Optimal Defenses Against Data Reconstruction Attacks

Yuxiao Chen [1] **Gamze Gürsoy** [2] **Qi Lei** [3]

Abstract

Federated Learning (FL) is designed to prevent data leakage through collaborative model training without centralized data storage. However, it is vulnerable to reconstruction attacks that recover original training data from shared gradients. To optimize the trade-off between data leakage and utility loss, we first derive a theoretical lower bound of reconstruction error (among all attackers) for the two standard methods: adding noise, and gradient pruning. We then customize these two defenses to be parameter- and model-specific and achieve the optimal trade-off between our obtained reconstruction lower bound and model utility. Experimental results validate that our methods outperform Gradient Noise and Gradient Pruning by protecting the training data better while also achieving better utility. The code for this project is available here.

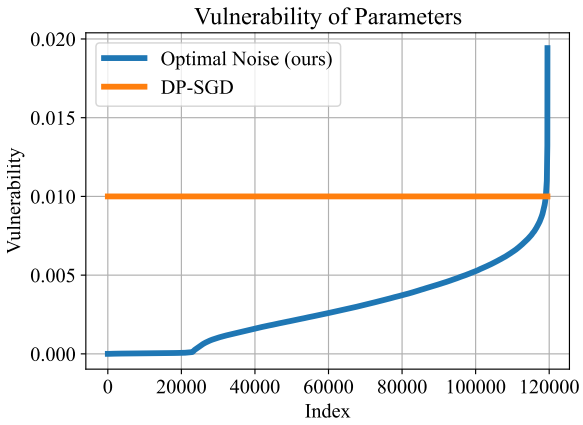

Figure 1. DP-SGD treats all parameters with the same vulnerability, while our method distinguishes the vulnerability of each parameter and designs a customized defense strategy.

1. Introduction

Recent advancements in machine learning have led to remarkable achievements across multiple domains. These successes are largely driven by the ability to gather vast, diverse datasets to train large, powerful models. However, this can be challenging to obtain in certain sectors such as healthcare and finance due to concerns about privacy and institutional restrictions.

Federated or Collaborative Learning (FL) (McMahan et al., 2017) has emerged as a solution to these concerns. Federated learning is a machine learning approach where multiple institutions or devices collaboratively train a model while keeping their data localized. Instead of sharing raw data, each participant shares model updates, aggregated centrally to train a global model that benefits from all participants' insights without compromising data privacy. The assumption is that the shared model weights or intermediate gradients contains less information about the training data.

However, FL is not immune to privacy risks, one type of attack that may harm privacy is the data reconstruction attack via gradient information (Gradient Inversion Attack), where adversaries attempt to reconstruct original training data from the shared gradients. Methods such as DLG (Zhu et al., 2019), CAFE (Jin et al., 2021), and GradInversion (Yin et al., 2021) have shown the feasibility of these attacks. To mitigate these risks, several defense mechanisms have been proposed. The most common approach is perturbing the gradients, such as DP-SGD (Abadi et al., 2016) and Gradient Pruning (Zhu et al., 2019). Although these methods offer some level of data protection, they often encounter a trade-off between maintaining privacy and preserving model performance (Zhang et al., 2023).

In this work, we optimize gradient noise and gradient pruning to an optimal and parameter-specific defense. These methods, serving as basic components of more advanced defenses, could be applied to improve these advanced defenses. Thus, we focus on optimizing these two defenses. A universal defense strategy provides undifferentiated protection and is not optimal for utility-privacy trade-offs. (Shi et al., 2022)

[1]School of Mathematical Sciences, Peking University, Beijing, China [2]Department of Biomedical Informatics, Columbia University, New York, United States of America [3]Courant Institute for Mathematical Sciences and Center for Data Science, New York University, New York, United States of America. Correspondence to: Yuxiao Chen <yuxiaoc@umich.edu>.

Published at Data in Generative Models Workshop: The Bad, the Ugly, and the Greats (DIG-BUGS) at ICML 2025, Vancouver, Canada.

As illustrated in Figure 1, different parameters impact privacy and utility differently. Therefore the optimal defense should treat different parameters differently. Our objective is to optimize the balance between a model's resilience to data reconstruction attacks and its training effectiveness. Our primary contributions are:

- We establish a theoretical lower bound for the expected reconstruction error, which can be easily evaluated.
- We propose two defense mechanisms—Optimal Gradient Noise and Optimal Gradient Pruning—that maximize this bound for a given level of utility.

In Section 2, we provide a brief overview of federated learning and theoretical backgrounds for our method. In Section 3, we present the theoretical foundation for our lower bound and optimal defense methods, and present the implementations of our proposed algorithms. Section 4 evaluates their effectiveness against data reconstruction attacks in image classification tasks.

1.1. Related Works

Federated learning (FL) was introduced by McMahan et al. (2017) as a framework for collaborative model training without centralized data storage. Differential privacy (DP) (Dwork et al., 2006b;a; Dwork & Roth, 2014) has been used to define privacy of the algorithm. Abadi et al. (2016) introduced a differential private SGD algorithm to provide DP guarantees to the trained model. Stock et al. (2022); Guo et al. (2022) provided different types of guarantee of privacy into the model.

However, Zhu et al. (2019) revealed a significant vulnerability in FL by demonstrating how training data can be reconstructed from shared gradients using the DLG algorithm. This attack was refined through subsequent works, such as iDLG (Zhao et al., 2020) and Inverting Gradients (Geiping et al., 2020). More advanced techniques, including GradInversion (Yin et al., 2021) and CAFE (Jin et al., 2021), further enhance reconstruction quality but often rely on additional information or specific model architectures. More works featuring attacks include Wang et al. (2023); Jeon et al. (2021); Chen & Campbell (2021); Li et al. (2022).

In response to these attacks, several defense strategies have been proposed. One line of methods perturb the gradients shared to the server(Sun et al., 2021; Andrew et al., 2021; Sun et al., 2021), while another line of work focus on directly preprocessing the data instead of the gradients(Huang et al., 2020; Zhang et al., 2018; Fan, 2018; Gao et al., 2021). More details about attacks and defenses could be found in Zhang et al. (2022); Bouacida & Mohapatra (2021); Jegorova et al. (2023). To consolidate research in this area, Liu et al. (2024) proposed a framework to systematically analyze the effectiveness of different attacks and defenses. Balunovic et al. (2021) proposed a framework for evaluating defenses. Wen et al. (2022) offered an integrated implementation of attacks and defenses.

In a similar vein, Fay et al. (2023) explored hyperparameter selection to optimize the privacy-utility trade-off in DP-SGD, Xue et al. (2024) proposed DP-SGD with adaptive noise. However, these approaches do not account for the local parameter landscape, which we address in our work.

The lower bound on the reconstruction error was first introduced in previous work (Liu et al., 2024) except that only a local approximation was used.

2. Preliminaries

Notations. Let $\mathcal{P}(A)$ denote the family of distributions over the set A. $\lambda_1(M), ..., \lambda_d(M)$ represent the eigenvalues of a matrix M ranked large to small. If not especially mentioned then $\|\cdot\|$ represents the l_2-norm for vectors and the Frobenius norm for matrices. For a function $f : \mathbb{R}^a \to \mathbb{R}^b$, denote $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$ the Jacobian matrix in $\mathbb{R}^{a\times b}$.

We denote $\boldsymbol{x} \in \mathbb{R}^m$ the training data generated from a distribution $\mathcal{D}$. $L_\Theta : \mathbb{R}^m \to \mathbb{R}$ is the loss function parameterized by $\Theta \in \mathbb{R}^d$. The model gradient $g_\Theta(\boldsymbol{x}) \in \mathbb{R}^d$ is $g_\Theta(\boldsymbol{x}) \coloneqq \nabla_\Theta L_\Theta(\boldsymbol{x})$. When no ambiguity, we write $g(\boldsymbol{x})$ for brevity. $\boldsymbol{y}$ is an (random) observation generated from $\boldsymbol{x}$: $\boldsymbol{y} = S(g(\boldsymbol{x}))$, where $S : \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$ is a random defense mechanism such as adding noise. $R : \mathbb{R}^d \to \mathbb{R}^d$ is an algorithm attempting to reconstruct $\boldsymbol{x}$ from $\boldsymbol{y}$.

2.1. Federated Learning and Data Reconstruction Attacks

Different from traditional centralized optimization where we train a model on curated datasets, federated learning collaboratively train a model while the data remains decentralized and stored locally on clients. This setup intends to protect users' sensitive data without directly sharing them.

In FL, each client $u_i \in \{u_1, \dots, u_n\}$ owns a private dataset D_i, and the global dataset is $D = \bigcup_{i=1}^n D_i$. A central server aims to train a model Θ by solving the optimization problem:

$$\min_{\Theta} \sum_{i=1}^{n} \sum_{\boldsymbol{x}_j \in D_i} L(\boldsymbol{x}_j, \Theta).$$

During training, stochastic gradient descent (SGD) is conducted, where a subset of (well-connected and active) clients $U \subset \{1, \dots, n\}$ will interact with the global server: Each active client $i \in U$ uses a subset $D_i' \subset D_i$ to create a minibatch $B = \bigcup_{i\in U} D_i'$. The global minibatch gradient $\nabla_\Theta L(B, \Theta)$ is computed as a weighted average of the indi-

vidual client gradients:

$$\nabla_\Theta L(B,\Theta) = \frac{1}{|B|}\sum_{i\in U}|D_i'|\nabla_\Theta L(D_i',\Theta^t).$$

Each client shares $\langle |D_i'|, \nabla_\Theta L(D_i',\Theta^t)\rangle$ with the server, which then updates the model parameters as:

$$\Theta^{t+1} \leftarrow \Theta^t - \eta\nabla_\Theta L(B,\Theta).$$

Although these shared gradients contain less information than the raw data, there remains a risk of data leakage, as demonstrated by increasing attention recently (Yin et al., 2021; Huang et al., 2020; Geiping et al., 2020). This work focuses on defending local gradients while minimizing the impact on training utility.

2.2. The Bayesian Cramér-Rao Lower Bound

The data reconstruction problem is essentially the problem of estimation from random observations. Let $\boldsymbol{x} \in \mathbb{R}^d$ represent training data drawn from a distribution $\mathcal{D}$, $\boldsymbol{y} \in \mathbb{R}^K$ denote random observations generated from $\boldsymbol{x}$, and $\hat{\boldsymbol{x}}(\boldsymbol{y})$ be an estimator of $\boldsymbol{x}$. We will introduce the Bayesian Cramer-Rao lower bound that relates to the lowest possible estimation error $\mathbb{E}[\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2]$. First, assume the following regularity conditions hold (Crafts et al., 2024; Van Trees, 1992):

Assumption 1 (Support). The support of $\mathcal{D}$ is either $\mathbb{R}^d$ or an open bounded subset of $\mathbb{R}^d$ with a piecewise smooth boundary.

Assumption 2 (Existence of Derivatives). The derivatives $[\nabla_{\boldsymbol{x}} p(\boldsymbol{x},\boldsymbol{y})]_i$ for $i = 1,\dots,d$, exist and are absolutely integrable.

Assumption 3 (Finite Bias). The bias, defined as

$$B(\boldsymbol{x}) \coloneqq \int (\hat{\boldsymbol{x}}(\boldsymbol{y}) - \boldsymbol{x}) p(\boldsymbol{y} \mid \boldsymbol{x})\, d\boldsymbol{y},$$

is finite for all $\boldsymbol{x}$.

