Appendices

A Background and Related Works

A.1 Differential Privacy

Differential Privacy (DP) (Dwork, 2011; Dwork et al., 2006; 2014) is a theoretical privacy framework for aggregate data analysis. It ensures the output distributions of a randomized algorithm are indistinguishable with a certain probability when running on two neighboring datasets differing in one record or bounded by a distance metric.

Definition 1. *((ϵ, δ)-Differential Privacy) A randomized mechanism $\mathcal{M} : \mathbf{D} \rightarrow \mathbf{R}$ with domain $\mathbf{D}$ and range $\mathbf{R}$ satisfies (ϵ, δ)-differential privacy if for any two adjacent inputs $\mathcal{D}, \mathcal{D}' \in \mathbf{D}$ and for any subset of outputs $\mathbf{S} \subseteq \mathbf{R}$ it holds that*

$$Pr(\mathcal{M}(\mathcal{D}) \in \mathbf{S}) \leq e^{\epsilon} Pr(\mathcal{M}(\mathcal{D}') \in \mathbf{S}) + \delta,$$

where ϵ is the privacy budget and δ is the probability that privacy is broken.

The common mechanism to achieve (ϵ, δ)-DP is Gaussian Mechanism (**GM**) that adds calibrated noise to the output.

Theorem 4. *Gaussian Mechanism(Dwork et al., 2014). Let $\mathcal{G} : \mathbb{R}^v \rightarrow \mathbb{R}^w$ be an arbitrary w-dimensional function, and its sensitivity $\Delta_{\mathcal{G}} = \max_{\mathcal{D},\mathcal{D}'} \|\mathcal{G}(\mathcal{D}) - \mathcal{G}(\mathcal{D}')\|_2$. A Gaussian Mechanism $\mathcal{M}$ with σ adds element-wise noise $\mathcal{N}(0, \sigma^2)$ to the output. The mechanism $\mathcal{M}$ is (ϵ, δ)-DP, with*

$$\epsilon \in (0, 1],\ \sigma \geq \sqrt{2 \ln(1.25/\delta)} \Delta_{\mathcal{G}} / \epsilon.$$

Gradient Perturbation. Gradient perturbation is a widely used technique that injects perturbation to the gradient of each parameter to guarantee DP for deep learning models. Gradient perturbation injects calibrated noise to the gradient during training with the following objective function and update steps.

$$\mathcal{J}(\theta_t) = \frac{1}{N} \sum_{i=1}^{N} l(\mathbf{x}_i, y_i, \theta_t),$$
$$\theta_{t+1} = \theta_t - \eta(\nabla \mathcal{J}(\theta_t) + \mathbf{p}),$$

where θ_t denotes the parameter at training step t, $\nabla \mathcal{J}(\theta_t)$ denotes the gradient, and $\mathbf{p}$ denotes the gradient perturbation. The gradient $\nabla \mathcal{J}(\theta_t)$ is bounded by the clipping norm or constrained by Lipschitz continuity of loss function l.

Song et al. Song et al. (2013) first propose the gradient perturbation method by injecting perturbation to the gradients during parameter updates with stocastic gradient descent (SGD). Bassily et al. Bassily et al. (2014) improve the gradient perturbation by leveraging privacy amplification via sampling Beimel et al. (2010) (Lemma II.2 in Bassily et al. (2014)) and strong composition Dwork et al. (2010) (Lemma II.3 in Bassily et al. (2014)) to achieve a tighter bound. Abadi et al. Abadi et al. (2016) make further improvement by proposing a novel privacy composition tool: moments accountant, which can compute the overall privacy cost during training and achieve a tighter bound. Shokri et al. Shokri & Shmatikov (2015) propose the gradient perturbation method under the distributed learning scenario. Wang et al. Wang et al. (2017) replace SGD optimizer used in previous work with stochastic variance-reduced gradient (SVRG) Xiao & Zhang (2014) to achieve a faster optimization. However, it requires the loss function l to be convex, G-Lipschitz and β-smooth. Lee et al. Lee & Kifer (2018) and Yu et al. Yu et al. (2019) improve the gradient perturbation method by dynamically allocating the privacy budget per iteration and leverage zero-concentrated DP (zCDP) Lee & Kifer (2018) to analyze the privacy cost. In this paper, we mainly leverage the input perturbation designed for certified robustness to achieve DP simultaneously and use additional gradient perturbation as needed to achieve overall DP.

Input Perturbation. Input perturbation is a technique that directly adds calibrated noise to input data to achieve DP. As a result, the objective function is generalized as,

$$\mathcal{J}_{priv}(\theta) = \frac{1}{N}\sum_{i=1}^{N} l((\hat{\mathbf{x}}_{(i)}, \hat{y}_{(i)}), \theta),$$
$$\theta_{priv} = \arg\min \mathcal{J}_{priv}(\theta),$$

where θ denotes the model parameters, $\mathcal{J}_{priv}$ denotes the perturbed objective function due to the perturbed input, θ_{priv} denotes the parameters of the final model, N is the size of the training dataset, $(\hat{\mathbf{x}}_{(i)}, \hat{y}_{(i)})$ denote the perturbed data and l is the loss function.

Fukuchi et al. Fukuchi et al. (2017) first attempted to use Taylor expansion to transform input perturbation into gradient perturbation. Although input perturbation framework theoretically guarantees that model trained with perturbed inputs is DP, this work imposes several constraints on the loss function, which cannot be practically applied with deep learning systems. Kang et al. Kang et al. (2020b) propose an input perturbation that generalizes the constraints on the loss function to less strict conditions. They also take a further step by finding that different training data will affect the model in different ways Kang et al. (2020a). However, this work requires a pre-trained model that should also be DP, which also requires privacy budget. In summary, all the above works impose strict constraints on the loss function to analyze DP for input perturbation. These constraints can not be satisfied by typical deep learning models. In this paper, we relax the constraints of Kang et al. (2020b), propose tools for transforming the input perturbation into gradient perturbation, and formally analyze its DP guarantee for deep learning systems. As a result, we can leverage input perturbation to achieve both certified robustness and DP simultaneously.

A.2 CERTIFIED ROBUSTNESS

Adversarial Examples. Adversarial examples are designed by adding small perturbations to clean examples, which are imperceptible to human eyes but can easily fool a deep learning model to produce incorrect output. Generating an adversarial example $\mathbf{x}'$ can be expressed as a constrained optimization problem. For untargeted adversarial examples, it can be expressed as: given a clean input $\mathbf{x}$, its label y, and a classifier f, minimize $L(\mathbf{x}, \mathbf{x}')$, such that the prediction score at label y is not the maximum among all labels: $f_y(\mathbf{x}') \neq \max_{j=1,..,c} f_j(\mathbf{x}')$, where L is a distance metric, such as l_2 or l_∞, and c is the number of classes. The quality of adversarial examples is measured by both the perturbation scale, i.e. $L(\mathbf{x}, \mathbf{x}')$, and the misclassfication rate or attack success rate.

Certified Robustness. Certified robustness is a principled technique to defend against adversarial examples. A classifier f is certified robust if for any input $\mathbf{x} \in \mathcal{R}^v$, its prediction $\max_{j=1,...,c} f_j(\mathbf{x})$ is constant within some set around $\mathbf{x}$, that is:

$$\forall \tau \in l_p(\kappa) : f_y(\mathbf{x}+\tau) > \max_{j\neq y} f_j(\mathbf{x}+\tau),$$

where y is the label of $\mathbf{x}$, and $l_p(\kappa)$ denotes $\forall \tau \in \mathcal{R}^v, \|\tau\|_p \leq \kappa$.

