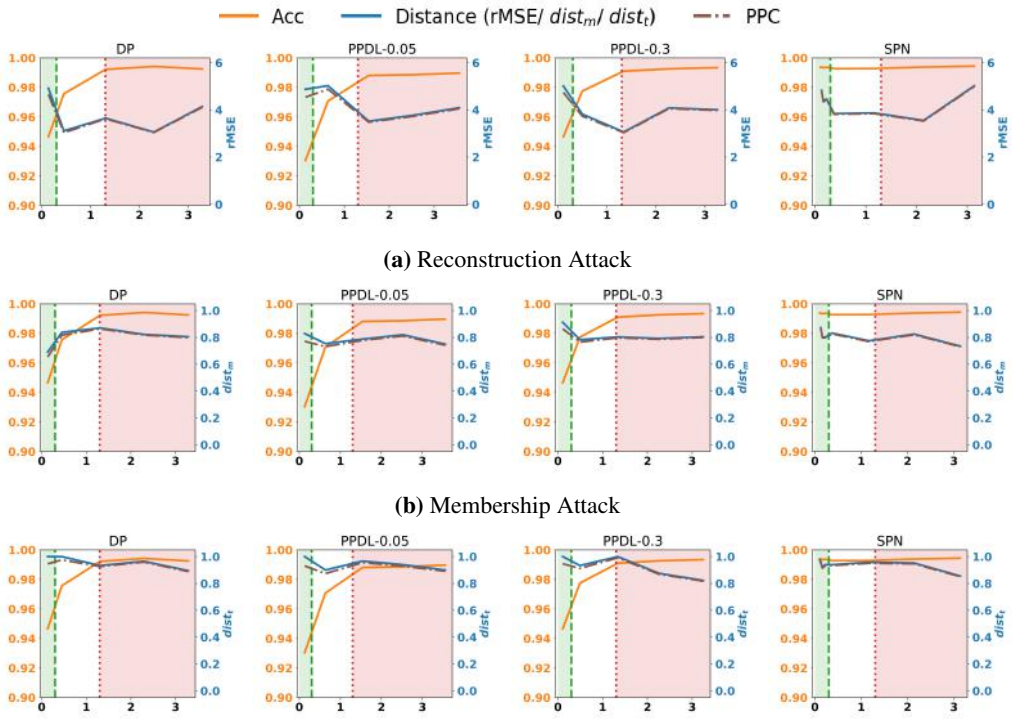


(a) Reconstruction Attack

(b) Membership Attack

(c) Tracing Attack

Figure 10: Attack with Batch Size 4



(a) Reconstruction Attack

(b) Membership Attack

(c) Tracing Attack

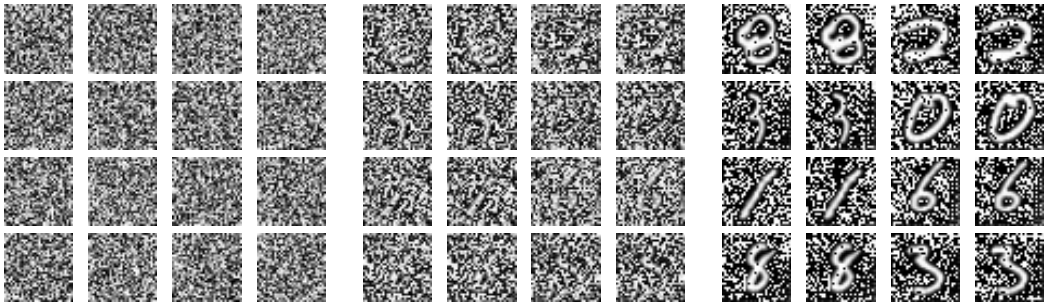
Figure 11: Attack with Batch Size 8

5.1.2 Calibrated Averaged Performance (CAP)

	Reconstruction			Membership			Tracing		
BS	1	4	8	1	4	8	1	4	8
DP [1]	3.38	**3.83**	3.69	0.00	0.71	0.79	0.91	0.93	**0.95**
PPDL-0.05 [15]	**4.42**	3.62	4.13	0.37	0.72	0.77	0.92	0.92	0.92
PPDL-0.3 [15]	4.04	4.65	3.91	0.00	0.66	0.80	**0.95**	**0.95**	**0.95**
SPN (ours)	4.30	3.72	**4.33**	**0.37**	**0.73**	**0.81**	0.90	0.93	0.94

Table 7: CAP performance with different batch size on MNIST for reconstruction, membership, and tracing attack. Higher better. BS = Attack Batch Size.

5.1.3 Reconstructed Images

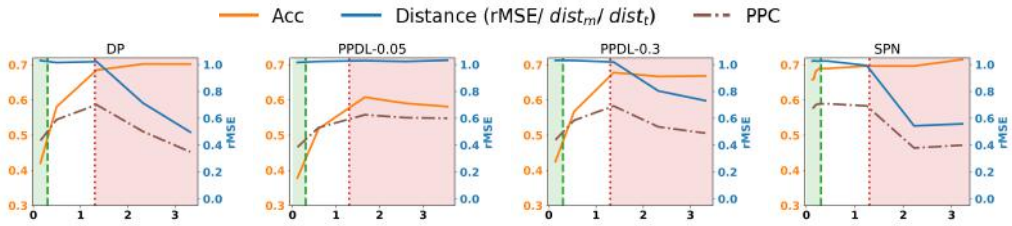


(a) $5.17(0.37)$ **(b)** $4.70(16.89)$ **(c)** $1.46(1011.30)$

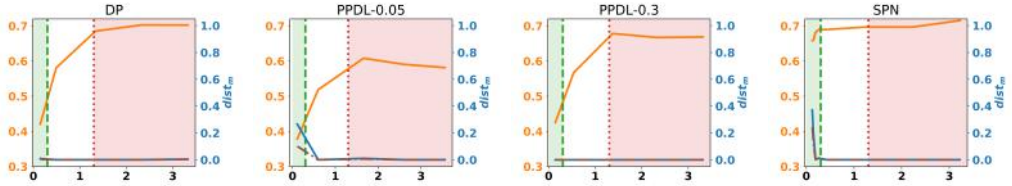
Figure 12: Reconstructed images from different region mentioned in the main paper. **(a)** Green region **(b)** White region **(c)** Red region. Values inside bracket are mean of $\frac{||B_I||}{||E_B||}$ and values outside are mean of rMSE of reconstructed w.r.t. original images. High rMSE (e.g. 1.46 in the red region) is due to original image is having a lot of zero valued pixel, hence getting smaller $||x||$ and higher rMSE. Also, found out that images that have solid color pixels (e.g. fully dark (0,0,0) or fully white (255,255,255)) are more difficult to attack.