Assumption 4 (Exchanging Derivative and Integral). The probability function $p(\boldsymbol{x},\boldsymbol{y})$ and estimator $\hat{\boldsymbol{x}}(\boldsymbol{y})$ satisfy:

$$\nabla_{\boldsymbol{x}} \int p(\boldsymbol{x},\boldsymbol{y})\left[\hat{\boldsymbol{x}}(\boldsymbol{y}) - \boldsymbol{x}\right]^T d\boldsymbol{y}$$
$$= \int \nabla_{\boldsymbol{x}} \left(p(\boldsymbol{x},\boldsymbol{y})\left[\hat{\boldsymbol{x}}(\boldsymbol{y}) - \boldsymbol{x}\right]^T \right) d\boldsymbol{y}$$

for all $\boldsymbol{x}$.

Assumption 5 (Error Boundary Conditions). For any point $\boldsymbol{x}$ on the boundary of $\text{supp}(\mathcal{D})$, and any sequence $\{\boldsymbol{x}_i\}_{i=0}^{\infty}$ such that $\boldsymbol{x}_i \in \text{supp}(\mathcal{D})$ and $\boldsymbol{x}_i \to \boldsymbol{x}$, we have $B(\boldsymbol{x}_i)p(\boldsymbol{x}_i) \to 0$.

These assumptions are satisfied by a wide range of setups. For image classification, the dataset has bounded support and the defense a differentiable density function $p(\boldsymbol{x},\boldsymbol{y})$. When we add a small Gaussian noise to the training data, all Assumptions 1 to 5 hold. (Crafts et al., 2024)

Given these assumptions, the Bayesian Cramér-Rao Lower Bound is as follows:

$$\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\left[(\hat{\boldsymbol{x}}(\boldsymbol{y}) - \boldsymbol{x})(\hat{\boldsymbol{x}}(\boldsymbol{y}) - \boldsymbol{x})^T\right] \succeq \boldsymbol{V}_B \coloneqq \boldsymbol{J}_B^{-1};$$

where $\boldsymbol{J}_B \in \mathbb{R}^{D\times D}$ is the Bayesian information matrix:

$$\boldsymbol{J}_B \coloneqq \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\left[\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x},\boldsymbol{y}) \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x},\boldsymbol{y})^T\right].$$

The matrix $\boldsymbol{J}_B$ can be decomposed into two components:

$$\boldsymbol{J}_B = \boldsymbol{J}_P + \boldsymbol{J}_D;$$

where $\boldsymbol{J}_P$ is the prior-informed term:

$$\boldsymbol{J}_P \coloneqq \mathbb{E}_{\boldsymbol{x}}\left[\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})^T\right];$$

and $\boldsymbol{J}_D$ is the data-informed term:

$$\boldsymbol{J}_D \coloneqq \mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{J}_F(\boldsymbol{x})\right].$$

Here, $\boldsymbol{J}_F(\boldsymbol{x})$ represents the Fisher information matrix:

$$\boldsymbol{J}_F(\boldsymbol{x}) \coloneqq \mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[\nabla_{\boldsymbol{x}} \log p(\boldsymbol{y} \mid \boldsymbol{x}) \nabla_{\boldsymbol{x}} \log p(\boldsymbol{y} \mid \boldsymbol{x})^T\right].$$

3. Methodology

To optimize the trade-off between privacy and training utility, we treat each parameter separately and design defending strategies customized to the current data batch and model parameters, instead of a universal strategy like a constant noise level in DP-SGD. We will first present our derivation of the reconstruction error lower bound and our definition of the training utility. Then, we introduce an optimization objective to find the optimal defense parameters (such as noise's covariance matrix) that balance reconstruction error and utility.

3.1. The Reconstruction Error Lower Bound

To prevent data leakage, our goal is to maximize the lower bound of the reconstruction error among all reconstruction algorithms. For a randomized defense mechanism $S : \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$ (e.g., adding noise to the gradients), the defended gradient is $\boldsymbol{y} \sim S(g(\boldsymbol{x}))$. For any reconstruction algorithm $R : \mathbb{R}^d \to \mathbb{R}^m$, the expected reconstruction error against the defense is:

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))}\left\|R(\boldsymbol{y}) - \boldsymbol{x}\right\|^2.$$

Definition 3.1. For a data distribution $\mathcal{D} \in \mathcal{P}(\mathbb{R}^m)$, a gradient function $g : \mathbb{R}^m \to \mathbb{R}^d$, and a defense mechanism $S : \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$, the reconstruction error lower bound $B_{\mathcal{D},S}$ is the minimum expected reconstruction error among all reconstruction algorithms $R : \mathbb{R}^d \to \mathbb{R}^m$ following Assumptions 1 to 5:

$$B_{\mathcal{D},S} \coloneqq \min_{R:\mathbb{R}^d\to\mathbb{R}^m} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))}\left\|R(\boldsymbol{y}) - \boldsymbol{x}\right\|^2.$$

We utilize the Bayesian C-R lower bound to lower bound the reconstruction error lower bound:

Theorem 3.2. *Let $B_{\mathcal{D},S}$ be as defined in Definition 3.1. Under Assumptions 1 to 5, we lower bound $B_{\mathcal{D},S}$ by:*

$$B_{\mathcal{D},S} \geq \frac{d^2}{\mathbb{E}_{x\sim\mathcal{D}}\left[\mathrm{tr}(\boldsymbol{J}_F(x))\right] + d\cdot\lambda_1(\boldsymbol{J}_P)}, \tag{1}$$

where $\boldsymbol{J}_F(\boldsymbol{x})$ is given by:

$$\boldsymbol{J}_F(\boldsymbol{x}) \coloneqq \mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))}\left[\nabla_{\boldsymbol{x}}\log p_{S(\boldsymbol{x})}(\boldsymbol{y})\nabla_{\boldsymbol{x}}\log p_{S(\boldsymbol{x})}(\boldsymbol{y})^\top\right]$$

and $\boldsymbol{J}_P$ by:

$$\boldsymbol{J}_P \coloneqq \mathbb{E}_{\boldsymbol{x}}\left[\nabla_{\boldsymbol{x}}\log p_{\mathcal{D}}(\boldsymbol{x})\nabla_{\boldsymbol{x}}\log p_{\mathcal{D}}(\boldsymbol{x})^\top\right].$$

Here, $\boldsymbol{J}_F(\boldsymbol{x})$ depends on the defense method S, while $\boldsymbol{J}_P$ depends only on the distribution $\mathcal{D}$. If the prior is flat, $\lambda_1(\boldsymbol{J}_P)\approx 0$.

Remark 3.3. The lower bound decreases with $\mathrm{tr}(\boldsymbol{J}_F(x))$. Thus, to improve our reconstruction error lower bound, we minimize $\mathrm{tr}(\boldsymbol{J}_F(x))$ for our defense method.

3.2. Training Utility

To assess utility, we analyze the model loss after one step of gradient descent update. Due to the complexity of the loss landscape, we make an approximation by the first-order Taylor expansion. The expected loss using the second-order Taylor approximation may seem more accurate, but could lead to a case where larger noise increases utility, leading to an unrealistic result of infinitely large optimal noise. Fay et al. (2023) analyzed the utility of DP-SGD by using the lower bound of the expected loss, derived by assuming the loss function M-smooth. However, this oversimplifies the loss landscape by using the same isotropic convex function regardless of training data or model parameters. Optimizing this bound also requires choosing the optimal learning rate, while we aim to separate the defense method from the learning rate to make our defense more general.

To avoid these issues, we use the expectation and variance of the model loss after one gradient update, approximated by the first-order Taylor expansion, as our utility measure. These measures are both independent of the learning rate; and also contain information about the loss function's landscape. A good defense method should minimally impact training utility, therefore we maximize the expectation of the training loss and minimize its variance.

Definition 3.4. Given training data $\boldsymbol{x}\in\mathbb{R}^m$ from distribution $\mathcal{D}$, a model with d parameters Θ, and a loss function $L:\mathbb{R}^m\times\mathbb{R}^d\to\mathbb{R}$, the first-order utility of a defense method S on $\boldsymbol{x}$ is the expected decrease in the training loss on $\boldsymbol{x}$ after one gradient update:

$$U_1(S,\Theta) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathbb{E}_{\boldsymbol{y}\sim S(\nabla_{\boldsymbol{x}}L(\boldsymbol{x},\Theta))}\nabla_{\boldsymbol{x}}L(\boldsymbol{x},\Theta)\cdot\boldsymbol{y}. \tag{2}$$

The second-order utility is defined as the negative variance of the training loss on $\boldsymbol{x}$ after the update:

$$U_2(S,\Theta) = -\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\mathrm{Var}_{\boldsymbol{y}\sim S(\nabla_{\boldsymbol{x}}L(\boldsymbol{x},\Theta))}\nabla_{\boldsymbol{x}}L(\boldsymbol{x},\Theta)\cdot\boldsymbol{y}. \tag{3}$$

3.3. Optimal Gradient Noise

Gradient Noise. One of the simplest defense methods, also a step in DP-SGD (Abadi et al., 2016), is to add Gaussian noise to the model gradients before sharing. For a given covariance matrix $\boldsymbol{\Sigma}\in\mathbb{R}^{d\times d}$, the gradient noise defense is as follows:

$$S_{\text{noise},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x},\boldsymbol{\Sigma}). \tag{4}$$

Optimal Gradient Noise. The first-order utility defined in Eq. 2 remains constant regardless of the choice of the covariance matrix $\boldsymbol{\Sigma}$:

$$U_1(S_{\text{noise},\boldsymbol{\Sigma}},\Theta) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\nabla_{\boldsymbol{x}}L(\boldsymbol{x},\Theta)\cdot\nabla_{\boldsymbol{x}}L(\boldsymbol{x},\Theta)^\top.$$

Thus, we focus on maximizing the second-order utility. Assuming independent noise across parameters (as in DP-SGD), we limit our analysis to diagonal matrices. Since the noise added to each parameter are independent, the second-order utility equals:

$$U_2(S_{\text{noise},\boldsymbol{\Sigma}},\Theta) = -\sum_{i=1}^{d}\left(\frac{\partial L(\boldsymbol{x},\Theta)}{\partial\boldsymbol{x}_i}\right)^2\Sigma_{i,i}. \tag{5}$$

For a higher reconstruction error lower bound, we minimize $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\,\mathrm{tr}(\boldsymbol{J}_F(\boldsymbol{x}))$, where:

$$\mathrm{tr}(\boldsymbol{J}_F(\boldsymbol{x})) = \sum_{i=1}^{d}\frac{\|\nabla_{\boldsymbol{x}}g(\boldsymbol{x})\|^2}{\Sigma_{i,i}}. \tag{6}$$

This decomposition allows us to separate the influence the defense of each parameter has on utility and privacy, setting the stage for deriving the optimal noise.

Theorem 3.5 (Optimal Gradient Noise)**.** *Under assumptions 1 to 5, and assuming $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}g_i(\boldsymbol{x})^2 > 0$ for all i,* [1] *the optimal noise matrix $\boldsymbol{\Sigma}$ for a given utility budget $U_2(S_{noise,\boldsymbol{\Sigma}},\Theta)\geq -C$ has diagonal elements:*

$$\Sigma_{i,i} = \lambda\sqrt{\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\|^2}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}g_i(\boldsymbol{x})^2}},$$

where λ is a constant, and g_i is the i-th component of the gradient $g(\boldsymbol{x})=\nabla_{\boldsymbol{x}}L_\Theta(\boldsymbol{x})$.