Randomized Smoothing. Randomized smoothing is the state-of-the-art approach to achieve certified robustness. Given a classifier f, the random smoothing technique converts f into a smooth classifier g, s.t., for input $\mathbf{x}$, $\tilde{f}$ returns

$$\tilde{f}(\mathbf{x}) = \arg\max_{j=1,...,c} Pr[f(\mathbf{x}+\mathbf{b}) = j],$$

where $\mathbf{b} \sim \mathcal{N}(0, \sigma^2 I)$. Randomized smoothing calculates a certified radius κ, s.t.,

$$\forall \tau \in l_p(\kappa) : \tilde{f}(\mathbf{x}+\tau) = y,$$

where y is the label of $\mathbf{x}$, and $l_p(\kappa)$ denotes $\forall \tau \in \mathcal{R}^v, \|\tau\|_p \leq \kappa$. Theoretically, randomized smoothing only works during the testing phase by injecting input perturbation to the testing samples. However, there is a common practice during the training phase, which perturbs the training samples with the input perturbation to improve the utility performance.

PixelDP. Lecuyer et al. Lecuyer et al. (2019) propose PixelDP to achieve certified robustness by considering an input image as a database in DP parlance and each pixel of the image as each record in DP. PixelDP shows that adding a randomization layer in the model to preserve DP on image pixels guarantees certified robustness of the model against adversarial examples.

Randomized Smoothing. Randomized smoothing is another technique that adds random noise to the input for achieving certified robustness and has been shown to outperform PixelDP with tighter robustness guarantee. Li et al. Li et al. (2019) derive a certified bound for robustness to adversarial examples using R*é*nyi Divergence Gil (2011) by adding additive random noise to the input. Cohen et al. Cohen et al. (2019) leveraged Neyman-Person lemma to analyze the correlation between the highest scored class and the second highest class. Compared to the previous work, they provide a tight certified robustness guarantee for the model. All above-mentioned work are certified within an L_2 radius which means that the adversary cannot alter the prediction within a L_2 unit ball. Lee et al. Lee et al. (2019) provide certified robustness for discrete cases where the adversary is L_0 bounded (the number of pixel changes in a figure). Salman et al. Salman et al. (2019) further employ adversarial training to improve the certified robustness of models.

To provide more efficient certified robustness, Salman et al. Salman et al. (2020) also propose to obtain a certified robust classifier from a fixed pre-trained model. Randomized Smoothing is then applied during testing and provides certified robustness without retraining the pre-trained model. In this paper, we adopt a similar setting as Salman et al. (2020) but focus on achieving both certified robustness and DP at the same time via randomized smoothing (input perturbation).

A.3 DIFFERENTIAL PRIVACY WITH CERTIFIED ROBUSTNESS

There are few works on simultaneously achieving both DP and certified robustness. Phan et al. Phan et al. (2019) first attempt to propose a framework called Secure-SGD to simultaneously achieve certified robustness and differential privacy. They use a PixelDP Lecuyer et al. (2019) based approach to achieve certified robustness and propose a Heterogeneous Gaussian Mechanism (HGM) to improve the performance of certified robustness. To achieve DP, they use gradient perturbation and propose a Heterogeneous Gaussian Mechanism (HGM) by adding heterogeneous Gaussian noise instead of element-wise Gaussian noise. There are two major limitations of this work: 1) The random perturbation used to achieve certified robustness during training should have contributed a certain degree of randomness to preserve DP but is ignored. Instead, additional Gaussian perturbation is added onto the gradient to achieve DP. 2) The robustness bound provided by PixelDP is loose compared with other randomized smoothing Li et al. (2019); Cohen et al. (2019); Lee et al. (2019) based approaches for certified robustness.

Another work from Phan et al. Phan et al. (2020) developed StoBatch algorithm to guarantee DP and certified robustness. It first leverages Autoencoder (AE) Hinton & Zemel (1994) and functional mechanism (objective perturbation) Zhang et al. (2012) to reconstruct input examples with DP. Then, these reconstructed DP data is used to train a deep neural network for classification. To make the neural network robust and DP, they apply adversarial training Tramèr et al. (2017) and functional mechanism as mentioned earlier during training. Although this framework achieves better performance compared to their previous work Phan et al. (2019), there are still two limitations: 1) The approach is only applicable to simple architectures of AE due to the constraints of function mechanism and hence cannot learn complicated representation on high-dimensional or complex data. 2) Similar to Secure-SGD Phan et al. (2019), the robustness bound provided by PixelDP is loose.

Although these works attempt to achieve both DP and certified robustness, there are several major limitations. In our work, we focus on addressing these limitations and compare our approach with these two approaches in the experimental studies.

B BRIEF PROOF OF RANDOMIZED SMOOTHING

Given denoiser g and pre-trained classifier h, let $f = g \circ h$, a randomized smoothed classifier $\tilde{f}$ works as follows,

$$\tilde{f}(\mathbf{x}) = \arg\max_{j=1,\ldots,c} Pr[f(\mathbf{x}+\mathbf{b}) = j], \tag{6}$$

where input perturbation $\mathbf{b} \sim \mathcal{N}(0, \sigma^2 I)$.

Suppose the probability that $\tilde{f}$ return the class y of $\mathbf{x}$ is $p_y = \mathcal{P}(\tilde{f}(\mathbf{x}) = y)$ and the probability for "runner-up" class b is $p_b = \mathcal{P}(\tilde{f}(\mathbf{x}) = b)$. The smoothed classifier $\tilde{f}$ is certified robust around $\mathbf{x}$ within the radius

$$\kappa = \frac{\sigma}{2}(\Phi^{-1}(p_y) - \Phi^{-1}(p_b)),$$

where Φ^{-1} is the inverse of the standard Gaussian CDF. However, computing p_y and p_b is not practical for deep learning models. To solve this challenge, Cohen et al. (Cohen et al., 2019) used Monte Carlo sampling to get the estimation $\underline{p_y}$ and $\overline{p_b}$ s.t. $\underline{p_y} \leq p_y$ and $\overline{p_b} \geq p_b$ with arbitrarily high probability. $\underline{p_y}$ and $\overline{p_b}$ is then used to substitute p_y and p_b in Equation (6) and the certified radius κ is obtained.

C THEOREM 5 (**wEGM**) AND PROOF

w-Element Gaussian Mechanism (**wEGM**) is actually a rewrite of the traditional Gaussian mechanism and can guarantee that $\mathcal{M}$ is still DP after applying perturbation to each component of the output.

Theorem 5. *w-Element Gaussian Mechanism. Let $\mathcal{G} : \mathbb{R}^v \rightarrow \mathbb{R}^w$ be an arbitrary w-dimensional function, and $\Delta_{\mathcal{G}} = \max_{\mathcal{D},\mathcal{D}'} \|\mathcal{G}(\mathcal{D}) - \mathcal{G}(\mathcal{D}')\|_2$. A w-Element Gaussian Mechanism $\mathcal{M}$ with σ adds noise to each of the w elements of the output. The mechanism $\mathcal{M}$ is (ϵ, δ)-**DP**, with*

$$\epsilon \in (0, 1],\ \sigma \geq \sqrt{2\ln(1.25/\delta)}\Delta_{\mathcal{G}}/\epsilon.$$

As can be seen in the definition of the theorem, the perturbation in **wEGM** is i.i.d. with the same scale σ. The proof of Theorem 5 is as follows:

Proof. The privacy loss of an output $\boldsymbol{o}$ is defined as:

$$\mathcal{L}(\boldsymbol{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}') = ln\frac{Pr[\mathcal{M}(\mathcal{D}, \mathcal{G}, \sigma)] = \boldsymbol{o}}{Pr[\mathcal{M}(\mathcal{D}', \mathcal{G}, \sigma)] = \boldsymbol{o}} \tag{7}$$