5.2 CIFAR10

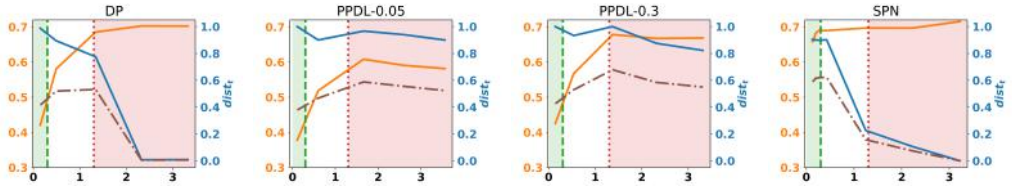
5.2.1 Privacy-Preserving Characteristics (PPC)



(a) Reconstruction Attack

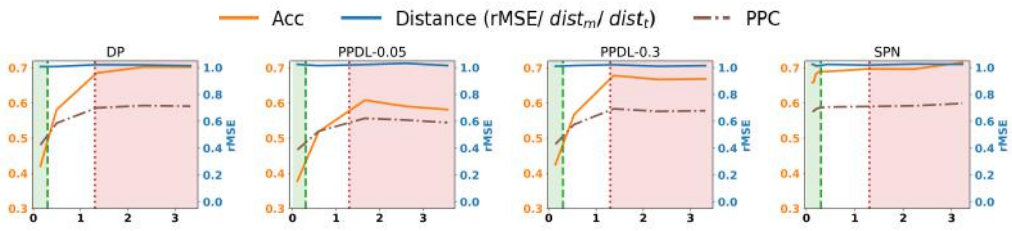


(b) Membership Attack

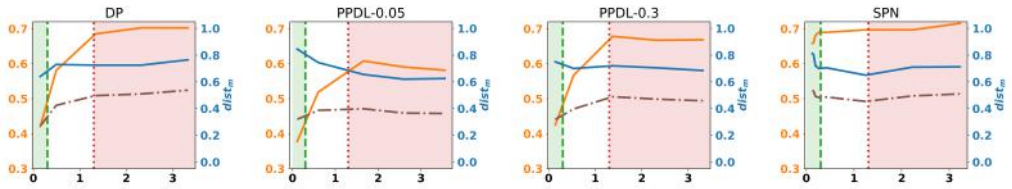


(c) Tracing Attack

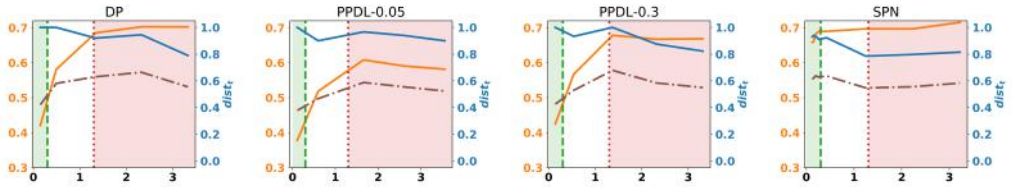
Figure 13: Attack with Batch Size 1



(a) Reconstruction Attack

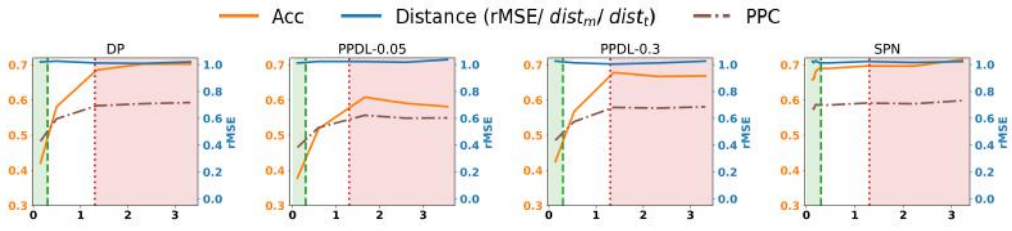


(b) Membership Attack

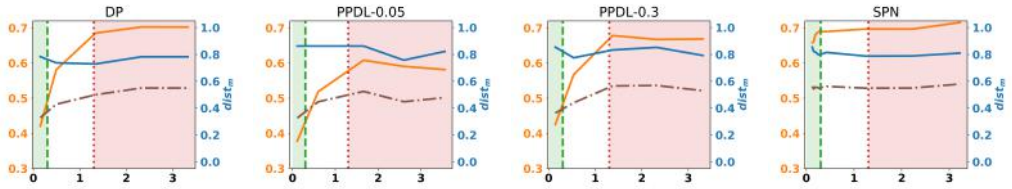


(c) Tracing Attack

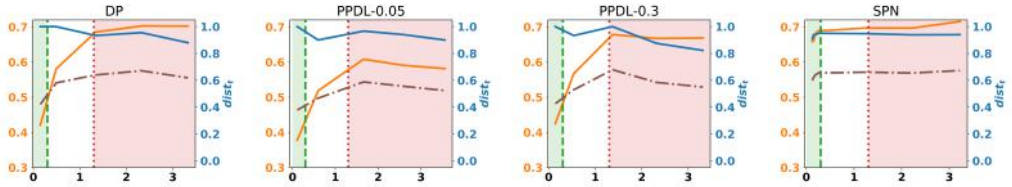
Figure 14: Attack with Batch Size 4



(a) Reconstruction Attack



(b) Membership Attack



(c) Tracing Attack

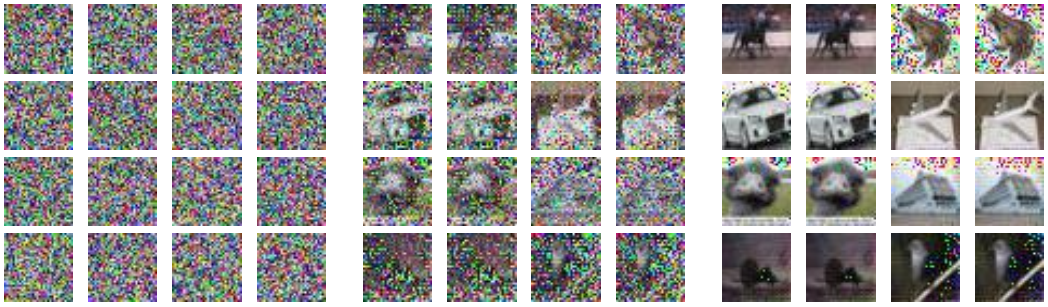
Figure 15: Attack with Batch Size 8

5.2.2 Calibrated Averaged Performance (CAP)

	Reconstruction			Membership			Tracing		
BS	1	4	8	1	4	8	1	4	8
DP [1]	0.57	0.63	0.63	0.00	0.45	0.47	0.42	0.57	0.58
PPDL-0.05 [15]	0.55	0.55	0.55	0.00	0.37	0.44	0.50	0.50	0.50