[1] Special cases where certain entries of $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}g_i(\boldsymbol{x})^2$ are zero (e.g., in models with ReLU activation) are discussed in the appendix.

In the special case where $\mathcal{D}$ is supported on a small neighborhood of $\boldsymbol{x}$, corresponding to the attacker having an approximation of the data, we could approximate and simplify the locally optimal noise by using the value at $\boldsymbol{x}$ to replace the expectations:

$$\Sigma_{i,i}(\boldsymbol{x}) = \lambda(\boldsymbol{x}) \frac{\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|}{|g_i(\boldsymbol{x})|}. \tag{7}$$

3.4. Optimal Gradient Pruning

Gradient Pruning. Gradient pruning reduces the number of parameters in the shared gradient by zeroing out less significant gradients during training. Inspired by gradient compression (Lin et al., 2018; Tsuzuku et al., 2018), this approach prunes gradients with the smallest magnitude (Zhu et al., 2019). It is also the most effective defense against DLG (Zhu et al., 2019).

For a given set of parameters $\mathbb{A}$, the gradient pruning defense method $S_{\text{prune},\mathbb{A}} : \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$ is as follows:

$$S_{\text{prune},\mathbb{A}}(\boldsymbol{x})_i = \begin{cases} 0, & \text{if } i \in \mathbb{A}, \\ \boldsymbol{x}_i, & \text{if } i \notin \mathbb{A}. \end{cases} \tag{8}$$

Optimal Gradient Pruning Under assumptions 1 to 5, the first-order utility of gradient pruning equals the sum of the squared unpruned gradients:

$$U_1(S, \Theta) = \sum_{i \notin \mathbb{A}} \left(\frac{\partial L(\boldsymbol{x}, \Theta)}{\partial \boldsymbol{x}_i} \right)^2. \tag{9}$$

Since gradient pruning introduces no randomness, an accurate reconstruction is theoretically possible when the number of unpruned parameters exceeds the input dimension. To address this problem, we add a small noise to the unpruned gradients and analyze the noisy version of gradient pruning:

$$S_{\text{prune},\mathbb{A},\boldsymbol{\Sigma}}(\boldsymbol{x})_i = \begin{cases} 0, & \text{if } i \in \mathbb{A}, \\ \mathcal{N}(\boldsymbol{x}_i, \boldsymbol{\Sigma}_{i,i}), & \text{if } i \notin \mathbb{A}. \end{cases} \tag{10}$$

For $\boldsymbol{\Sigma} = \epsilon \boldsymbol{\Sigma}_0$, this collapses to the original pruning method when $\boldsymbol{\Sigma}_0$ remains constant and $\epsilon \to 0$.

Theorem 3.6 (Optimal Gradient Pruning)**.** *Under assumptions 1 to 5, and assuming $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2 > 0$ for all i, the optimal pruning distribution $\mathcal{R}$ for generating $\mathbb{A}$, given the utility budget*

$$\mathbb{E}_{\mathbb{A}\sim\mathcal{R}} U_1(S_{\textit{prune},\mathbb{A},\boldsymbol{\Sigma}}, \Theta) \geq C,$$

the optimal pruning set prunes elements with the largest value of:

$$k_i = \frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2},$$

where $g(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} L_\Theta(\boldsymbol{x})$ is the model gradient, and g_i represents its i-th component.

Remark 3.7. When a deterministic set does not match the utility budget, parameters on the borderline are pruned with positive probability. When this happens, the optimal defense is a mixed defense.

Similar to previous sections, we derive locally optimal gradient pruning, which prunes parameters i with the largest index k_i':

$$k_i' = \frac{\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|}{|g_i(\boldsymbol{x})|}. \tag{11}$$

An additional feature of the locally optimal version is that it is also the optimal defense when using optimal noise instead of identical noise in Theorem 3.5 in our analysis.

3.5. Algorithm design

Because of the high computational cost of the expectation terms in the globally optimal defense methods (Theorems 3.5 and 3.6), our implementations are based on the locally optimal versions (Eqs. 7 and 11).

When computing optimal defense parameters, calculating the Jacobian matrix of model gradients on input data is especially challenging. The full Jacobian matrix for an image with a resolution of 32×32 would require roughly 3000 times the memory of the model itself, which is prohibitively large. We resolve this problem by using the forward differentiation method to save computational cost and use approximation to save memory cost.

The forward method (Griewank & Walther, 2008) tracks gradients based on the input tensor size rather than the output tensor size, and therefore more efficient since we are dealing with low input and high output dimensions. We approximate the l_2-norm of the gradients using Lemma 3.8:

Lemma 3.8. *Given a differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ and a constant $\epsilon > 0$. For any number of samples $k \in \mathbb{Z}$ and random vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ sampled from $\mathcal{N}(0, \boldsymbol{I}_n)$, we have that*

$$\left| \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2 - \frac{1}{k} \sum_{j=1}^{k} \left\| \frac{\partial f_i(\boldsymbol{x} + \alpha \boldsymbol{x}_j)}{\partial \alpha} \right\|_{\alpha=0}^2 \right| \leq \epsilon \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2 \tag{12}$$

holds with probability at least $1 - \frac{2}{k\epsilon^2}$ for any $\boldsymbol{x} \in \mathbb{R}^d$.

This allows us to approximate the l_2-norms without the entire Jacobian matrix, significantly reducing computational and memory cost. Our resulting algorithm is outlined in Algorithm 1.

4. Experiments

We compare our proposed algorithms with existing defense methods on the CIFAR-10 dataset (Krizhevsky & Hinton, 2009). As our algorithm employs different defenses on

Algorithm 1 Approximate Locally Optimal Defense

Input: Model parameters $\Theta \in \mathbb{R}^d$, loss function $L(\boldsymbol{x}, \Theta)$, number of samples k, prune threshold η (pruning), noise scale λ (noise), small constant c(noise)
Output: Defended model gradients $S(g(\boldsymbol{x}))$
Step 1: Compute the model gradients $g(\boldsymbol{x}) = \nabla_\Theta L(\boldsymbol{x}, \Theta)$
Step 2: Sample k random vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \sim \mathcal{N}(0, \boldsymbol{I}_n)$
Step 3: Approximate $\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2$ with $\frac{1}{k}\sum_{j=1}^{k} \left\| \frac{\partial g_i(\boldsymbol{x}+\alpha \boldsymbol{x}_j)}{\partial \alpha} \right\|^2_{\alpha=0}$.
Step 4a (Gradient Noise): Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \Sigma)$ where

$$\Sigma_{i,i} = \lambda \frac{\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2}{\max(|g_i(\boldsymbol{x})|, c)}$$

and return the defended gradients $g^{\text{noise}}(\boldsymbol{x}) = g(\boldsymbol{x}) + \boldsymbol{\epsilon}$
Step 4b (Gradient Pruning): Return pruned gradients $g^{\text{prune}}(\boldsymbol{x})$ with elements

$$g_i^{\text{prune}}(\boldsymbol{x}) = \mathbf{1}_{\frac{\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|}{|g_i(\boldsymbol{x})|} \leq \eta} \cdot g_i(\boldsymbol{x}). \tag{13}$$

different parameters, we use an attack that treats parameters equally. One attack with such property is the Inverting Gradients attack (Geiping et al., 2020), a powerful attack that does not require extra information or specific model architecture. We defer additional experiments on MNIST to Appendix D.

4.1. Optimal Gradient Pruning

As shown in Figure 2, our approximately optimal strategy for gradient pruning (optimal pruning in short) achieved higher reconstruction error than gradient pruning for the same level of training utility, with a pruning ratio of 70% outperforming 90% pruning in gradient pruning. Visual

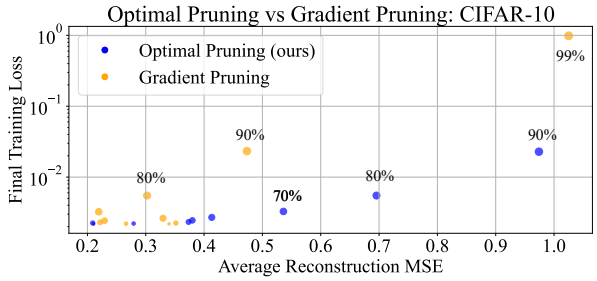

Figure 2. Comparison of optimal pruning and gradient pruning on CIFAR-10. X-axis: average MSE. Y axis: Training loss on 8 samples.

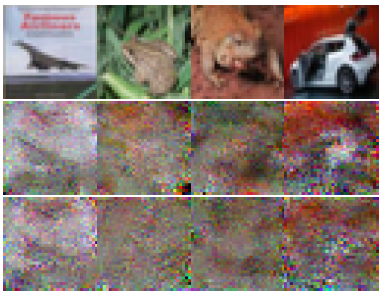

Figure 3. Reconstruction from the CIFAR-10 dataset with batch size 4. First row: ground truth. Second row: 90% gradient pruning. Third row: 80% optimal pruning. 80% optimal pruning has better training utility and better protection against reconstruction.

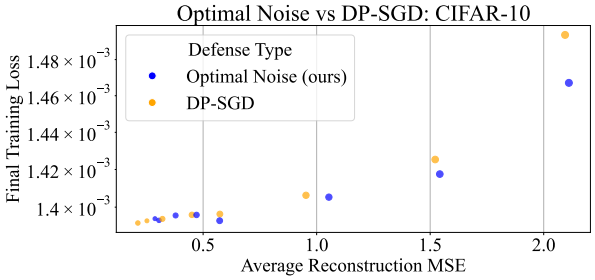

Figure 4. Comparison of optimal noise and DP-SGD on CIFAR-10. X-axis: average MSE. Y axis: Training loss on 8 samples.

comparison (Figure 3) also indicates better protection using our method.

4.2. Optimal Gradient Noise

As shown in Figure 4, our proposed defense method for adding noise still offers a better privacy-utility trade-off.

5. Discussion

In this work, we derived a theoretical reconstruction lower bound and used it to formulate optimal defense methods as improvements of gradient noise and gradient pruning. Since our work only shows the theoretical possibility of a higher privacy-utility tradeoff, a key limitation of our methods is the high computational cost of our algorithm. This could be mitigated by simplifications (e.g. layer-wise defense) or lowering the frequency of updating defense parameters. Additionally, the reconstruction bound used in our analysis is not tight. The utilization of more precise bounds or privacy measures that integrate current attack methods remains an open challenge. Furthermore, just as how we applied optimal noise to replace the noise in DP-SGD, our analysis could potentially be incorporated into other defense methods. This also remains an open challenge for further research.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 308–318, 2016.

Andrew, G., Thakkar, O., McMahan, B., and Ramaswamy, S. Differentially private learning with adaptive clipping. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17455–17466. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/91cff01af640a24e7f9f7a5ab407889f-Paper.pdf.

Balunovic, M., Dimitrov, D. I., Staab, R., and Vechev, M. T. Bayesian framework for gradient leakage. *CoRR*, abs/2111.04706, 2021. URL https://arxiv.org/abs/2111.04706.