Given $\boldsymbol{v} = \mathcal{G}(\mathcal{D}) - \mathcal{G}(\mathcal{D}')$, we have

$$\begin{aligned}
&|\mathcal{L}(\boldsymbol{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= |ln\frac{Pr[\mathcal{M}(\mathcal{D}, \mathcal{G}, \sigma)] = \boldsymbol{o}}{Pr[\mathcal{M}(\mathcal{D}', \mathcal{G}, \sigma)] = \boldsymbol{o}}| \\
&= |ln\frac{Pr[\mathcal{G}(\mathcal{D}) + \mathcal{N}(0, \sigma^2)] = \boldsymbol{o}}{Pr[\mathcal{G}(\mathcal{D}') + \mathcal{N}(0, \sigma^2)] = \boldsymbol{o}}| \\
&= |ln\frac{\prod_{k=1}^{w} exp(-\frac{1}{2\sigma^2}(\boldsymbol{o}_k - \mathcal{G}(\mathcal{D})_k)^2}{\prod_{k=1}^{w} exp(-\frac{1}{2\sigma^2}(\boldsymbol{o}_k - \mathcal{G}(\mathcal{D})_k + \boldsymbol{v}_k)^2}| \\
&= \frac{1}{2\sigma^2}|\sum_{k=1}^{w}(\boldsymbol{o}_k - \mathcal{G}(\mathcal{D})_k)^2 - (\boldsymbol{o}_k - \mathcal{G}(\mathcal{D})_k + \boldsymbol{v}_k)^2|.
\end{aligned}$$

Let $\boldsymbol{p}$ denotes $\boldsymbol{o} - \mathcal{G}(\mathcal{D})$, then $\boldsymbol{p}_k = \boldsymbol{o}_k - \mathcal{G}(\mathcal{D})_k$, and $\boldsymbol{p}_k \sim \mathcal{N}(0, \sigma^2)$, Then we have

$$\begin{aligned}
&|\mathcal{L}(\boldsymbol{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= \frac{1}{2\sigma^2}|\sum_{k=1}^{w}(\boldsymbol{p}_k^2 - (\boldsymbol{p}_k + \boldsymbol{v}_k)^2)| \\
&= \frac{1}{2\sigma^2}|\|\boldsymbol{p}\|^2 - \|\boldsymbol{p} + \boldsymbol{v}\|^2|.
\end{aligned}$$

According to the fact that a spherically symmetric normal distribution is independent of the orthogonal basis, we can always find a basis $b_1, b_2, ..., b_w$ that is aligned with $\boldsymbol{p}$. Then we can depict $\boldsymbol{p} = \sum_{k=1}^{w} {p'}_k$ with ${p'}_k$ being the component on basis b_k and ${p'}_k \sim \mathcal{N}(0, \sigma^2)$. Without loss of generality, we assume b_1 is parallel to $\boldsymbol{v}$, which means that

$$\|\boldsymbol{p}\|^2 = \sum_{k=1}^{w} \|p'_k\|^2,$$

$$\|\boldsymbol{p}+\boldsymbol{v}\|^2 = \|p'_1 + \boldsymbol{v}\|^2 + \sum_{k=2}^{w} \|p'_k\|^2.$$

Therefore, we have

$$\begin{aligned}
&|\mathcal{L}(o; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= \frac{1}{2\sigma^2} |\|\boldsymbol{p}\|^2 - \|\boldsymbol{p}+\boldsymbol{v}\|^2| \\
&= \frac{1}{2\sigma^2} |\|p'_1 + \boldsymbol{v}\|^2 - \|p'_1\|^2| \\
&\leq \frac{1}{2\sigma^2} |\Delta_{\mathcal{G}}^2 + 2|p'_1|\Delta_{\mathcal{G}}|
\end{aligned}$$

Following the proof of Theorem A.1 in Dwork et al. (2014), we can show that the mechanism $\mathcal{M}$ is (ϵ, δ)-DP, with

$$\epsilon \in (0, 1], \sigma \geq \sqrt{2ln(1.25/\delta)}\Delta_{\mathcal{G}}/\epsilon.$$

□

D PROOF OF THEOREM 1 (MGM)

Proof. The privacy loss of an output o is defined as:

$$\mathcal{L}(\mathbf{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}') = ln \frac{Pr[\mathcal{M}(\mathcal{D}, \mathcal{G}, \mathbf{\Sigma})] = \mathbf{o}}{Pr[\mathcal{M}(\mathcal{D}', \mathcal{G}, \mathbf{\Sigma})] = \mathbf{o}}$$

Given $\mathbf{\Sigma}$, we have

$$\begin{aligned}
&|\mathcal{L}(\mathbf{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= |ln \frac{Pr[\mathcal{M}(\mathcal{D}, \mathcal{G}, \mathbf{\Sigma})] = \mathbf{o}}{Pr[\mathcal{M}(\mathcal{D}', \mathcal{G}, \mathbf{\Sigma})] = \mathbf{o}}| \\
&= |ln \frac{Pr[\mathcal{G}(\mathcal{D}) + \mathcal{N}(0, \mathbf{\Sigma})] = \mathbf{o}}{Pr[\mathcal{G}(\mathcal{D}') + \mathcal{N}(0, \mathbf{\Sigma})] = \mathbf{o}}| \\
&= |ln \frac{exp(-\frac{1}{2}(\mathbf{o} - \mathcal{G}(\mathcal{D}))^\intercal \mathbf{\Sigma}^{-1} (\mathbf{o} - \mathcal{G}(\mathcal{D}))}{exp(-\frac{1}{2}(\mathbf{o} - \mathcal{G}(\mathcal{D}'))^\intercal \mathbf{\Sigma}^{-1} (\mathbf{o} - \mathcal{G}(\mathcal{D}'))}|
\end{aligned}$$

Given the transformation matrix $\mathbf{A}_{(i)}$, we denote $\mathbf{M}$ as $a^2\mathbf{A}_{(i)}\mathbf{A}_{(i)}^\intercal$, which is a symmetric matrix. Therefore, $\mathbf{M}$ can be decomposed by Singular Value Decomposition as:

$$\mathbf{M} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\intercal,$$

where $\mathbf{U}, \mathbf{\Lambda} \in \mathbb{R}^{w \times w}$, and $\mathbf{\Lambda}$ is a diagonal matrix with each element on its diagonal being the singular values of $\mathbf{M}$. We construct a matrix $\mathbf{K} \in \mathbb{R}^{w \times w}$, s.t., $\mathbf{K} = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U}^\intercal$, then

$$\mathbf{M} = \mathbf{K}\mathbf{K}^\intercal,$$

Let $\mathbf{p}$ denotes $\mathbf{o} - \mathcal{G}(\mathcal{D})$, then $\mathbf{p} \sim \mathcal{N}(0, \mathbf{\Sigma})$. Construct a Gaussian random variable $\tilde{\mathbf{p}} \sim \mathcal{N}(0, \sigma^2)$. We can have $\mathbf{p} = \mathbf{K}\tilde{\mathbf{p}}$. Given $\mathbf{v} = \mathcal{G}(\mathcal{D}) - \mathcal{G}(\mathcal{D}')$, we have

$$\begin{aligned}
&|\mathcal{L}(\mathbf{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= \frac{1}{2} |(\mathbf{p}+\mathbf{v})^\intercal \mathbf{\Sigma}^{-1} (\mathbf{p}+\mathbf{v}) - \mathbf{p}^\intercal \mathbf{\Sigma}^{-1} \mathbf{p}|
\end{aligned}$$