PPDL-0.3 [15]	0.57	0.61	0.61	0.00	0.43	0.49	0.54	0.54	0.54
SPN (ours)	**0.69**	**0.70**	**0.70**	**0.24**	**0.50**	**0.56**	**0.61**	**0.63**	**0.64**

Table 8: CAP performance with different batch size on CIFAR10 for reconstruction, membership, and tracing attack. Higher better. BS = Attack Batch Size.

5.2.3 Reconstructed Images

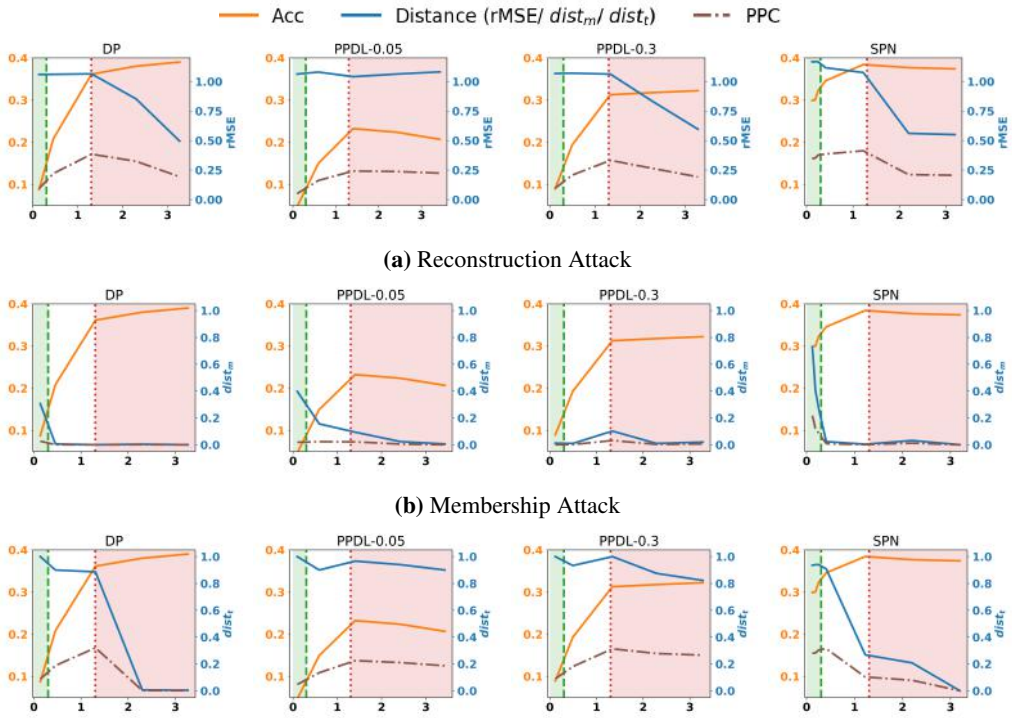


(a) $1.05(0.36)$ **(b)** $1.05(18.44)$ **(c)** $0.48(1106.17)$

Figure 16: Reconstructed images from different region mentioned in the main paper. **(a)** Green region **(b)** White region **(c)** Red region. Values inside bracket are mean of $\frac{||B_I||}{||E_B||}$ and values outside are mean of rMSE of reconstructed w.r.t. original images. Noted that some images are having solid color pixels (e.g. the frog image), causing difficulty in reconstruction (e.g. noise pixel in the white solid color region), which is similar to MNIST dataset.