Bouacida, N. and Mohapatra, P. Vulnerabilities in federated learning. *IEEE Access*, 9:63229–63249, 2021. doi: 10.1109/ACCESS.2021.3075203.

Chen, C. and Campbell, N. D. F. Understanding training-data leakage from gradients in neural networks for image classification, 2021. URL https://arxiv.org/abs/2111.10178.

Crafts, E. S., Zhang, X., and Zhao, B. Bayesian cramér-rao bound estimation with score-based models. *IEEE Transactions on Information Theory*, pp. 1–1, 2024. ISSN 1557-9654. doi: 10.1109/tit.2024.3447552. URL http://dx.doi.org/10.1109/TIT.2024.3447552.

Dwork, C. and Roth, A. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, August 2014. ISSN 1551-305X. doi: 10.1561/0400000042. URL https://doi.org/10.1561/0400000042.

Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., and Naor, M. Our data, ourselves: privacy via distributed noise generation. In *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques*, EUROCRYPT'06, pp. 486–503, Berlin, Heidelberg, 2006a. Springer-Verlag. ISBN 3540345469. doi: 10.1007/11761679_29. URL https://doi.org/10.1007/11761679_29.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pp. 265–284, Berlin, Heidelberg, 2006b. Springer-Verlag. ISBN 3540327312. doi: 10.1007/11681878_14. URL https://doi.org/10.1007/11681878_14.

Fan, L. Image pixelization with differential privacy. In Kerschbaum, F. and Paraboschi, S. (eds.), *Data and Applications Security and Privacy XXXII - 32nd Annual IFIP WG 11.3 Conference, DBSec 2018, Bergamo, Italy, July 16-18, 2018, Proceedings*, volume 10980 of *Lecture Notes in Computer Science*, pp. 148–162. Springer, 2018. doi: 10.1007/978-3-319-95729-6_10. URL https://doi.org/10.1007/978-3-319-95729-6_10.

Fay, D., Magnússon, S., Sjölund, J., and Johansson, M. Adaptive hyperparameter selection for differentially private gradient descent. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=LLKI5Lq2YN.

Fukushima, K. Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, 5(4):322–333, 1969. doi: 10.1109/TSSC.1969.300225.

Gao, W., Guo, S., Zhang, T., Qiu, H., Wen, Y., and Liu, Y. Privacy-preserving Collaborative Learning with Automatic Transformation Search . In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 114–123, Los Alamitos, CA, USA, June 2021. IEEE Computer Society. doi: 10.1109/CVPR46437.2021.00018. URL https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00018.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. Inverting gradients - how easy is it to break privacy in federated learning? In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 16937–16947. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/c4ede56bbd98819ae6112b20ac6bf145-Paper.pdf.

Griewank, A. and Walther, A. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Society for Industrial and Applied Mathematics, USA, second edition, 2008. ISBN 0898716594.

Guo, C., Karrer, B., Chaudhuri, K., and van der Maaten, L. Bounding training data reconstruction in private (deep) learning. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine*

Learning Research, pp. 8056–8071. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/guo22c.html.

Huang, Y., Song, Z., Li, K., and Arora, S. InstaHide: Instance-hiding schemes for private distributed learning. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4507–4518. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/huang20i.html.

Jegorova, M., Kaul, C., Mayor, C., O'Neil, A. Q., Weir, A., Murray-Smith, R., and Tsaftaris, S. A. Survey: Leakage and Privacy at Inference Time . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 45(07):9090–9108, July 2023. ISSN 1939-3539. doi: 10.1109/TPAMI.2022.3229593. URL https://doi.ieeecomputersociety.org/10.1109/TPAMI.2022.3229593.

Jeon, J., Kim, j., Lee, K., Oh, S., and Ok, J. Gradient inversion with generative image prior. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 29898–29908. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/fa84632d742f2729dc32ce8cb5d49733-Paper.pdf.

Jin, X., Chen, P.-Y., Hsu, C.-Y., Yu, C.-M., and Chen, T. Cafe: Catastrophic data leakage in vertical federated learning. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 994–1006. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/08040837089cdf46631a10aca5258e16-Paper.pdf.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Li, Z., Zhang, J., Liu, L., and Liu, J. Auditing Privacy Defenses in Federated Learning via Generative Gradient Leakage . In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10122–10132, Los Alamitos, CA, USA, June 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.00989. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.00989.

Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, B. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SkhQHMW0W.

Liu, S., Wang, Z., Chen, Y., and Lei, Q. Data reconstruction attacks and defenses: A systematic evaluation, 2024. URL https://arxiv.org/abs/2402.09478.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Singh, A. and Zhu, J. (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 20–22 Apr 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Shi, W., Cui, A., Li, E., Jia, R., and Yu, Z. Selective differential privacy for language modeling. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2848–2859, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.205. URL https://aclanthology.org/2022.naacl-main.205.

Stock, P., Shilov, I., Mironov, I., and Sablayrolles, A. Defending against reconstruction attacks with rényi differential privacy, 2022. URL https://arxiv.org/abs/2202.07623.

Sun, J., Li, A., Wang, B., Yang, H., Li, H., and Chen, Y. Soteria: Provable Defense against Privacy Leakage in Federated Learning from Representation Perspective . In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9307–9315, Los Alamitos, CA, USA, June 2021. IEEE Computer Society. doi: 10.1109/CVPR46437.2021.00919. URL https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00919.

Tsuzuku, Y., Imachi, H., and Akiba, T. Variance-based gradient compression for efficient distributed deep learning, 2018. URL https://openreview.net/forum?id=rkEfPeZRb.

Van Trees, H. L. *Detection, Estimation, and Modulation Theory: Radar-Sonar Signal Processing and Gaussian Signals in Noise*. Krieger Publishing Co., Inc., Melbourne, FL, USA, 1992. ISBN 0894647482. URL http://portal.acm.org/citation.cfm?id=530789.

Wang, Z., Lee, J., and Lei, Q. Reconstructing training data from model gradient, provably. In Ruiz, F., Dy, J., and van de Meent, J.-W. (eds.), *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 6595–6612. PMLR, 25–27 Apr 2023. URL https://proceedings.mlr.press/v206/wang23g.html.

Wen, Y., Geiping, J., Fowl, L. H., Goldblum, M., and Goldstein, T. Fishing for user data in large-batch federated learning via gradient magnification. *ArXiv*, abs/2202.00580, 2022. URL https://api.semanticscholar.org/CorpusID:246441809.

Xue, R., Xue, K., Zhu, B., Luo, X., Zhang, T., Sun, Q., and Lu, J. Differentially private federated learning with an adaptive noise mechanism. *IEEE Transactions on Information Forensics and Security*, 19:74–87, 2024. doi: 10.1109/TIFS.2023.3318944.

Yin, H., Mallya, A., Vahdat, A., Alvarez, J. M., Kautz, J., and Molchanov, P. See through gradients: Image batch recovery via gradinversion. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16332–16341, 2021. doi: 10.1109/CVPR46437.2021.01607.

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

Zhang, R., Guo, S., Wang, J., Xie, X., and Tao, D. A survey on gradient inversion: Attacks, defenses and future directions. In Raedt, L. D. (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 5678–5685. International Joint Conferences on Artificial Intelligence Organization, 7 2022. doi: 10.24963/ijcai.2022/791. URL https://doi.org/10.24963/ijcai.2022/791. Survey Track.

Zhang, X., Kang, Y., Chen, K., Fan, L., and Yang, Q. Y. Q. Trading off privacy, utility and efficiency in federated learning. *ACM Transactions on Intelligent Systems and Technology*, 14(6):1–32, nov 2023. doi: 10.1145/3595185. URL https://doi.org/10.1145/3595185.

Zhao, B., Mopuri, K. R., and Bilen, H. idlg: Improved deep leakage from gradients, 2020. URL https://arxiv.org/abs/2001.02610.

Zhu, L., Liu, Z., and Han, S. Deep leakage from gradients. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf.

A. MISSING PROOFS

A.1. Proof of Theorem 3.2

Proof. Recall the definition

$$B_{\mathcal{D},S} = \min_{R:\mathbb{R}^d \to \mathbb{R}^m} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))} \left\| R(\boldsymbol{y}) - \boldsymbol{x} \right\|^2.$$

By the Bayesian Cramér-Rao lower bound, for any reconstruction algorithm R we have that:

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))} \left[\left(R(\boldsymbol{y}) - \boldsymbol{x} \right) \left(R(\boldsymbol{y}) - \boldsymbol{x} \right)^\top \right] \succeq \boldsymbol{J}_B^{-1},$$

where $\boldsymbol{J}_B = \boldsymbol{J}_P + \boldsymbol{J}_D$ and $\boldsymbol{J}_D \coloneqq \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left[\boldsymbol{J}_F(\boldsymbol{x}) \right]$.

Therefore:

$$\begin{aligned} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))} \left\| R(\boldsymbol{y}) - \boldsymbol{x} \right\|^2 &= \operatorname{tr} \left(\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \mathbb{E}_{\boldsymbol{y}\sim S(g(\boldsymbol{x}))} \left[\left(R(\boldsymbol{y}) - \boldsymbol{x} \right) \left(R(\boldsymbol{y}) - \boldsymbol{x} \right)^\top \right] \right) \\ &\geq \operatorname{tr} \left(\boldsymbol{J}_B^{-1} \right). \end{aligned}$$

Since both $\boldsymbol{J}_D$ and $\boldsymbol{J}_P$ are Fisher information matrices and hence symmetric, we could apply Weyl's inequality to bound the eigenvalues of $\boldsymbol{J}_B$. Let λ_i denote sorted eigenvalues with λ_1 being the smallest and λ_d the largest. For the eigenvalues λ_i of $\boldsymbol{J}_B$, we have:

$$\lambda_i(\boldsymbol{J}_B) \leq \lambda_i(\boldsymbol{J}_D) + \lambda_1(\boldsymbol{J}_P).$$

This implies that

$$\begin{aligned} \operatorname{tr}(\boldsymbol{J}_B^{-1}) &= \sum_{i=1}^{d} \frac{1}{\lambda_i(\boldsymbol{J}_B)} \\ &\geq \sum_{i=1}^{d} \frac{1}{\lambda_i(\boldsymbol{J}_D) + \lambda_1(\boldsymbol{J}_P)} \\ &\geq \frac{d^2}{\operatorname{tr}(\boldsymbol{J}_D) + d \cdot \lambda_1(\boldsymbol{J}_P)}. \end{aligned}$$

The last equation is from Cauchy's inequality since $\lambda_i(\boldsymbol{J}_D) + \lambda_1(\boldsymbol{J}_P) > 0$.