Construct $\hat{\mathbf{v}} = \mathbf{K}^{-1}\mathbf{v}$, then the above formula becomes

$$\begin{aligned}
&|\mathcal{L}(\mathbf{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= \frac{1}{2}|(\mathbf{K}\tilde{\mathbf{p}} + \mathbf{K}\hat{\mathbf{v}})^\intercal \mathbf{\Sigma}^{-1}(\mathbf{K}\tilde{\mathbf{p}} + \mathbf{K}\hat{\mathbf{v}}) - \mathbf{p}^\intercal \mathbf{\Sigma}^{-1}\mathbf{p}| \\
&= \frac{1}{2\sigma^2}|(\mathbf{K}\tilde{\mathbf{p}} + \mathbf{K}\hat{\mathbf{v}})^\intercal (\mathbf{K}\mathbf{K}^\intercal)^{-1}(\mathbf{K}\tilde{\mathbf{p}} + \mathbf{K}\hat{\mathbf{v}}) - \\
&\qquad (\mathbf{K}\tilde{\mathbf{p}})^\intercal (\mathbf{K}\mathbf{K}^\intercal)^{-1}(\mathbf{K}\tilde{\mathbf{p}})| \\
&= \frac{1}{2\sigma^2}|(\tilde{\mathbf{p}} + \hat{\mathbf{v}})^\intercal \mathbf{K}^\intercal (\mathbf{K}\mathbf{K}^\intercal)^{-1}\mathbf{K}(\tilde{\mathbf{p}} + \hat{\mathbf{v}}) - \\
&\qquad \tilde{\mathbf{p}}^\intercal \mathbf{K}^\intercal (\mathbf{K}\mathbf{K}^\intercal)^{-1}\mathbf{K}\tilde{\mathbf{p}}| \\
&= \frac{1}{2\sigma^2}|(\tilde{\mathbf{p}} + \hat{\mathbf{v}})^\intercal (\tilde{\mathbf{p}} + \hat{\mathbf{v}}) - \tilde{\mathbf{p}}^\intercal \tilde{\mathbf{p}}| \\
&= \frac{1}{2\sigma^2}|\|\tilde{\mathbf{p}}\|^2 - \|\tilde{\mathbf{p}} + \hat{\mathbf{v}}\|^2|,
\end{aligned}$$

where $\tilde{\mathbf{p}}_k \sim \mathcal{N}(0, \sigma^2)$ and $\hat{\mathbf{v}} = \mathbf{K}^{-1}\mathbf{v}$. This is very similar to the proof in 5, except for the $\hat{\mathbf{v}}$. We can get the similar result as,

$$\begin{aligned}
&|\mathcal{L}(o; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= \frac{1}{2\sigma^2}|\|\tilde{\mathbf{p}}\|^2 - \|\tilde{\mathbf{p}} + \hat{\mathbf{v}}\|^2| \\
&= \frac{1}{2\sigma^2}|\|\tilde{p'}_1 + \hat{\mathbf{v}}\|^2 - \|\tilde{p'}_1\|^2| \\
&= \frac{1}{2\sigma^2}|\|\tilde{p'}_1 + \mathbf{K}^{-1}\mathbf{v}\|^2 - \|\tilde{p'}_1\|^2| \\
&= \frac{1}{2\sigma^2}|\|\mathbf{K}^{-1}\mathbf{v}\|^2 + 2|\tilde{p'}_1 \mathbf{K}^{-1}\mathbf{v}|| \\
&\leq \frac{1}{2\sigma^2}|\|\mathbf{K}^{-1}\|_2^2 \Delta_{\mathcal{G}}^2 + 2|\tilde{p'}_1|\|\mathbf{K}^{-1}\|_2 \Delta_{\mathcal{G}}| \\
&= \frac{1}{2\sigma^2}|\hat{\Delta}_{\mathcal{G}}^2 + 2|p'_1|\hat{\Delta}_{\mathcal{G}}|,
\end{aligned}$$

where $\hat{\Delta}_{\mathcal{G}} = \|\mathbf{K}^{-1}\|_2 \Delta_{\mathcal{G}} = S_{max}(\mathbf{K}^{-1})\Delta_{\mathcal{G}} = \frac{\Delta_{\mathcal{G}}}{S_{min}(\mathbf{M})^{\frac{1}{2}}}$, and $S_{min}(\mathbf{M})$ is the minimum singular value of $\mathbf{M}$

Following the proof of Theorem A.1 in Dwork et al. (2014), we can show that the mechanism $\mathcal{M}$ is (ϵ, δ)-DP, with

$$\epsilon \in (0, 1], \sigma \geq \frac{\sqrt{2ln(1.25/\delta)}\hat{\Delta}_{\mathcal{G}}}{a\epsilon}.$$

Consequently, the theorem holds.

□

E PROOF OF LEMMA 1 (PERTURBATION TRANSFORMATION)

Proof. We leverage the Taylor expansion to rewrite $l(\mathbf{z}_{(i)}^{non}, \theta)$ at the data point $\mathbf{x}_{(i)}^{non}$ as follows,

$$\begin{aligned}
&l(\mathbf{z}_{(i)}^{non}, \theta) = l(\mathbf{x}_{(i)}^{non}, \theta) + \\
&(\mathbf{z}_{(i)}^{non} - \mathbf{x}_{(i)}^{non})^\intercal \nabla_{\mathbf{x}_{(i)^{non}}} l(\mathbf{x}_{(i)}^{non}, \theta) + o(\mathbf{z}_{(i)}^{non} - \mathbf{x}_{(i)}^{non})
\end{aligned} \tag{8}$$

Since the only mild constraint on $l(\mathbf{z}_{(i)}^{non}, \theta)$ is C-Lipschitz continuous, the higher order terms $o(\mathbf{z}_{(i)} - \mathbf{x}_{(i)})$ is non-negative for “non-negative cases”. Therefore, Equation 8 can be further approximated as,

$$l(\mathbf{z}_{(i)}^{non}, \theta) \geq l(\mathbf{x}_{(i)}^{non}, \theta) + (\mathbf{z}_{(i)}^{non} - \mathbf{x}_{(i)}^{non})^\intercal \nabla_{\mathbf{x}_{(i)}^{non}} l(\mathbf{x}_{(i)}^{non}, \theta), \tag{9}$$

which essentially tightens the privacy budget. Calculating the gradient of the above loss function, we can have:

$$\begin{aligned} &\nabla_\theta l(\mathbf{z}^{non}_{(i)}, \theta) \\ \geq &\nabla_\theta l(\mathbf{x}^{non}_{(i)}, \theta) + \nabla_\theta((\mathbf{z}^{non}_{(i)} - \mathbf{x}^{non}_{(i)})^\intercal \nabla_{\mathbf{x}^{non}_{(i)}} l(\mathbf{x}^{non}_{(i)}, \theta))x \\ = &\nabla_\theta l(\mathbf{x}^{non}_{(i)}, \theta) + (\mathbf{z}^{non}_{(i)} - \mathbf{x}^{non}_{(i)})^\intercal \mathbf{J}_\theta \nabla_{\mathbf{x}^{non}_{(i)}} l(\mathbf{x}^{non}_{(i)}, \theta), \end{aligned} \tag{10}$$

where $\mathbf{J}_\theta \nabla_{\mathbf{x}^{non}_{(i)}} l(\mathbf{x}^{non}_{(i)}, \theta)$ denotes the Jacobian matrix. We assume the gradient perturbation $\mathbf{p}_{(i)}$ is defined as $\mathbf{p}_{(i)} = \mathbf{J}_\theta \nabla_{\mathbf{x}^{non}_{(i)}} l(\mathbf{x}^{non}_{(i)}, \theta)\mathbf{b}_{(i)}$, which is transformed from the input perturbation $\mathbf{b}_{(i)}$. The gradient can thus be rewritten as Equation 3:

$$\nabla_\theta l(\mathbf{z}^{non}_{(i)}, \theta) = \nabla_\theta l(\mathbf{x}^{non}_{(i)}, \theta) + \mathbf{p}_{(i)}.$$