5.3 CIFAR100

5.3.1 Privacy-Preserving Characteristics (PPC)

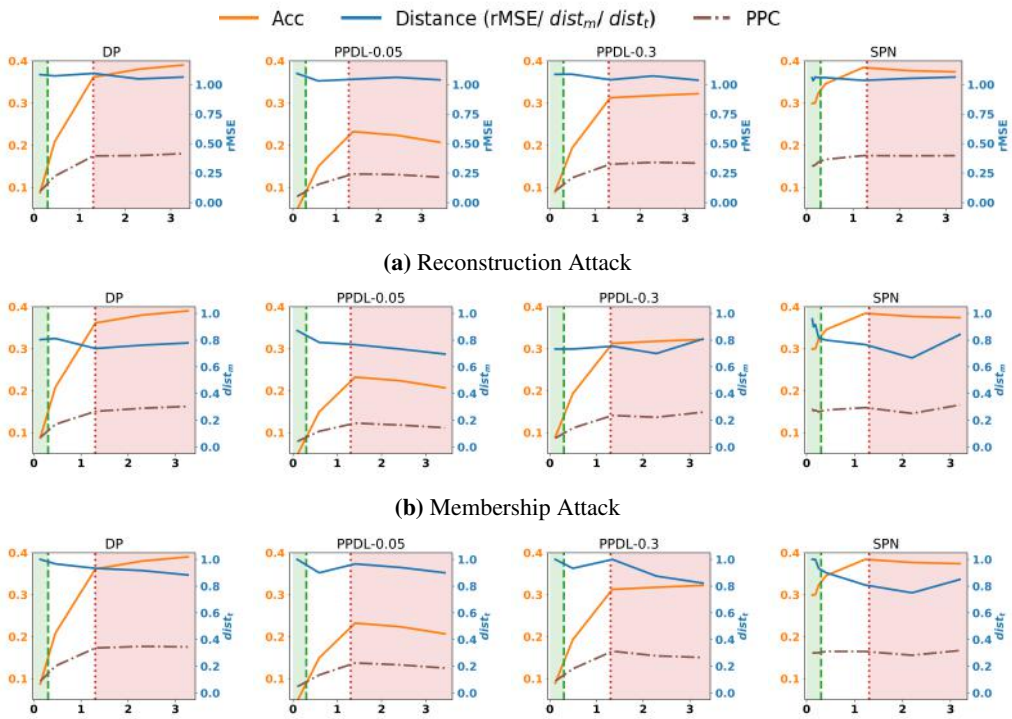


(a) Reconstruction Attack

(b) Membership Attack

(c) Tracing Attack

Figure 17: Attack Batch Size 1

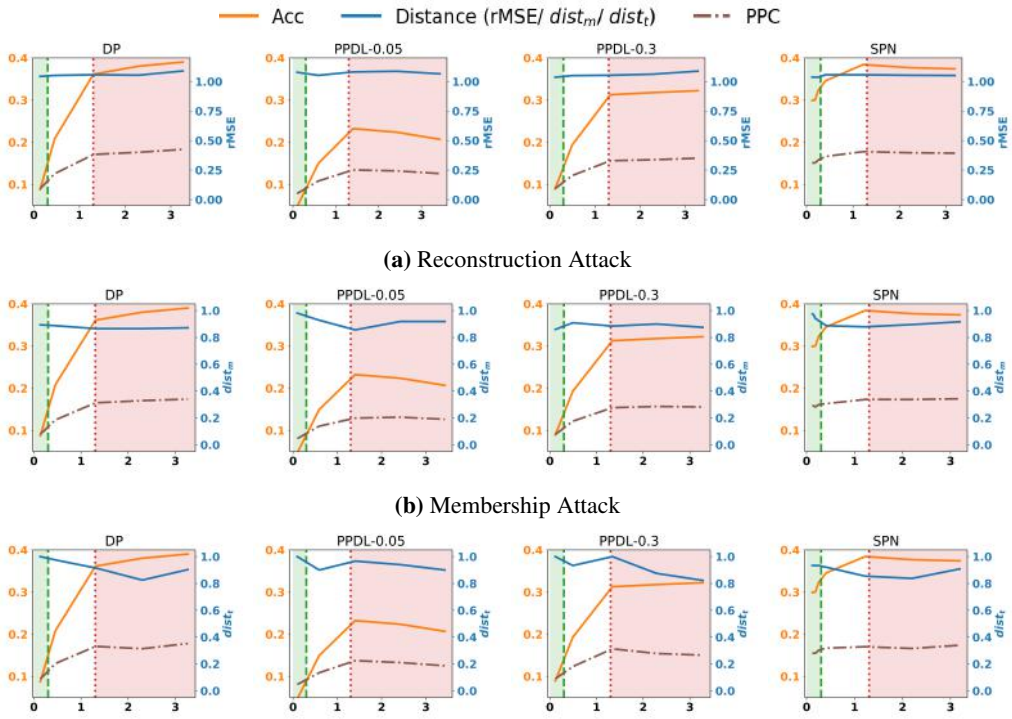


(a) Reconstruction Attack

(b) Membership Attack

(c) Tracing Attack

Figure 18: Attack with Batch Size 4



(a) Reconstruction Attack

(b) Membership Attack

(c) Tracing Attack

Figure 19: Attack with Batch Size 8

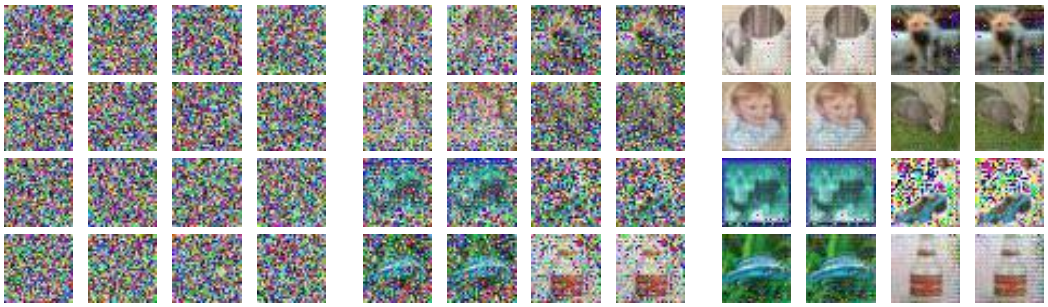
Figure 13, 14, 15 are privacy-preserving characteristics (PPC) with different attacks on CIFAR100.

5.3.2 Calibrated Averaged Performance (CAP)

	Reconstruction			Membership			Tracing		
BS	1	4	8	1	4	8	1	4	8
DP [1]	0.23	0.31	0.30	0.03	0.22	0.25	0.14	0.24	0.24
PPDL-0.05 [15]	0.18	0.18	0.18	0.02	0.13	0.16	0.16	0.16	0.16
PPDL-0.3 [15]	0.21	0.26	0.26	0.00	0.19	0.22	0.19	0.19	0.19
SPN (ours)	**0.37**	**0.36**	**0.35**	**0.17**	**0.28**	**0.31**	**0.29**	**0.30**	**0.30**

Table 9: CAP performance with different batch size on CIFAR100 for reconstruction, membership and tracing attack. Higher better. BS = Attack Batch Size.