Substituting $\operatorname{tr}(\boldsymbol{J}_D) = \mathbb{E}_{x\sim\mathcal{D}} \left[\operatorname{tr}(\boldsymbol{J}_F(\boldsymbol{x})) \right]$, we obtain:

$$\operatorname{tr}(\boldsymbol{J}_B^{-1}) \geq \frac{d^2}{\mathbb{E}_{x\sim\mathcal{D}} \left[\operatorname{tr}(\boldsymbol{J}_F(\boldsymbol{x})) \right] + d \cdot \lambda_1(\boldsymbol{J}_P)}.$$

Thus, we have shown that

$$B_{\mathcal{D},S} \geq \frac{d^2}{\mathbb{E}_{x\sim\mathcal{D}} \left[\operatorname{tr}(\boldsymbol{J}_F(\boldsymbol{x})) \right] + d \cdot \lambda_1(\boldsymbol{J}_P)}.$$

□

A.2. Proof of Theorem 3.5

To prove the theorem, we first need to calculate $\boldsymbol{J}_F(\boldsymbol{x})$:

Lemma A.1. *Let $\boldsymbol{J}_F(\boldsymbol{x})$ be the Fisher information matrix defined in Theorem 3.2. Let $\boldsymbol{y} = S(\boldsymbol{x})$ be gradients defended with gradient noise using covariance matrix $\boldsymbol{\Sigma}$. We have that:*

$$\boldsymbol{J}_F(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} g(\boldsymbol{x}) \boldsymbol{\Sigma}^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x})^\top.$$

Proof. Recall that $g : \mathbb{R}^m \to \mathbb{R}^d$ is the function from input data to model gradients. The $\nabla_{\boldsymbol{x}} g(\boldsymbol{x})$ is a $m * d$ matrix, the noise matrix $\boldsymbol{\Sigma}$ is a $d * d$ matrix.

Given $\boldsymbol{y} = g(\boldsymbol{x}) + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \Sigma)$, we have the log-likelihood function:

$$\log p(\boldsymbol{y}|\boldsymbol{x}) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log|\Sigma| - \frac{1}{2}(\boldsymbol{y} - g(\boldsymbol{x}))^\top \Sigma^{-1}(\boldsymbol{y} - g(\boldsymbol{x})).$$

The gradient of the log-likelihood with respect to $\boldsymbol{x}$ is:

$$\nabla_{\boldsymbol{x}} \log p(\boldsymbol{y}|\boldsymbol{x})^\top = (\boldsymbol{y} - g(\boldsymbol{x}))^\top \Sigma^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x})^\top.$$

By definition:

$$\begin{aligned}\boldsymbol{J}_F(\boldsymbol{x}) &= \mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[\nabla_{\boldsymbol{x}} \log p(\boldsymbol{y}|\boldsymbol{x}) \nabla_{\boldsymbol{x}} \log p(\boldsymbol{y}|\boldsymbol{x})^\top\right] \\ &= \nabla_{\boldsymbol{x}} g(\boldsymbol{x}) \mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[\Sigma^{-1}(\boldsymbol{y} - g(\boldsymbol{x}))(\boldsymbol{y} - g(\boldsymbol{x}))^\top \Sigma^{-1}\right] \nabla_{\boldsymbol{x}} g(\boldsymbol{x})^\top.\end{aligned}$$

Since $\mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}\left[(\boldsymbol{y} - g(\boldsymbol{x}))(\boldsymbol{y} - g(\boldsymbol{x}))^\top\right] = \Sigma$,

$$\boldsymbol{J}_F(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} g(\boldsymbol{x}) \Sigma^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x})^\top.$$

□

Now we are ready to prove Theorem 3.5:

Proof. We want to minimize $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \operatorname{tr}(\boldsymbol{J}_F(\boldsymbol{x}))$. By the lemma above, we have that

$$\boldsymbol{J}_F(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} g(\boldsymbol{x}) \Sigma^{-1} \nabla_{\boldsymbol{x}} g(\boldsymbol{x})^\top = \sum_{i=1}^{d} \frac{\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2}{\boldsymbol{\Sigma}_{i,i}}$$

when $\boldsymbol{\Sigma}$ is diagonal.

The second-order utility for the defense equals:

$$U_2(S, \Theta) = -\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \sum_{i=1}^{d} \left(\frac{\partial L(\Theta, \boldsymbol{x})}{\partial x_i}\right)^2 \Sigma_{i,i} = -\sum_{i=1}^{d} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2 \Sigma_{i,i},$$

where the second equation is from the definition of $g_i(\boldsymbol{x})$. By Cauchy's inequality, we have that

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \operatorname{tr}(\boldsymbol{J}_F(\boldsymbol{x})) &= \sum_{i=1}^{d} \frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2}{\boldsymbol{\Sigma}_{i,i}} \\ &\geq \frac{\left(\sum_{i=1}^{d} \sqrt{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2 \cdot \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2}\right)^2}{\sum_{i=1}^{d} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2 \Sigma_{i,i}} \\ &\geq \frac{1}{C}\left(\sum_{i=1}^{d} \sqrt{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2 \cdot \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2}\right)^2.\end{aligned}$$

The first inequality holds when and only when

$$\Sigma_{i,i} \propto \sqrt{\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2}},$$

and the second inequality also holds when taking

$$\lambda = \frac{C}{\sum_{i=1}^{d} \sqrt{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[||\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})||^2] \cdot \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[g_i(\boldsymbol{x})^2]}}$$

and

$$\Sigma_{i,i} = \lambda \sqrt{\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\right\|^2}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2}}.$$

This yields $U_2(S, \Theta) = -C$.

Therefore $\mathbb{E}_{x\sim D} \operatorname{tr}\left(\boldsymbol{J}_F(\boldsymbol{x})\right)$ is minimized when any only when

$$\Sigma_{i,i} = \lambda \sqrt{\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\right\|^2}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2}},$$

for some λ. The reconstruction error lower bound is maximized when the above applies. □

A.3. Proof of Theorem 3.6

For mixed defense methods, we derive a bound similar to Eq. 1. Let S be defined as $Q(g(\boldsymbol{x}), i)$, where i is an identifier sampled from distribution $\mathcal{I}$. For each i, $Q(\cdot, i)$ represents a unique defense mechanism, satisfying Assumptions 1 to 5 independently.

Using Jensen's inequality, we further obtain the lower bound for mixed defense:

$$\begin{aligned} B_{\mathcal{D},\mathcal{I}} =& \mathbb{E}_{i\sim\mathcal{I}} B_{\mathcal{D},Q(\cdot,i)} \\ \geq& \frac{d^2}{\mathbb{E}_{i\sim\mathcal{I}}\mathbb{E}_{x\sim\mathcal{D}}\left[\operatorname{tr}(\boldsymbol{J}_{F,Q(\cdot,i)}(x))\right] + d\cdot\lambda_1(\boldsymbol{J}_P)}, \end{aligned} \tag{14}$$

where $\boldsymbol{J}_{F,Q(\cdot,i)}(\boldsymbol{x})$ represents

$$\mathbb{E}_{\boldsymbol{y}\sim Q(\boldsymbol{x},i)}\left[\nabla_{\boldsymbol{x}} \log p_{S(\boldsymbol{x})}(\boldsymbol{y}) \nabla_{\boldsymbol{x}} \log p_{S(\boldsymbol{x})}(\boldsymbol{y})^\top\right].$$

For generalized gradient pruning, we need to use mixed defense. Similar as Theorem 3.5, we first calculate $\operatorname{tr}(\boldsymbol{J}_{F,S_0(\cdot,r)}(x))$ in the mixed defense version of Theorem 3.2. For the noisy gradient pruning defense, the identifier r is $\mathbb{A}$, $S_0(\cdot, \mathbb{A})$ is noisy gradient pruning that prunes parameters in the set $\mathbb{A}$. $\mathcal{R}$ is the distribution generating the pruning set. We find the optimal distribution $\mathcal{R}$.

Lemma A.2. *Let $\operatorname{tr}(\boldsymbol{J}_{F,S_{prune},\boldsymbol{\Sigma},\mathbb{A}}(x))$ be the trace of the Fisher information matrix $\boldsymbol{J}_F(x)$ defined in Theorem 3.2. Let $\boldsymbol{y} = S(\boldsymbol{x})$ be the model gradients defended by noisy gradient pruning with covariance matrix $\boldsymbol{\Sigma} = \epsilon \boldsymbol{I}_d$ and pruning set $\mathbb{A} \sim \mathcal{R}$. Then:*

$$\operatorname{tr}(\boldsymbol{J}_{F,S_{prune},\boldsymbol{\Sigma},\mathbb{A}}(x)) = \frac{1}{\epsilon}\sum_{i\notin\mathbb{A}} \left\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\right\|^2.$$

Proof. Denote $\{1:d\}$ the set of integers 1 to d. Denote $A - B$ the set of elements included only in A and not in B. By the lemma used in the proof of Theorem 3.5, the left-hand side equals

$$\begin{aligned} \operatorname{tr}(\boldsymbol{J}_{F,S_{\text{prune}},\boldsymbol{\Sigma},\mathbb{A}}(x)) =& \operatorname{tr}(\frac{1}{\epsilon}\nabla_{\boldsymbol{x}_{\{1:d\}-\mathbb{A}}} g(\boldsymbol{x}) \nabla^\top_{\boldsymbol{x}_{\{1:d\}-\mathbb{A}}} g(\boldsymbol{x})) \\ =& \frac{1}{\epsilon}\left\|\nabla_{\boldsymbol{x}_{\{1:d\}-\mathbb{A}}} g(\boldsymbol{x})\right\|_F^2 \\ =& \frac{1}{\epsilon}\sum_{i\notin\mathbb{A}} \left\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\right\|^2. \end{aligned}$$

The last equation is true since for each i, $\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})$ corresponds to a column in $\nabla_{\boldsymbol{x}_{\{1:d\}-\mathbb{A}}} g(\boldsymbol{x})$. □

Now we can prove Theorem 3.6:

Proof. For the training utility, we have:

$$\begin{aligned} U_1(S,\Theta) =& \mathbb{E}_{\mathbb{A}\sim\mathcal{R}}\mathbb{E}_{x\sim\mathcal{D}}\sum_{i\notin\mathbb{A}}\left(\frac{\partial L(\Theta,\boldsymbol{x})}{\partial x_i}\right)^2 \\ =& \mathbb{E}_{\mathbb{A}\sim\mathcal{R}}\sum_{i\notin\mathbb{A}}\mathbb{E}_{x\sim\mathcal{D}}\left(g_i(\boldsymbol{x})\right)^2 \\ =& \sum_{i=1}^{d} P_{\mathbb{A}\sim\mathcal{R}}(i\notin\mathbb{A})\mathbb{E}_{x\sim\mathcal{D}}\left(g_i(\boldsymbol{x})\right)^2, \end{aligned}$$

where $P_{\mathbb{A}\sim\mathcal{R}}(i\notin\mathbb{A})$ is the probability of $i\notin\mathbb{A}$ when $\mathbb{A}$ is sampled from $\mathcal{R}$ (i.e. the probability of $g_i(x)$ not being pruned). With the given utility constraint $U_1(S,\Theta)\geq C$ we want to minimize $\mathbb{E}_{\mathbb{A}\sim\mathcal{R}}\sum_{i\notin\mathbb{A}}\mathbb{E}_{x\sim\mathcal{D}}\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|^2$. By the previous lemma, we have:

$$\mathbb{E}_{\mathbb{A}\sim\mathcal{R},\boldsymbol{x}\sim\mathcal{D}}\operatorname{tr}(\boldsymbol{J}_{F,\mathbb{A}}(\boldsymbol{x})) = \mathbb{E}_{\mathbb{A}\sim\mathcal{R}}\sum_{i\notin\mathbb{A}}\mathbb{E}_{x\sim\mathcal{D}}\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|^2 = \sum_{i=1}^{d} P_{\mathbb{A}\sim\mathcal{R}}(i\notin\mathbb{A})\mathbb{E}_{x\sim\mathcal{D}}\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|^2.$$

For all i, we have that $0\leq P_{\mathbb{A}\sim\mathcal{R}}(i\notin\mathbb{A})\leq 1$, $\mathbb{E}_{x\sim\mathcal{D}}\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|^2>0$, and $\mathbb{E}_{x\sim\mathcal{D}}\left(g_i(\boldsymbol{x})\right)^2>0$. Therefore, for the optimal defense minimizing $\mathbb{E}_{\mathbb{A}\sim\mathcal{R},\boldsymbol{x}\sim\mathcal{D}}\operatorname{tr}(\boldsymbol{J}_{F,\mathbb{A}}(\boldsymbol{x}))$, the two restrictions apply:

- $U_1(S,\Theta)=C$.
- If $P_{\mathbb{A}\sim\mathcal{R}}(i\notin\mathbb{A})>0$, then for any j such that

$$\frac{\mathbb{E}_{x\sim\mathcal{D}}\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|^2}{\mathbb{E}_{x\sim\mathcal{D}}\left(g_i(\boldsymbol{x})\right)^2} > \frac{\mathbb{E}_{x\sim\mathcal{D}}\left\|\nabla_{\boldsymbol{x}}g_j(\boldsymbol{x})\right\|^2}{\mathbb{E}_{x\sim\mathcal{D}}\left(g_j(\boldsymbol{x})\right)^2},$$

  we must have $P_{\mathbb{A}\sim\mathcal{R}}(j\notin\mathbb{A})=1$.