To analyze the statistics of $\mathbf{J}_\theta \nabla_{\mathbf{x}^{non}_{(i)}} l(\mathbf{x}^{non}_{(i)}, \theta)$, we vectorize $\mathbf{b}_{(i)}$, $\mathbf{z}^{non}_{(i)}$ and θ, and let $\mathbf{A}_{(i)} = \mathbf{J}_\theta \nabla_{\mathbf{x}^{non}_{(i)}} l(\mathbf{x}^{non}_{(i)}, \theta)$. Therefore, the problem becomes that given $\mathbf{p}_{(i)} = \mathbf{A}_{(i)}\mathbf{b}_{(i)}$, where $\mathbf{z}^{non}_{(i)}, \mathbf{b}_{(i)} \in \mathbb{R}^{v\times 1}$, $\mathbf{b}^{(k)}_{(i)} \sim \mathcal{N}(0, \sigma^2), \theta \in \mathbb{R}^{w\times 1}$, $\mathbf{A}_{(i)} \in \mathbb{R}^{v\times 1\times w}$, what is the scale of $\mathbf{p}_{(i)}$.

Denoting $\mathbf{M}_{(i)}$ as $\mathbf{A}_{(i)}\mathbf{A}^\intercal_{(i)}$, then according to the linear transformation of Gaussian random variable, we can conclude that $\mathbf{p}_{(i)} \sim \mathcal{N}(0, \mathbf{\Sigma}_{(i)})$, where $\mathbf{\Sigma}_{(i)}$ is the covariance matrix of $\mathbf{p}_{(i)}$ and $\mathbf{\Sigma}_{(i)} = \mathbf{M}_{(i)}\sigma^2$. ☐

F THEOREM 6 (HGM) AND PROOF

Theorem 6. *Heterogeneous Gaussian Mechanism. Let* $\mathcal{G} : \mathbb{R}^v \rightarrow \mathbb{R}^w$ *be an arbitrary* w*-dimensional function, and* $\Delta_\mathcal{G} = \max_{\mathcal{D},\mathcal{D}'} \|\mathcal{G}(\mathcal{D}) - \mathcal{G}(\mathcal{D}')\|_2$. *A Heterogeneous Gaussian Mechanism* $\mathcal{M}$ *with the diagonal matrix* $\mathbf{\Sigma} \in \mathbb{R}^{w\times w}$ *adds noise to each of the* w *elements of the output. The mechanism* $\mathcal{M}$ *is* (ϵ, δ)*-DP, with*

$$\epsilon \in (0, 1], \sigma_{min} \geq \sqrt{2ln(1.25/\delta)}\Delta_\mathcal{G}/\epsilon.$$

where $\mathbf{\Sigma}$ *is a diagonal matrix with each element being* $\sigma_1^2, \sigma_2^2, ..., \sigma_w^2$ *and* $\sigma_{min} = \min_{i\in\{1,2,...,w\}} \sigma_i$.

The proof is as follows:

Proof. The privacy loss of an output o is defined as:

$$\mathcal{L}(o; \mathcal{M}, \mathcal{D}, \mathcal{D}') = ln\frac{Pr[\mathcal{M}(\mathcal{D}, \mathcal{G}, \mathbf{\Sigma})] = o}{Pr[\mathcal{M}(\mathcal{D}', \mathcal{G}, \mathbf{\Sigma})] = o}$$

Given $\mathbf{\Sigma}$, we have

$$\begin{aligned} &|\mathcal{L}(\boldsymbol{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\ =&|ln\frac{Pr[\mathcal{M}(\mathcal{D}, \mathcal{G}, \mathbf{\Sigma})] = \boldsymbol{o}}{Pr[\mathcal{M}(\mathcal{D}', \mathcal{G}, \mathbf{\Sigma})] = \boldsymbol{o}}| \\ =&|ln\frac{Pr[\mathcal{G}(\mathcal{D}) + \mathcal{N}(0, \mathbf{\Sigma})] = \boldsymbol{o}}{Pr[\mathcal{G}(\mathcal{D}') + \mathcal{N}(0, \mathbf{\Sigma})] = \boldsymbol{o}}| \\ =&|ln\frac{exp(-\frac{1}{2}(\boldsymbol{o} - \mathcal{G}(\mathcal{D}))^\intercal \mathbf{\Sigma}^{-1}(\boldsymbol{o} - \mathcal{G}(\mathcal{D}))}{exp(-\frac{1}{2}(\boldsymbol{o} - \mathcal{G}(\mathcal{D}'))^\intercal \mathbf{\Sigma}^{-1}(\boldsymbol{o} - \mathcal{G}(\mathcal{D}'))}| \end{aligned}$$

According to the theorem, $\mathbf{\Sigma}$ is a diagonal matrix with each element being $\sigma_1^2, \sigma_2^2, ..., \sigma_w^2$. We can construct a diagonal matrix $\boldsymbol{K} \in \mathbb{R}^{w\times w}$, s.t., the diagonal elements of $\boldsymbol{K}$ are $\sigma_1, \sigma_2, ..., \sigma_w$, then

$$\begin{aligned} \boldsymbol{M} &= \boldsymbol{K}\boldsymbol{K}^\intercal, \\ \boldsymbol{K} &= \boldsymbol{K}^\intercal. \end{aligned}$$

Following the proof of Theorem 1, we can show that the mechanism $\mathcal{M}$ is (ϵ, δ)-DP, with

$$\epsilon \in (0, 1], 1 \geq \sqrt{2ln(1.25/\delta)}\hat{\Delta}_\mathcal{G}/\epsilon.$$

where $\hat{\Delta}_{\mathcal{G}} = \frac{\Delta_{\mathcal{G}}}{\sigma_{min}}$, and $\sigma_{min} = \min_{i \in \{1,2,\ldots,w\}} \sigma_i$. Consequently, the theorem holds.

Next, we will prove that this mechanism indeed describe the same fact as HGM in Phan et al. (2019) but from different views.

We define $\sigma^2 = \frac{1}{w}\sum_{i=1}^{w}\sigma_i^2$, and construct a vector $\boldsymbol{r} \in \mathbb{R}^w$ with each element being $r_i = \frac{\sigma_i^2}{\sum_{i=1}^{w}\sigma_i^2}$. Then we have $\sigma_i^2 = \sigma^2 \times wr_i$, and each element of the diagonal matrix $\boldsymbol{K}$ becomes $\sqrt{wr_1}\sigma, \sqrt{wr_2}\sigma, ..., \sqrt{wr_w}\sigma$.

Given $\boldsymbol{v} = \mathcal{G}(\mathcal{D}) - \mathcal{G}(\mathcal{D}')$ and v_i denotes i-th element of $\boldsymbol{v}$, we construct a vector $\hat{\boldsymbol{v}} = \sqrt{\sum_{i=1}^{w}\frac{1}{wr_i}v_i^2}$. Then we can show that,

$$\begin{aligned}
&|\mathcal{L}(o; \mathcal{M}, \mathcal{D}, \mathcal{D}')| \\
&= \frac{1}{2}|\|\boldsymbol{K}^{-1}\boldsymbol{v}\|^2 + 2|\tilde{p'}_1\boldsymbol{K}^{-1}\boldsymbol{v}|| \\
&= \frac{1}{2\sigma^2}|\|\hat{\boldsymbol{v}}\|^2 + 2|\tilde{p'}_1\hat{\boldsymbol{v}}|| \\
&\leq \frac{1}{2\sigma^2}|\hat{\Delta}_{\mathcal{G}}^2 + 2|p'_1|\hat{\Delta}_{\mathcal{G}}|,
\end{aligned}$$

where $\hat{\Delta}_{\mathcal{G}} = \max_{\mathcal{D},\mathcal{D}'}\sqrt{\|\hat{\boldsymbol{v}}\|^2}$ represents the new sensitivity.