5.3.3 Reconstructed Images

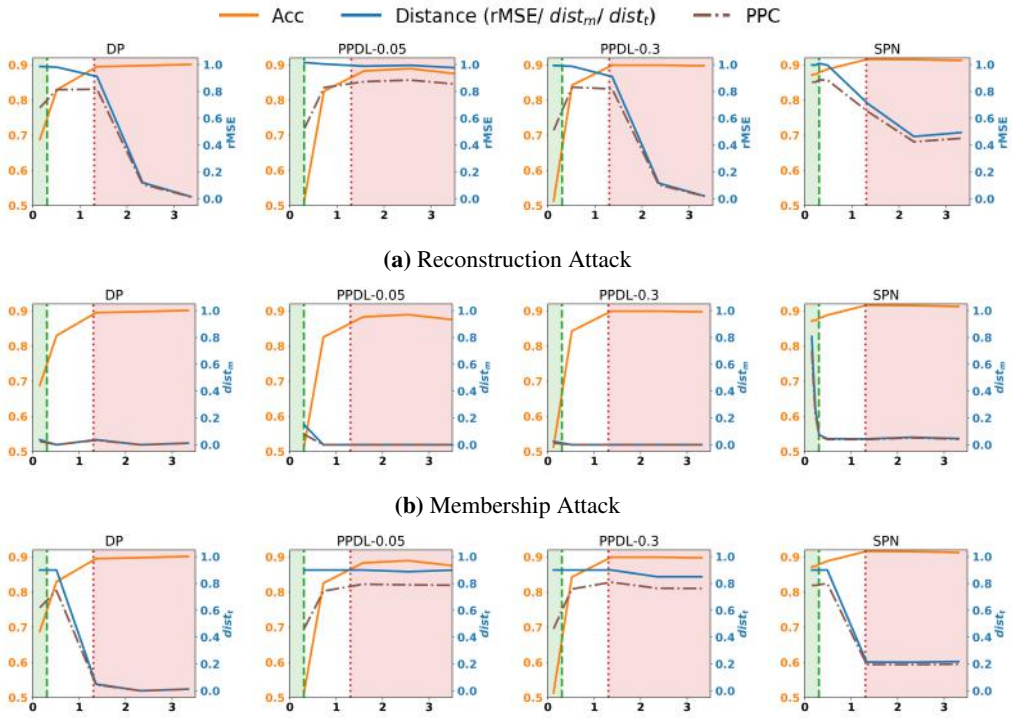


(a) $1.12(0.32)$ **(b)** $1.11(23.63)$ **(c)** $0.77(876.73)$

Figure 20: Reconstructed images from different region mentioned in the main paper. **(a)** Green region **(b)** White region **(c)** Red region. Values inside bracket are mean of $\frac{||B_I||}{||E_B||}$ and values outside are mean of rMSE of reconstructed w.r.t. original images.