If any of the above does not apply, we could trivially modify $\mathcal{R}$ to improve the lower bound while staying within the utility budget. If $U_1(S,\Theta)>C$ and a parameter is pruned with probability smaller than 1, we could slightly increase the probability of pruning that parameter. This slightly decreases utility but yields a better reconstruction lower bound. If the second restriction does not apply, we could increase the probability of pruning j and decrease the probability of pruning i to have the same training utility and obtain a higher reconstruction error lower bound. When the restrictions apply, the resulting defense follows our theorem. □

Furthermore, in locally optimal gradient pruning, adding our optimal noise instead of standard Gaussian noise after the gradient pruning step yields the same optimal defense (optimal pruning set). To show the claim, notice that in this case, the Fisher information matrix in the lower bound is:

$$\operatorname{tr}(\boldsymbol{J}_{F,S_{\text{prune}},\boldsymbol{\Sigma},\mathbb{A}}(x)) \approx \frac{1}{\epsilon}\sum_{i\notin\mathbb{A}}\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|\left|g_i(\boldsymbol{x})\right|.$$

Since the utility function remains the same, the index for optimal pruning is now

$$k_i = \frac{\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|\left|g_i(\boldsymbol{x})\right|}{|g_i(\boldsymbol{x})|^2} = \frac{\left\|\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x})\right\|}{|g_i(\boldsymbol{x})|},$$

which is equivalent to the locally optimal pruning with standard noise.

A.4. Proof of Lemma 3.8

Proof. Notice that for any given $\boldsymbol{y}\in\mathbb{R}^d$,

$$\frac{\partial f(\boldsymbol{x}+\alpha\boldsymbol{y})}{\partial\alpha} = \nabla_{\boldsymbol{x}}f(\boldsymbol{x}+\alpha\boldsymbol{y})\circ\boldsymbol{y},$$

where $\circ$ represents element-wise multiplication. Therefore

$$\left\| \frac{\partial f(\boldsymbol{x}+\alpha \boldsymbol{y})}{\partial \alpha} \right\|_{\alpha=0} = \left\| \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \circ \boldsymbol{y} \right\|.$$

When $\boldsymbol{y} \sim \mathcal{N}(0, \boldsymbol{I}_d)$, $\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \circ \boldsymbol{y}\|$ follows a normal distribution with mean 0 and variance $\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2$.

Therefore we have that

$$\frac{\left\| \frac{\partial f(\boldsymbol{x}+\alpha \boldsymbol{y})}{\partial \alpha} \right\|_{\alpha=0}}{\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|} \sim \mathcal{N}(0,1),$$

furthermore,

$$A := \frac{\sum_{i=1}^{k} \left\| \frac{\partial f(\boldsymbol{x}+\alpha \boldsymbol{x}_i)}{\partial \alpha} \right\|_{\alpha=0}}{\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|} \sim \chi^2(k).$$

Since $\mathbb{E}(A) = k$ and $\text{Var}(A) = 2k$, by Markov's inequality we have that

$$P(|A-k| > k\epsilon) \leq \frac{2k}{k^2\epsilon^2} = \frac{2}{k\epsilon^2}.$$

Since

$$\left| \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2 - \frac{1}{k} \sum_{j=1}^{k} \left\| \frac{\partial f_i(\boldsymbol{x}+\alpha \boldsymbol{x}_j)}{\partial \alpha} \right\|_{\alpha=0}^2 \right| \leq \epsilon \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2$$

is equivalent to $|A-k| > k\epsilon$, we finished the proof. □

B. OPTIMAL DEFENSE WITH ReLU ACTIVATION

In this section, we briefly discuss how we modify our optimal defenses when $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2 = 0$. The defenses for the entries where the gradients are not 0 are the same, but we deal with entries with gradient 0 separately.

For Gradient Pruning, pruning gradients that are 0 do not affect the defended gradients. Therefore, we focus on analyzing how we apply Optimal Gradient Noise. By the definition of second-order utility, the second-order utility is unaffected by the noise scale added on parameters with gradient 0. If we follow the proof of Theorem 3.5, the reconstruction lower bound scales negatively with

$$\sum_{i=1}^{d} \frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2}{\Sigma_{i,i}}.$$

Since $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} g_i(\boldsymbol{x})^2 = 0$, we have that $g_i(\boldsymbol{x})$ is constant on the support of $\mathcal{D}$ so $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2 = 0$. Therefore, the noise scale does not affect the reconstruction lower bound and any noise scale for these parameters are optimal. This typically happens when the model has activation functions that are constant on an interval (e.g. ReLU). Nonetheless, since a larger noise scale typically decreases training utility, the optimal method would be not to add noise to the gradients.

However, the above case does not completely cover problems in the locally optimal defenses. In the locally optimal versions, we used the values at $\boldsymbol{x}$ as approximations of expectations. Therefore we might have $\|\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x})\|^2 > 0$. In this case, theoretically we should set the noise to be as large as possible, which deviates from reality. To resolve this problem, a good solution would be to set an upper limit to the noise scale and clip the noise scales added to the gradients.

To summarize, for optimal gradient pruning, pruning gradients with a scale of 0 is trivial. For optimal gradient noise (and its application in DP-SGD), we set an upper limit to the noise scale.

C. EXPERIMENT DETAILS

For all datasets and algorithms, we used the implementation in Algorithm 1 with $k = 10$ as the number of samples and $c = 10^{-6}$ as the small constant. When comparing our optimal noise with DP-SGD, we applied clipping threshold 1 for DP-SGD and included the same clipping step before adding our optimal noise.

C.1. Experiments with MNIST

For experiments on the MNIST dataset, we used a Convolutional Neural Network with 120k parameters. To avoid the special case where the model utilizes the ReLU activation function, we use the LeakyReLU activation function instead. We trained the model on a subset of size 4096 from the whole dataset and used SGD algorithms with batch size 64. We simulated 4 clients each possessing one-fourth of the dataset (1024 samples). When training, each mini-batch contains data from all 4 clients, with each client providing 16 samples to form a 64-sample mini-batch. The gradients are computed and defended separately and then averaged to be provided to the central server. For the training process with DP-SGD (or our optimal noise) applied, we used the Adam optimizer with learning rate 10^{-3}. For the training process with gradient pruning (or our optimal pruning) applied, we used Adam with learning rate 5×10^{-4}. For the reconstruction process, we used the Inverting Gradients algorithm with a budget of 2000 updates. The experiments are conducted on a Nvidia RTX 2050.

For the scatter plot, we trained the model on 1 batch of 64 samples for 5 gradient descent updates. Each gradient used for update is the average of 4 gradients calculated and defensed separately from 4 clients. We used the Adam optimizer with learning rate 10^{-3}.

C.2. Experiments with CIFAR-10

For experiments on the CIFAR-10 dataset, we used a 2.9M parameter ConvNet with the LeakyReLU activation. For the scatter plot, we measured the training utility by training on a batch of 8 samples. For comparing DP-SGD with our optimal noise, we used the Adam optimizer for 5 steps with the learning rate being 10^{-4}. For comparing gradient pruning with our optimal pruning, we used Adam with the learning rate being 5×10^{-5} (for slower convergence). Every update we simulated 4 clients each calculating and defending gradients separately like in MNIST. For reconstruction, we reconstructed the images 2-at-a-time from the shared model gradients using the Inverting Gradients algorithm with a budget of 1000 updates. The experiments are conducted on a Nvidia RTX 4090D.

D. ADDITIONAL EXPERIMENTS

D.1. Experiments on MNIST

The MNIST dataset consists of 28×28 grayscale images of handwritten digits, serving as a simple test for our algorithm.

D.1.1. Gradient Pruning

We apply different pruning thresholds to a randomly initialized Convolutional Neural Network (Fukushima, 1969), using 4 batches of 16 images to compute gradients. These gradients were defended using gradient pruning and our optimal gradient pruning, followed by an Inverting Gradients attack. Figure 5 shows that our method consistently achieves higher Mean Squared Error (MSE) and lower Peak Signal-to-Noise Ratio (PSNR) at commonly used high pruning ratios, indicating stronger defenses.

To assess training utility, we trained the models under a federated learning setting. Figure 6 shows that 80% optimal pruning outperforms 90% gradient pruning in training speed, while we showed that they have similar privacy in Figure 5. The scatter plot in Figure 7 shows the privacy-utility trade-off for a wider range of pruning ratios, indicating the superior privacy-utility trade-off of our method.

D.1.2. DP-SGD

We also evaluated our defense on DP-SGD. As shown in Figure 8, our optimal noise achieves comparable performance to DP-SGD in terms of defense at the same noise scale, while our algorithm has faster learning speed (Figure 9). The scatter plot in Figure 10 further demonstrates the improved privacy-utility trade-off of our approach. Visualization of the reconstruction in Figure 11 shows better protection against attacks using our optimal noise for the same level of training utility.