Following the proof of Theorem 1, we can show the similar conclusion as in Phan et al. (2019) that this mechanism is (ϵ, δ)-DP, with

$$\epsilon \in (0, 1], \sigma \geq \sqrt{2ln(1.25/\delta)}\hat{\Delta}_{\mathcal{G}}/\epsilon.$$

$\square$

Compared with Phan et al. (2019), Theorem 6 gives different definitions on the sensitivity $\Delta_{\mathcal{G}}$ and perturbation scale σ. However, we have shown in the proof that these two mechanisms indeed describe the same fact from different views.

G PROOF OF LEMMA 2

In vanilla SGD, the algorithm picks one example at each iteration. Thus, the subscript t is eqivalent to (i). Lemma 2 can be easily derived from the following lemma:

Lemma 3. *Given perturbed example* $\mathbf{z}_t^{non} = \mathbf{x}_t^{non} + \mathbf{b}_t$ *with* $\mathbf{b}_t^{(k)} \sim \mathcal{N}(0, \sigma^2)$*, the number of training steps* T*, and* C*-Lipschitz continuous loss* l*. The gradient* $\nabla_{\theta_t} l(\mathbf{z}_t^{non}, \theta_t)$ *at each step of vanilla SGD can be reformulated as the gradient with respect to the original sample with a gradient perturbation:*

$$\nabla_{\theta_t} l(\mathbf{z}_t^{non}, \theta_t) \geq \nabla_{\theta_t} l(\mathbf{x}_t^{non}, \theta_t) + \mathbf{p}_t, \tag{11}$$

where $\mathbf{p}_t$ *is the transformed perturbation with* $\mathbf{p}_t \sim \mathcal{N}(0, \mathbf{\Sigma}_t)$*,* $\mathbf{\Sigma}_t = T\mathbf{M}_t\sigma^2$*,* $\mathbf{M}_t = \mathbf{A}_{(i)}\mathbf{A}_{(i)}^{\intercal}$*,* $\mathbf{A}_{(i)} = \mathbf{J}_{\theta_t}\nabla_{\mathbf{x}_{(i)}^{non}} l(\mathbf{x}_{(i)}^{non}, \theta_t)$

The T in Lemma 3 comes from the fact that compared to traditional gradient perturbation added at each iteration, the input perturbation is only added at the start of training for one single time. The variance is amplified by a coefficient T when input perturbation is transformed into the gradient perturbation.

Given two neighbouring datasets $\mathcal{D}$ and $\mathcal{D}'$, training data size N, the transformation matrix $\mathbf{M}_t$ and minimum singular value $S_{min}(\mathbf{M}_t)$ are calculated to analyze DP contribution from input perturbation. However, because the sample at each iteration is randomly picked and $S_{min}(\mathbf{M}_t)$ is data-dependent, it is challenging to bound the difference of the transformed gradient perturbation p between the optimization processes on these two datasets.

To solve this, we first claim that the minimum singular value of the transformation matrix, $S_{min}(\mathbf{M}_t)$ determines how much DP guarantee MGM can provide. According to Corollary 1, this DP guarantee is equivalent to the guarantee provided by a traditional Gaussian Mechanism with its perturbation following a Gaussian distribution $\mathcal{N}(0, TS_{min}(\mathbf{M}_t)\sigma^2)$. Therefore, Lemma 2 is derived.

H PERTURBATION TRANSFORMATION IN MINI-BATCH SGD

In mini-batch SGD, the algorithm randomly picks a batch of samples at each iteration and feeds them into the model for optimization. Given the initial parameters θ_0, iteration t, the parameters are updated as:

$$\theta_{t+1} = \theta_t - \eta\,\frac{1}{B}\sum_{i=1}^{B}(\nabla_\theta l(\theta_t, \mathbf{z}_t^{non}) + \nabla_\theta l(\theta_t, \mathbf{z}_t^{neg})), \tag{12}$$

where η denotes the learning rate, B denotes the batch size, $\mathbf{z}_t^{non}$ denotes the perturbed samples of "non-negative cases" and $\mathbf{z}_t^{neg}$ denotes the perturbed samples of "negative cases". Both $\mathbf{z}_t^{non}$ and $\mathbf{z}_t^{neg}$ are randomly picked at iteration t.

According to the definition of $\nabla_{\theta_t} l(\theta_t, \mathbf{z}_t^{non})$ in Equation (11), we have that

$$\frac{1}{B}\sum_{i=1}^{B}\nabla_{\theta_t} l(\theta_t, \mathbf{z}_t^{non}) \geq \frac{1}{B}\sum_{i=1}^{B}\nabla_\theta l(\mathbf{x}_t^{non}, \theta) + \mathbf{p}_t, \tag{13}$$

where $\mathbf{p}_t$ is the transformed perturbation with $\mathbf{p}_t \sim \mathcal{N}(0, \mathbf{\Sigma}_t)$, $\mathbf{\Sigma}_t = T\mathbf{M}_t\sigma^2$, $\mathbf{M}_t = \frac{1}{B^2}\sum_{i=1}^{B}\mathbf{A}_{(i)}\mathbf{A}_{(i)}^\intercal$, $\mathbf{A}_{(i)} = \mathbf{J}_{\mathbf{x}_{(i)}^{non}}\nabla_\theta l(\mathbf{x}_{(i)}^{non}, \theta)$.

This gradient is similar to that of vanilla SGD in Lemma 3, except that $\mathbf{M}_t$ is $\frac{1}{B^2}\sum_{i=1}^{B}\mathbf{A}_{(i)}\mathbf{A}_{(i)}^\intercal$ instead of $\mathbf{A}_{(i)}\mathbf{A}_{(i)}^\intercal$. We can directly follow the processes in vanilla SGD and derive Theorem 2 and Corollary 2 with $\mathbf{M}_t = \frac{1}{B^2}\sum_{i=1}^{B}\mathbf{A}_{(i)}\mathbf{A}_{(i)}^\intercal$.

According to Corollary 2, the DP guarantee that **MMGA** provides at iteration t for "non-negative cases" is equivalent to that provides by a **GM** with $\mathcal{N}(0, \xi_{up}^2)$.

On the other hand, we have the following for "negative cases":

$$\frac{1}{B}\sum_{i=1}^{B}\nabla_{\theta_t} l(\theta_t, \mathbf{z}_t^{neg}) = \frac{1}{B}\sum_{i=1}^{B}\nabla_\theta l(\mathbf{x}_t^{neg}, \theta) + \mathcal{N}(0, \xi_{up}^2). \tag{14}$$

Therefore, we can also claim that Corollary 2 is applicable to both "non-negative cases" and "negative cases" in mini-batch SGD.