5.4 SVHN

5.4.1 Privacy-Preserving Characteristics (PPC)

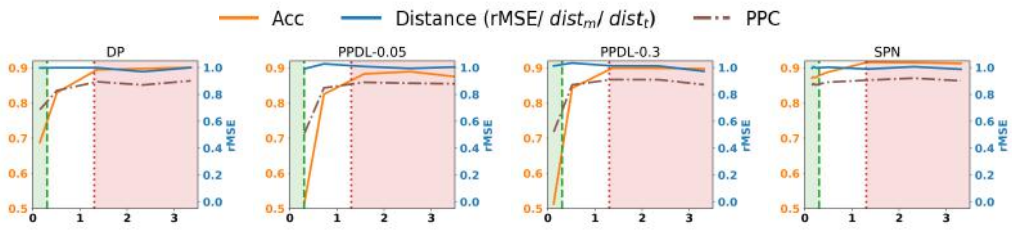


(a) Reconstruction Attack

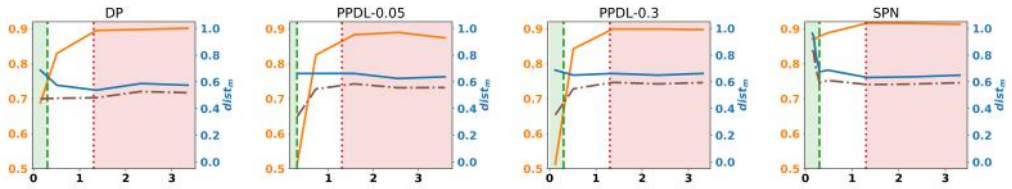
(b) Membership Attack

(c) Tracing Attack

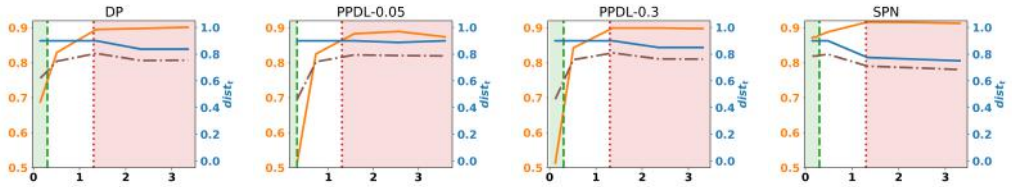
Figure 21: Attack Batch Size 1



(a) Reconstruction Attack

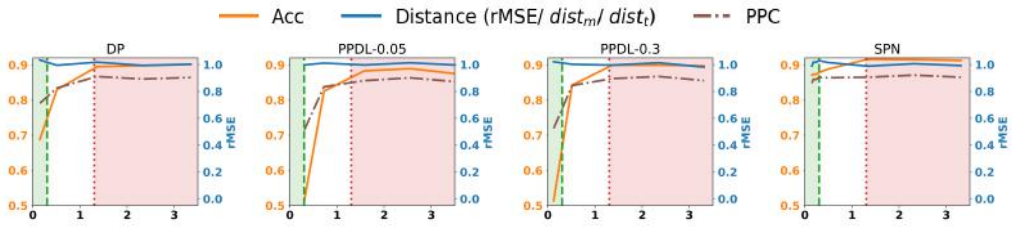


(b) Membership Attack

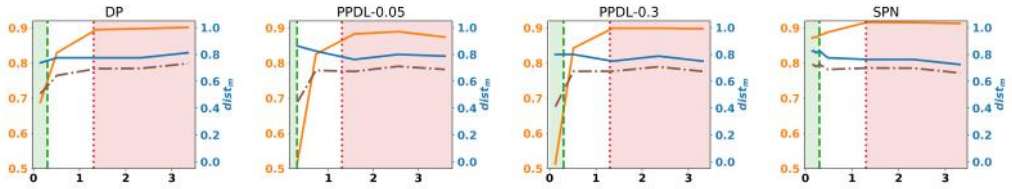


(c) Tracing Attack

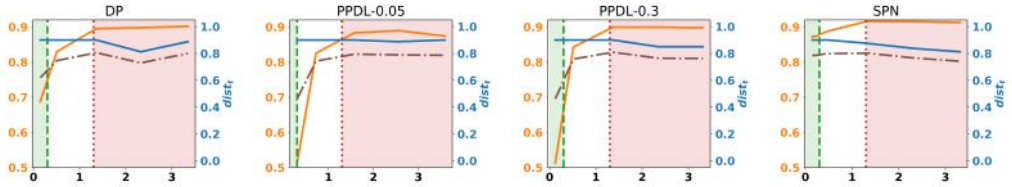
Figure 22: Attack with Batch Size 4



(a) Reconstruction Attack



(b) Membership Attack



(c) Tracing Attack

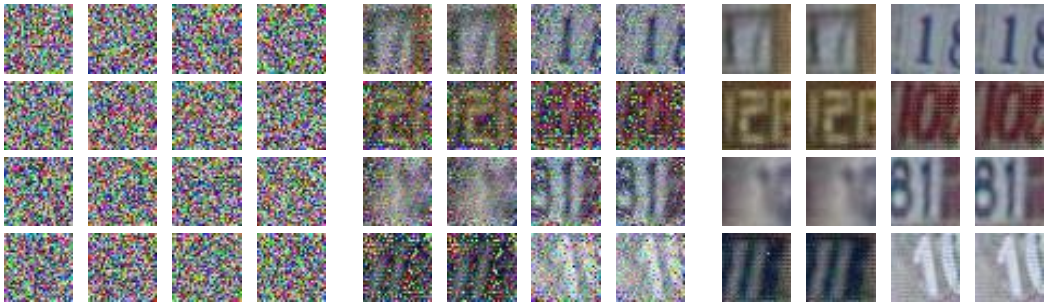
Figure 23: Attack with Batch Size 8

5.4.2 Calibrated Averaged Performance (CAP)

	Reconstruction			Membership			Tracing		
BS	1	4	8	1	4	8	1	4	8
DP [1]	0.77	0.84	0.85	0.00	0.50	0.65	0.68	0.72	0.72
PPDL-0.05 [15]	0.79	0.80	0.80	0.00	0.52	0.64	**0.70**	0.70	0.70
PPDL-0.3 [15]	0.72	0.82	0.81	0.00	0.54	0.63	0.68	0.68	0.68
SPN (ours)	**0.88**	**0.89**	**0.90**	**0.60**	**0.66**	**0.70**	0.66	**0.79**	**0.79**

Table 10: CAP performance with different batch size on SVHN for reconstruction, membership and tracing attack. Higher better. BS = Attack Batch Size.

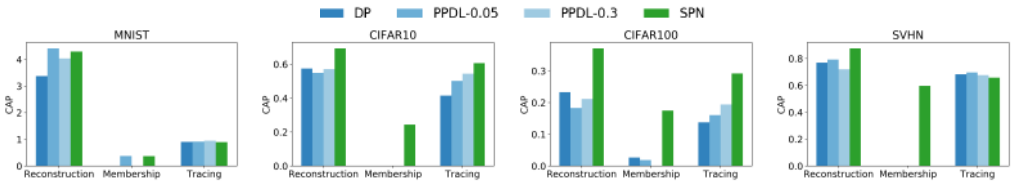
5.4.3 Reconstructed Images



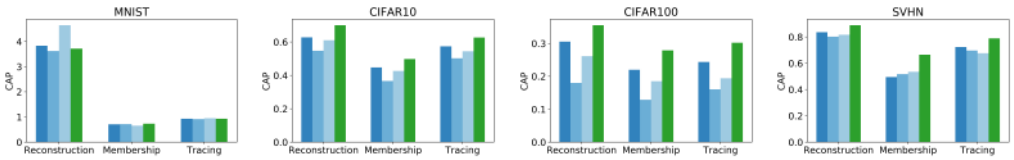
(a) $0.99(0.39)$ **(b)** $0.93(20.86)$ **(c)** $0.35(1251.48)$

Figure 24: Reconstructed images from different region mentioned in the main paper. **(a)** Green region **(b)** White region **(c)** Red region. Values inside bracket are mean of $\frac{||B_I||}{||E_B||}$ and values outside are mean of rMSE of reconstructed w.r.t. original images.

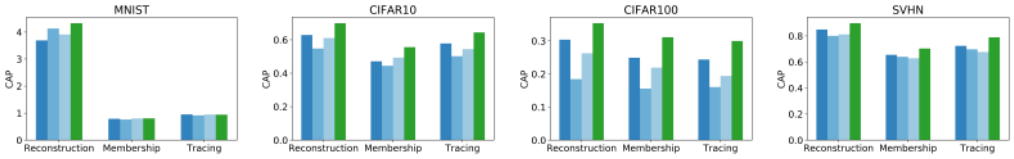
5.5 Summary of Calibrated Averaged Performance



(a) Attack Batch Size 1. Left to Right: MNIST, CIFAR10, CIFAR100 and SVHN



(b) Attack Batch Size 4. Left to Right: MNIST, CIFAR10, CIFAR100 and SVHN



(c) Attack Batch Size 8. Left to Right: MNIST, CIFAR10, CIFAR100 and SVHN.