D.2. Comparison on Generalization Loss

Though our theoretical analysis focuses on optimizing the training loss, we also calculated the validation loss of the trained models with the same setup as Figure 9. Training with noise scale 0.1 using our optimal noise resulted in test accuracy 0.910

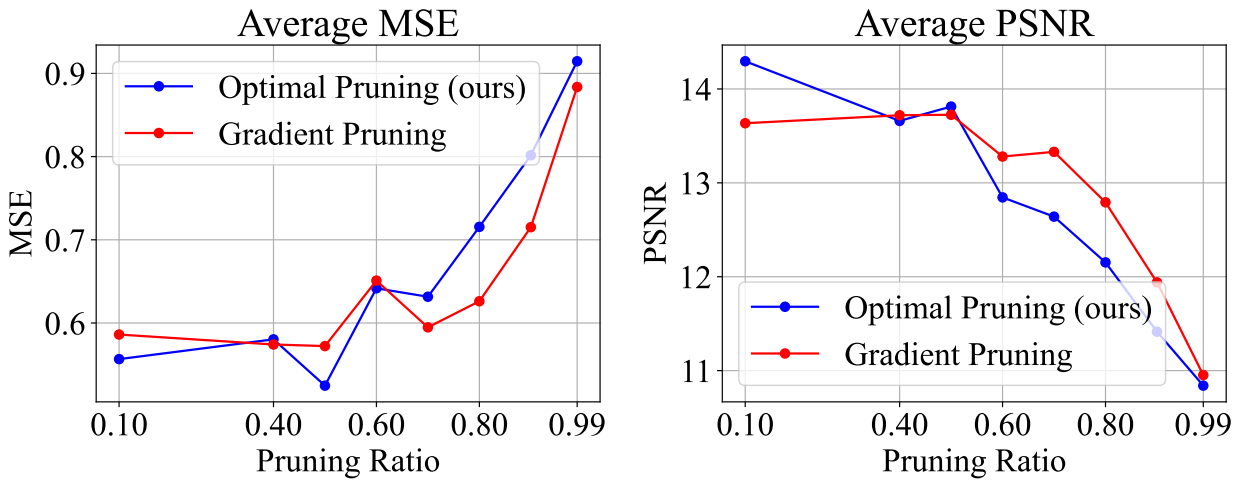

Figure 5. Average reconstruction indexes based on Gradient Inversion with batch size 16.

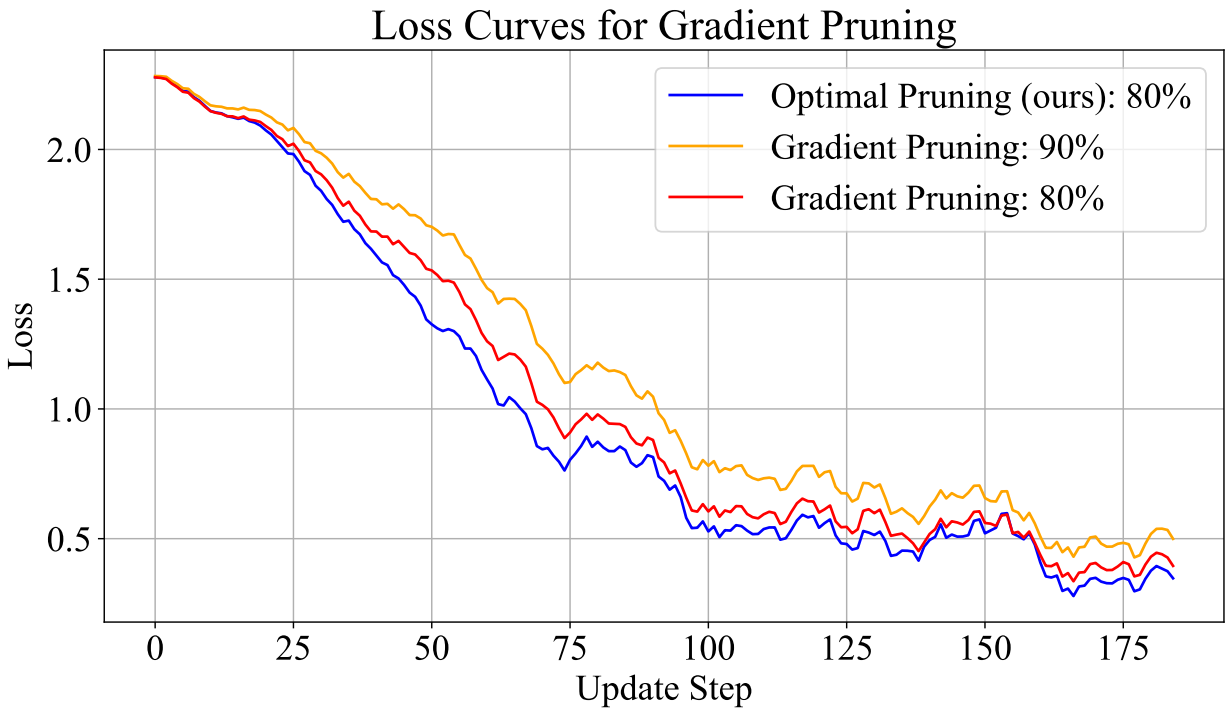

Figure 6. Training curves of CNN on MNIST with 80% & 90% gradient pruning and 80% optimal pruning (smoothed with window size 8). 80% optimal pruning outperforms 90% gradient pruning in training.

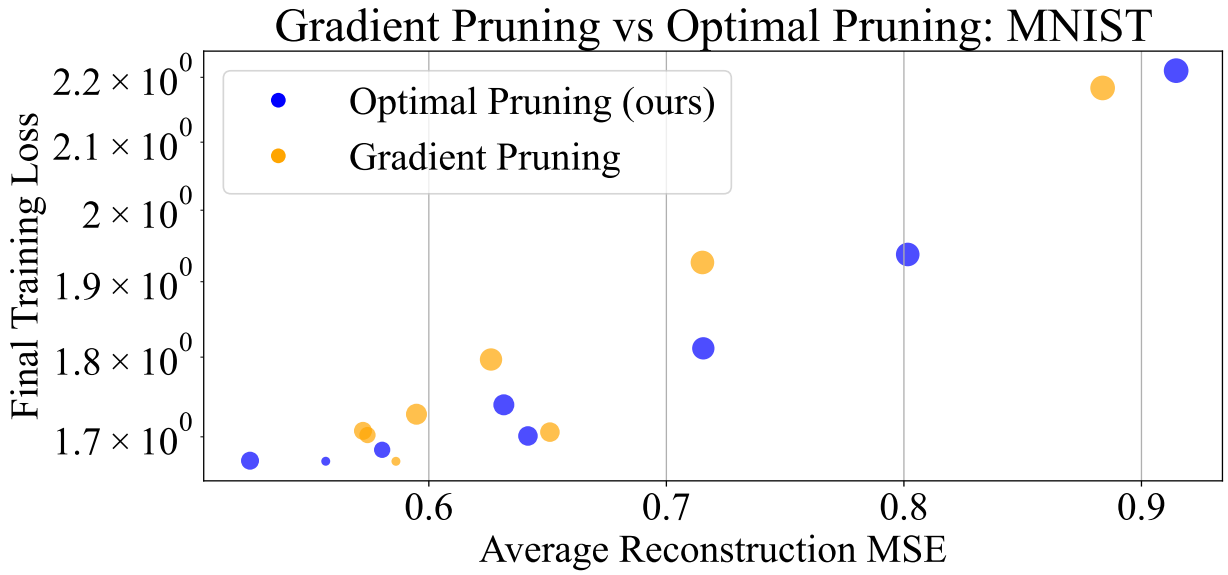

Figure 7. Scatter plot of gradient pruning and our optimal pruning on MNIST. X-axis: average reconstruction MSE. Y axis: Training loss on 64 samples. Size of points: Pruning ratio.

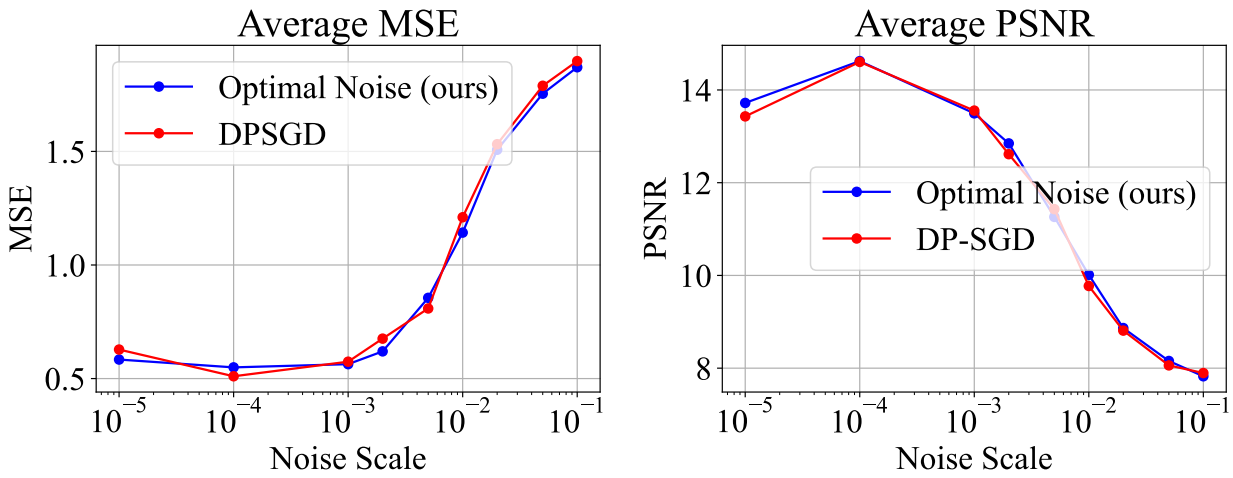

Figure 8. Average reconstruction indexes based on Gradient Inversion for DP-SGD. The noise scale equals the Frobenius norm of the covariance matrix.

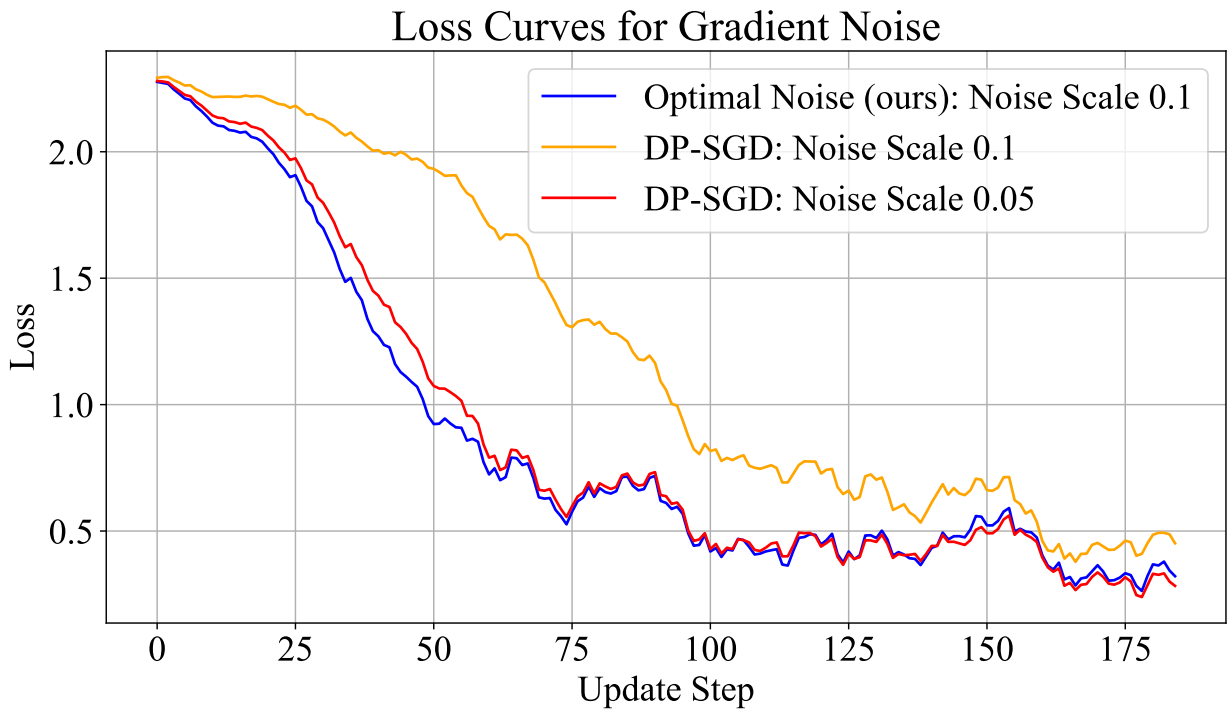

Figure 9. Training curves of CNN on the MNIST dataset. (Smoothed with window size 8) Optimal noise with a scale of 0.1 outperforms DP-SGD with a scale of 0.1.