I PROOF OF THEOREM 8 (MMGA WITH MA)

Given a multivariate Gaussian mechanism $\mathcal{M}$, two neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$, the output $\mathbf{o}$, we first define the log moments of the privacy loss random variable as follows,

$$m(\mathbf{o}; \mathcal{M}, \mathcal{D}, \mathcal{D}') \triangleq log\frac{Pr(\mathcal{M}(\mathcal{D}) \in \mathbf{o})}{Pr(\mathcal{M}(\mathcal{D}') \in \mathbf{o})}. \tag{15}$$

Then the log of the moment generating function at the value α is defined as follows,

$$\lambda_{\mathcal{M}}(\alpha; \mathcal{D}, \mathcal{D}') \triangleq log\,\mathbb{E}_{\mathbf{o}\sim\mathcal{M}(\mathcal{D})}(e^{\alpha m(\mathbf{o};\mathcal{M},\mathcal{D},\mathcal{D}')}). \tag{16}$$

We take the maximum over all possible neighboring pairs of $\mathcal{D}$ and $\mathcal{D}'$ to obtain the moments accountant:

$$\lambda_{\mathcal{M}}(\alpha) \triangleq \max_{\mathcal{D},\mathcal{D}'}\lambda_{\mathcal{M}}(\alpha; \mathcal{D}, \mathcal{D}'). \tag{17}$$

With the moments accountant $\lambda_{\mathcal{M}}(\alpha)$, we have the following lemma and theorem,

Theorem 7. *For any* $\epsilon > 0$, *the mechanism* $\mathcal{M}$ *is* (ϵ, δ)*-differentially private for*

$$\delta = \min_{\alpha} exp(\lambda_{\mathcal{M}}(\alpha) - \alpha\epsilon).$$

The proof of Theorem 7 is the same as the proof of Theorem 2.2 in (Abadi et al., 2016).

Lemma 4. *Given a function $f : \mathbb{R}^v \to \mathbb{R}^w$ with $\|f(.)\|_2 \geq 1$. Let $\xi_{up} \leq 1$, Ca be the gradient clipping coefficient and L be the sample from origin data with the sampling probability $q = \frac{B}{N} < \frac{1}{16\xi_{up}}$. Then for any positive integer $\alpha \leq \xi_{up}^2 ln\frac{1}{q\xi_{up}}$, the mechanism $\mathcal{M}(d) = \sum_{d_i \in L} f(d_i) + \mathcal{N}(0, \boldsymbol{\Sigma}_{(i)}\sigma^2)$ satisfies*

$$\lambda_{\mathcal{M}}(\alpha) \leq \frac{q^2\alpha(\alpha+1)}{(1-q)\xi_{up}^2} + \mathcal{O}(\frac{q^3\alpha^3}{\xi_{up}^3})$$

The detailed proof of the lemma 4 is in Appendix J.

Theorem 8. *There exist constants c_1 and c_2 so that given the sampling probability $q = \frac{B}{N}$ and the number of training steps T, for any $\epsilon < c_1 q^2$, Algorithm 1 is (ϵ, δ)-differential private for any $\delta > 0$ if*

$$\xi_{up} \geq c_2 \frac{q\sqrt{Tlog(1/\delta)}}{\epsilon} \tag{18}$$

With Lemma 4, the proof is similar as the proof of Theorem 1 in Abadi et al. (2016).

J PROOF OF LEMMA 4 (MA)

Proof. Given a function $f : \mathbb{R}^v \to \mathbb{R}^w$, we fix $\mathcal{D}$ and let $\mathcal{D} = \mathcal{D}' \cup \boldsymbol{d}_{(n)}$. Without loss of generality, let $\|f(\boldsymbol{d}_{(n)})\| = 1$ and $\sum_{i \in B \setminus [n]} f(\boldsymbol{d}_{(i)}) = 0$, and let ζ_0 denotes the pdf of $\mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$, ζ_1 denotes the pdf of $\mathcal{N}(\boldsymbol{d}_{(n)}, \boldsymbol{\Sigma})$, then we can have:

$$\begin{aligned}
&\mathcal{M}(\mathcal{D}') \sim \zeta_0, \\
&\mathcal{M}(\mathcal{D}) \sim \zeta \triangleq (1-q)\zeta_0 + q\zeta_1.
\end{aligned}$$

We want to show that

$$\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^{\alpha}] \leq \lambda, \tag{19}$$

$$\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_0(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^{\alpha}] \leq \lambda, \tag{20}$$

for certain λ.

For the Equation (19), we have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^{\alpha}] \\
=&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^{\alpha+1}] \\
=&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[1 + (\frac{\zeta(\boldsymbol{s}) - \zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^{\alpha+1}] \\
=&\sum_{i=0}^{\alpha+1}\binom{\alpha+1}{i}\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta(\boldsymbol{s}) - \zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^{i}].
\end{aligned} \tag{21}$$

We find that the first term of (21) is 1, and the second term is 0 since

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[\frac{\zeta(\boldsymbol{s}) - \zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})}] &= \int_{\infty}^{-\infty} \zeta_0(\boldsymbol{s})\frac{\zeta(\boldsymbol{s}) - \zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})}\,\mathrm{d}\boldsymbol{s} \\
&= \int_{\infty}^{-\infty} \zeta(\boldsymbol{s}) - \zeta_0(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s} \\
&= \int_{\infty}^{-\infty} \zeta(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s} - \int_{\infty}^{-\infty} \zeta_0(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s} \\
&= 0
\end{aligned}$$

The third term of (21) is:

$$\begin{aligned}&\binom{\alpha+1}{2}\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\=&\binom{\alpha+1}{2}q^2\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\end{aligned}$$

Leveraging the fact that for any $\boldsymbol{a}\in\mathbb{R}^v$, $\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[exp(\boldsymbol{a}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{s})]=exp(\boldsymbol{a}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{a}/2)$, then:

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\=&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})}-1)^2]\\=&1-2\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})]+\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\end{aligned}\tag{22}$$

Given that $\zeta_0\triangleq\mathcal{N}(\boldsymbol{0},\boldsymbol{\Sigma})$, $\zeta_1\triangleq\mathcal{N}(\boldsymbol{d}_{(n)},\boldsymbol{\Sigma})$, we have

$$\begin{aligned}\frac{\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})}=&exp(-\frac{1}{2}((\boldsymbol{s}-\boldsymbol{d}_{(n)})^\intercal\boldsymbol{\Sigma}^{-1}(\boldsymbol{s}-\boldsymbol{d}_{(n)})-\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{s}))\\=&exp(-\frac{1}{2}(\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{s}-2\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)}+\\&\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)}-\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{s}))\\=&exp(-\frac{1}{2}(\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)}-2\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)}))\\=&exp(-\frac{1}{2}\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\times exp(\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\end{aligned}$$

Then (22) becomes:

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\=&1-2\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})]+\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\=&1-2\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[exp(-\frac{1}{2}\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\times exp(\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})]+\\&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[exp(-\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\times exp(\boldsymbol{s}^\intercal\boldsymbol{\Sigma}^{-1}(2\boldsymbol{d}_{(n)}))]\\=&1-2exp(-\frac{1}{2}\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\times exp(\frac{1}{2}\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})+\\&exp(-\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\times exp(2\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\end{aligned}\tag{23}$$

$$\begin{aligned}=&1-2+exp(\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})\\=&exp(\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})-1\end{aligned}\tag{24}$$

Therefore, the third term of (21) is:

$$\begin{aligned}&\binom{\alpha+1}{2}\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\=&\binom{\alpha+1}{2}q^2\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_1(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\=&\binom{\alpha+1}{2}q^2(exp(\boldsymbol{d}_{(n)}^\intercal\boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)})-1)\end{aligned}\tag{25}$$