Figure 25: Comparison of Calibrated Averaged Performances (CAPs) for the proposed SPN, PPDL [15] and DP [1] methods, against reconstruction, membership and tracing attacks (CAP the higher the better, see threat model and evaluation protocol in main paper).

Appendix E: Ablation Studies

5.6 Replace Gaussian Noise with Laplacian Noise

In this section, we replace Gaussian Noise with Laplacian Noise. For Laplcian noise and Gaussian noise, the scales we used are $\{0.5, 0.1, 0.01, 0.001, 0.0001\}$.

In Figure 26, Laplacian noise and Gaussian noise with the same scale are having almost identical $\frac{||B_I||}{||E_B||}$ and protection strength (i.e. rMSEs at different $\frac{||B_I||}{||E_B||}$ are almost the same).

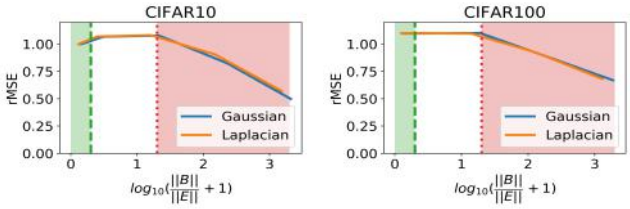


Figure 26: Left: CIFAR10; **Right**: CIFAR100.

5.7 Effect of Number of Bits in SPN

In this section, we shows that with number of bits in SPN will affect $\frac{||B_I||}{||E_B||}$ and hence improves the protection against reconstruction attack.

From 32-bit to 128-bit, $\frac{||B_I||}{||E_B||}$ increased as shown in Figure 27, protection strength (i.e. rMSE) is also increased.

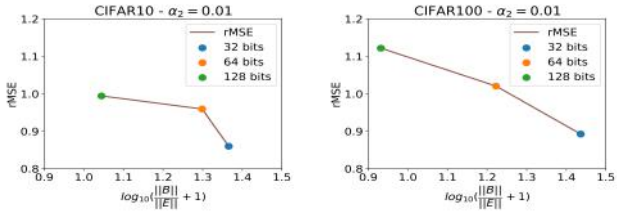


Figure 27: Left: CIFAR10; **Right**: CIFAR100.

Appendix F: Federated Learning

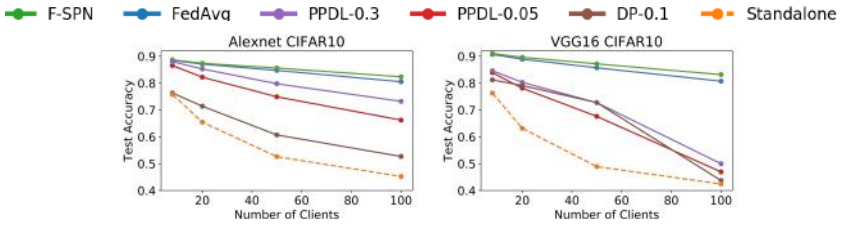
5.8 Accuracies for Privacy Attack Analysis

Table 11 shows accuracies of different privacy-preserving mechanisms using DLNet as network architecture and using round robin for model aggregation. *Accuracies are measured using test dataset on server model.* For SPN, we are using 64-bit.

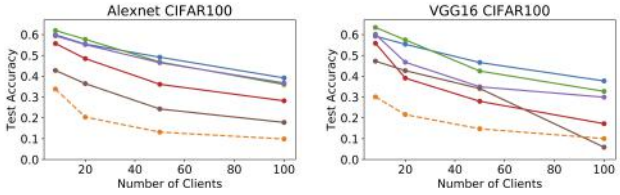
	MNIST	CIFAR10	CIFAR100	SVHN
DP-0.5	0.9466	0.4205	0.0876	0.6882
DP-0.1	0.9757	0.5808	0.2084	0.8295
DP-0.01	0.9922	0.6853	0.3614	0.8950
DP-0.001	0.9942	0.7027	0.3803	0.8980
DP-0.0001	0.9925	0.7025	**0.3902**	0.9015
PPDL-0.05, DP-0.5	0.9305	0.3783	0.0469	0.5151
PPDL-0.05, DP-0.1	0.9708	0.5184	0.1497	0.8254
PPDL-0.05, DP-0.01	0.9881	0.6081	0.2323	0.8829
PPDL-0.05, DP-0.001	0.9886	0.5906	0.2237	0.8894
PPDL-0.05, DP-0.0001	0.9897	0.5814	0.2068	0.8740
PPDL-0.3, DP-0.5	0.9466	0.4250	0.0894	0.5132
PPDL-0.3, DP-0.1	0.9775	0.5672	0.1931	0.8424
PPDL-0.3, DP-0.01	0.9910	0.6783	0.3127	0.8992
PPDL-0.3, DP-0.001	0.9926	0.6672	0.3177	0.8992
PPDL-0.3, DP-0.0001	0.9934	0.6686	0.3223	0.8975
SPN-0.5	0.9936	0.6583	0.2990	0.8711
SPN-0.4	0.9939	0.6594	0.2999	0.8740
SPN-0.3	0.9933	0.6814	0.2999	0.8727
SPN-0.2	0.9937	0.6898	0.3230	0.8788
SPN-0.1	0.9927	0.6897	0.3457	0.8880
SPN-0.01	0.9928	0.6970	0.3843	**0.9163**
SPN-0.001	0.9938	0.6968	0.3768	0.9154
SPN-0.0001	**0.9944**	**0.7154**	0.3740	0.9126

Table 11: Accuracies of different dataset on DLNet with different privacy-preserving mechanisms and their hyper-parameters. Bold values are highest accuracy among different mechanisms in the dataset.