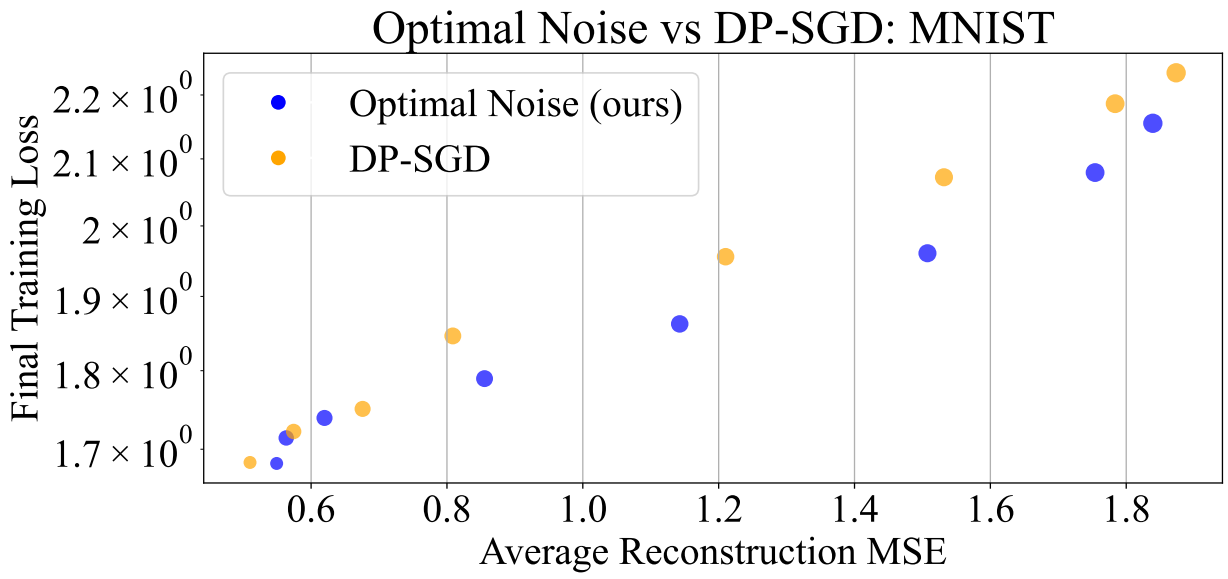

Figure 10. Comparison of optimal noise and DP-SGD on MNIST. X-axis: average reconstruction MSE. Y axis: Training loss on 64 samples.

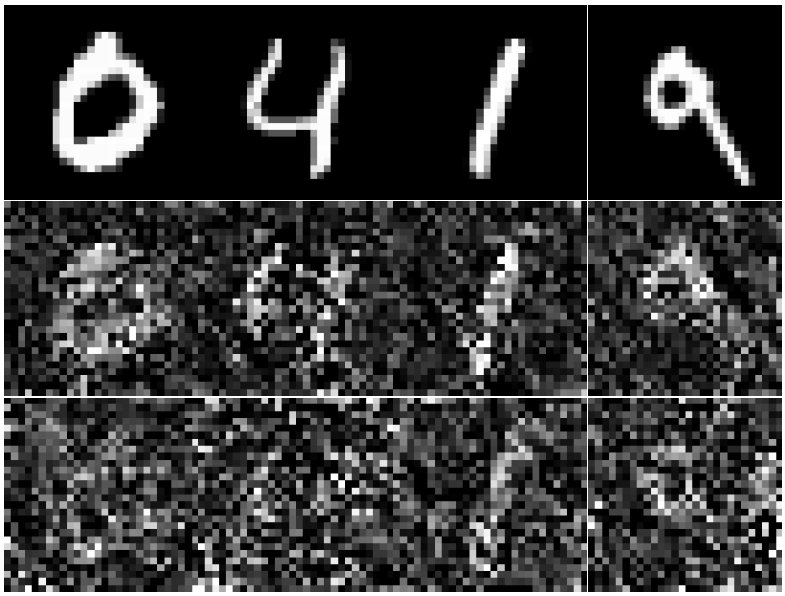

Figure 11. Reconstruction from the MNIST dataset with batch size 4. First row: ground truth. Second row: DP-SGD with scale 0.05. Third row: optimal noise with scale 0.1. Our method has better privacy when the performance in training is similar.

and using noise scale 0.1 with standard Gaussian noise resulted in test accuracy 0.886, indicating that our method performed better.

D.3. Effect of Noise Scale on DP-SGD in CIFAR-10

Since our experiments in the previous sections only cover small noise scales, we visualize the effects of the larger noise scales on CIFAR-10 in Figure 12 and 13. With higher noise scales, it is clearer that our optimal noise has higher training utility and (slightly) higher reconstruction error than DP-SGD.

D.4. Average Optimal Noise Through the Training Process

We additionally show how our method differs from DP-SGD by visualizing average noise added to each layer during the training process. The experiment utilized the MNIST dataset. As in Figure 14, some layers are given significantly higher noise than other layers. We additionally include how many parameters each layer contains in Table 1 and a code snippet of

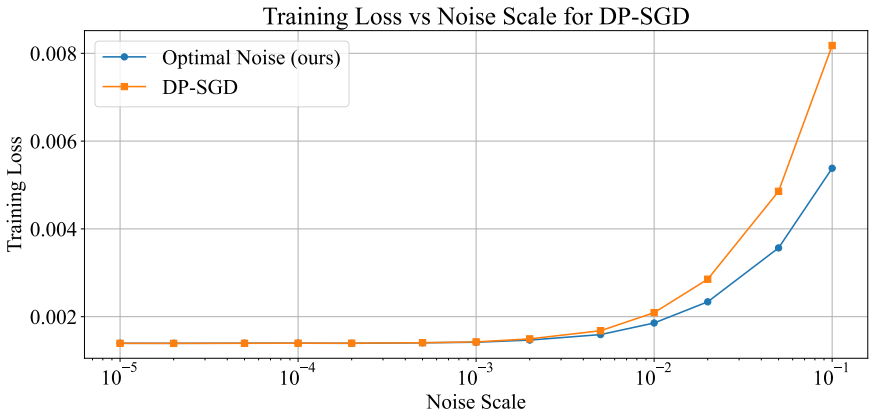

Figure 12. Effect of Noise Scale on Training Utility.

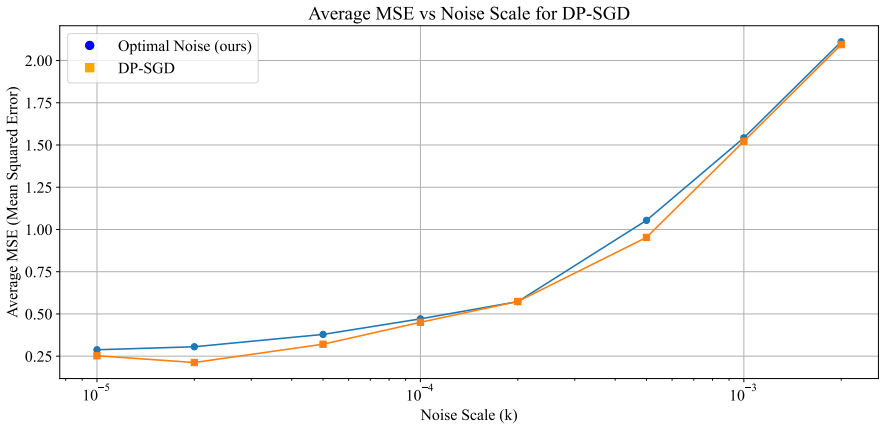

Figure 13. Effect of Noise Scale on Average Reconstruction MSE.

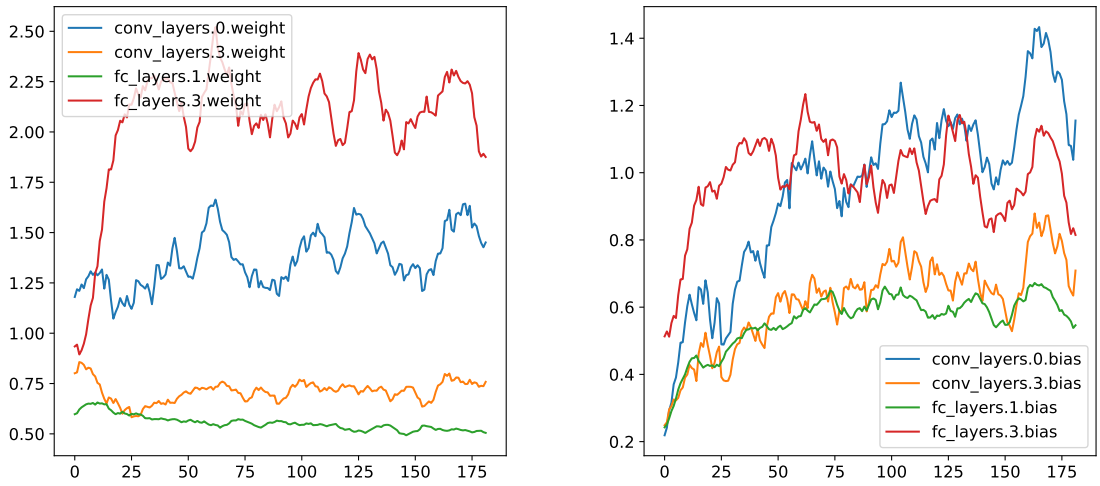

Figure 14. Average noise each layer smoothed by window size 10. Left: weight matrices. Right: bias matrices.

Layer	Parameters
conv_layers.0.weight	288
conv_layers.0.bias	32
conv_layers.3.weight	18,432
conv_layers.3.bias	64
fc_layers.1.weight	100,352
fc_layers.1.bias	32
fc_layers.3.weight	320
fc_layers.3.bias	10

Table 1. Number of parameters in the neural network.

the model in Pytorch for reference.

```
class SimpleConvNet(nn.Module):
    def __init__(self, num_classes=10, num_channels=1):
        super(SimpleConvNet, self).__init__()

        # Convolutional layers
        self.conv_layers = nn.Sequential(
            # conv_layers.0 (Conv2d)
            nn.Conv2d(num_channels, 32, kernel_size=3, padding=1),
            nn.LeakyReLU(),
            nn.MaxPool2d(2),

            # conv_layers.3 (Conv2d)
            nn.Conv2d(32, 64, kernel_size=3, padding=1),
            nn.LeakyReLU(),
            nn.MaxPool2d(2)
        )

        # Fully connected layers
        self.fc_layers = nn.Sequential(
            nn.Flatten(),
            # fc_layers.1 (Linear)
            nn.Linear(64 * 7 * 7, 32),
            nn.LeakyReLU(),
            # fc_layers.3 (Linear)
            nn.Linear(32, num_classes)
        )

    def forward(self, x):
        x = self.conv_layers(x)
        x = self.fc_layers(x)
        return x
```