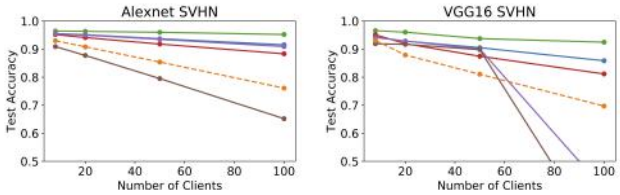
5.9 Accuracies on I.I.D dataset



(a) CIFAR10



(b) CIFAR100



(c) SVHN

Figure 28: Comparison of accuracies for *standalone* local models, *FedAvg* global model and models with different privacy-preserving mechanisms (*Federated SPN*($\alpha_2 = 0.1$), *PPDL-0.3*, *PPDL-0.05* and *DP-0.1*). Improvements over standalone models increase with the number of clients. **Left**: AlexNet; **Right**: VGG16; F-SPN with $\alpha_2 = 0.1$ and DP-0.1 both having $\frac{||B_I||}{||E_B||} \approx 1$ which is considered borderline between green region and white region, F-SPN-0.1 outperforms DP-0.1 in terms of performance (e.g. 25% test accuracy improved in AlexNet CIFAR10.) while maintaining privacy guarantee. While comparing with PPDL with no DP added, F-SPN consistently performs better than PPDL. We observed that PPDL-0.3 and DP-0.1 on VGG16 SVHN are unstable which failed to train at 100 clients.

5.10 Accuracies on Non-I.I.D dataset

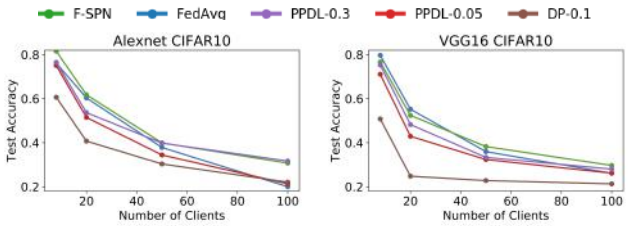


Figure 29: Left: AlexNet; **Right**: VGG16. For non-IID data, we split the dataset using Dirichlet distribution with $alpha = 0.9$ as mentioned in Appendix C. F-SPN is also showing improvements over FedAvg increased with the number of clients and on-par with PPDL-0.3 at 100 clients.

Appendix F: Algorithm

Algorithm 1: Client Update

Input: local model M_i, private target $\boldsymbol{t}_i$, local dataset D_i, global model G

$M_i \leftarrow$ download G from server ;
for *each* $d_j, y_j \in D_i$ **do**
  $\boldsymbol{u}, \boldsymbol{v} =$ forward(M_i, d_j) ;
  $\nabla W_i =$ backprop($(\alpha_1 * L_{CE}(W_i, \boldsymbol{u}, y_j)) + (\alpha_2 * L_P(W_i, \boldsymbol{v}, \boldsymbol{t}_i, y_j))$) ;
  $W_i = W_i - lr * \nabla W_i$;
$\Delta W_i^{new} = W_i - G$;
return ΔW_i^{new}

Algorithm 2: Server Update

Input: client ΔW_i^{new}, global model G

receive ΔW_i^{new} from K clients ;
$G^{new} = G + \frac{1}{K} \sum_{i=1}^{K} \Delta W_i^{new}$;
return G^{new}

Algorithm 3: Training Experiment (FedAvg)

Server: initialize global model G ;
for *each client* **do**
  initialize private target $\boldsymbol{t}_i$;
for *each communication* **do**
  for *each client* **do**
    ΔW_i^{new} = Client Update ; // Client Update
  $G^{new} =$ Server Update ; // Server Update
return G^{new}

Algorithm 4: Training Experiment (Round Robin)

Server: initialize global model G ;
for *each client* **do**
  initialize private target $\boldsymbol{t}_i$;
for *each communication* **do**
  for *each client* **do**
    ΔW_i^{new} = Client Update ; // Client Update
    $G^{new} =$ Server Update ; // Server Update
return G^{new}

Algorithm 5: Reconstruction and Membership Attack (Deep Leakage Attack)

Input: Global model G, dataset D, number of reconstruction X, private target t_i ;
for $1..X$ **do**
  $d_x, y_x \leftarrow$ random sample from D ;
  $\boldsymbol{u}, \boldsymbol{v} =$ forward(G, d_x) ;
  $\nabla W_G = (\alpha_1 * L_{CE}(W_G, \boldsymbol{u}, y_x)) + (\alpha_2 * L_P(W_G, \boldsymbol{v}, \boldsymbol{t}_i, y_x))$;
  $d_x^{rec} =$ INIT(shape of d_x) ;
  $y_x^{rec} =$ INIT(shape of y_x) ;
  while *not converged* **do**
    $\boldsymbol{u} =$ forward(G, d_x^{rec}) ;
    $\nabla W_G^{rec} = L_{CE}(W_G, \boldsymbol{u}, y_x^{rec})$;
    ; // Attacker doesn't know the existence of private term
    minimize $||\nabla W_G - W_G^{rec}||$;
    update d_x^{rec}, y_x^{rec} ;
return D^{rec}, Y^{rec} ; // Reconstructed Images and Labels (Membership)

Algorithm 6: Tracing Attack

Input: Reconstructed images D^{rec}, query dataset D ;
Split D^{rec} and D into N partitions ;
; // Index of reconstructed image is same as index of groundtruth image.
for d_j, *index*$_j \in D$ **do**
    for d_k^{rec}, *index*$_k \in D^{rec}$ **do**
        MSE $= ||d_j - d_k^{rec}||$;
        if *found lowest MSE* **then**
            $t_j =$ index$_k$
        ; // Assign index of reconstructed image into tracing result
    if $t_j ==$ *index*$_j$ **then**
        Tracing successful
return T ; // Tracing Indices