To bound (25), we need to determine that

$$\begin{aligned}
&\max_{\boldsymbol{d}_{(n)}\in\mathbb{R}^v} \boldsymbol{d}_{(n)}^\intercal \boldsymbol{\Sigma}^{-1}\boldsymbol{d}_{(n)}\\
=&\max_{\boldsymbol{d}_{(n)}\in\mathbb{R}^v} \boldsymbol{d}_{(n)}^\intercal \boldsymbol{\Sigma}^{-\frac{1}{2}}\boldsymbol{\Sigma}^{-\frac{1}{2}}\boldsymbol{d}_{(n)}\\
=&\max_{\boldsymbol{d}_{(n)}\in\mathbb{R}^v} (\frac{\|\boldsymbol{d}_{(n)}^\intercal \boldsymbol{\Sigma}^{-\frac{1}{2}}\|_2}{\|\boldsymbol{d}_{(n)}\|_2})^2\\
=&S_{max}(\boldsymbol{\Sigma}^{-\frac{1}{2}})^2\\
=&\frac{1}{S_{min}(\boldsymbol{\Sigma})},
\end{aligned} \tag{26}$$

Therefore, the third term of (21)

$$\begin{aligned}
&\binom{\alpha+1}{2}\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta(\boldsymbol{s})-\zeta_0(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]\\
\le&\binom{\alpha+1}{2}q^2(exp(\frac{1}{S_{min}(\boldsymbol{\Sigma})})-1)\\
\le&\binom{\alpha+1}{2}q^2\frac{2}{S_{min}(\boldsymbol{\Sigma})}\\
=&\frac{q^2(\alpha+1)\alpha}{S_{min}(\boldsymbol{\Sigma})}
\end{aligned} \tag{27}$$

For the Equation (20), we have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_0(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^\alpha]\\
=&\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta_0(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^{\alpha+1}]\\
=&\mathbb{E}_{\boldsymbol{s}\sim\zeta}[1+(\frac{\zeta_0(\boldsymbol{s})-\zeta(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^{\alpha+1}]\\
=&\sum_{i=0}^{\alpha+1}\binom{\alpha+1}{i}\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta_0(\boldsymbol{s})-\zeta(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^i].
\end{aligned} \tag{28}$$

Similar to the analysis of (19), the first term of (28) is 1, and the second term is 0. The third term is slightly different:

$$\begin{aligned}
&\binom{\alpha+1}{2}\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta_0(\boldsymbol{s})-\zeta(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^2]\\
=&\binom{\alpha+1}{2}q^2\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta_0(\boldsymbol{s})-\zeta_1(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^2]
\end{aligned}$$

Because $\zeta \triangleq (1-q)\zeta_0 + q\zeta_1$, we can get that $\zeta(\boldsymbol{s}) \ge (1-q)\zeta_0(\boldsymbol{s})$, and rewrite the above equation as:

$$\begin{aligned}
&\binom{\alpha+1}{2}\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta_0(\boldsymbol{s})-\zeta(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^2]\\
\le&\binom{\alpha+1}{2}\frac{q^2}{1-q}\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_0(\boldsymbol{s})-\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]
\end{aligned}$$

The following analysis is similar to that of (19), and we obtain the third term of (28)

$$\binom{\alpha+1}{2}\mathbb{E}_{\boldsymbol{s}\sim\zeta}[(\frac{\zeta_0(\boldsymbol{s})-\zeta(\boldsymbol{s})}{\zeta(\boldsymbol{s})})^2] \tag{29}$$

$$\leq\binom{\alpha+1}{2}\frac{q^2}{1-q}\mathbb{E}_{\boldsymbol{s}\sim\zeta_0}[(\frac{\zeta_0(\boldsymbol{s})-\zeta_1(\boldsymbol{s})}{\zeta_0(\boldsymbol{s})})^2]$$

$$\leq\frac{q^2(\alpha+1)\alpha}{(1-q)S_{min}(\boldsymbol{\Sigma})} \tag{30}$$

Compared with the proof of Lemma 3 in Abadi et al. (2016), it's obvious that the only difference between Gaussian mechanism and multivariate Gaussian mechanism is the noise scale. The covariance $\boldsymbol{\Sigma}$ of MGM can be decomposed into singular vectors and singular values, and the minimum singular value $S_{min}(\boldsymbol{\Sigma})$ actually solely determines the upper bound of the moment accountant $\lambda_{\mathcal{M}}(\alpha)$, which is the counterpart of $C\sigma^2$ of GM with C being the gradient clipping norm.

Following the proof of Lemma 3 in Abadi et al. (2016), and given $\boldsymbol{\Sigma} \triangleq \frac{S_{min}(\boldsymbol{M})\sigma^2}{C^2}$, we can get Lemma 4:

Given a function $f:\mathbb{R}^v\rightarrow\mathbb{R}^w$ with $\|f(.)\|_2\geq 1$. Let $\frac{S_{min}(\boldsymbol{M})\sigma^2}{C^2}\leq 1$, C be the gradient clipping norm and L be the sample from origin data with the sampling probability $q<\frac{1}{16\sqrt{\frac{S_{min}(\boldsymbol{M})\sigma^2}{C^2}}}$. Then for any positive integer $\alpha\leq\frac{S_{min}(\boldsymbol{M})\sigma^2}{C^2}ln\frac{1}{q\sqrt{\frac{S_{min}(\boldsymbol{M})\sigma^2}{C^2}}}$, the mechanism $\mathcal{M}(d)=\sum_{d_i\in L}f(d_i)+\mathcal{N}(0,\boldsymbol{\Sigma}\sigma^2)$ satisfies

$$\lambda_{\mathcal{M}}(\alpha)\leq\frac{q^2\alpha(\alpha+1)}{(1-q)\frac{S_{min}(\boldsymbol{M})\sigma^2}{C^2}}+\mathcal{O}(\frac{q^3\alpha^3C^3}{S_{min}(\boldsymbol{M})^{\frac{3}{2}}\sigma^3})$$

□

K THE IMPLEMENTATION OF C-LIPSCHITZ

In practice, the gradient $\nabla_\theta l(\mathbf{z}_{(i)},\theta)$ is always clipped by a norm threshold C:

$$Clipped(\nabla_\theta l(\mathbf{z}_{(i)},\theta))\triangleq\frac{\nabla_\theta l(\mathbf{z}_{(i)},\theta)}{\max(1,\|\nabla_\theta l(\mathbf{z}_{(i)},\theta)\|_2/C)} \tag{31}$$

Lemma 5. *Given perturbed example* $\mathbf{z}_{(i)}=\mathbf{x}_{(i)}+\mathbf{b}_{(i)}$ *with* $\mathbf{b}_{(i)}^{(k)}\sim\mathcal{N}(0,\sigma^2)$*, and denoting a clipping coefficient* $a=1/\max(1,\|\nabla_\theta l(\mathbf{z}_{(i)},\theta)\|_2/C)$*. The gradient* $\nabla_\theta l(\mathbf{z}_{(i)},\theta)$ *clipped by a norm threshold* C *can be reformulated as the gradient with respect to the original sample with a gradient perturbation:*

$$Clipped(\nabla_\theta l(\mathbf{z}_{(i)},\theta))=a\nabla_\theta l(\mathbf{x}_{(i)},\theta)+\mathbf{p}_{(i)}, \tag{32}$$

where $\mathbf{p}_{(i)}$ *is the transformed perturbation with* $\mathbf{p}_{(i)}\sim\mathcal{N}(0,\boldsymbol{\Sigma}_{(i)})$*,* $\boldsymbol{\Sigma}_{(i)}=\mathbf{M}_{(i)}\sigma^2$*,* $\mathbf{M}_{(i)}=a^2\mathbf{A}_{(i)}\mathbf{A}_{(i)}^\intercal$ *and* $\mathbf{A}_{(i)}=\nabla_{\mathbf{x}_{(i)}}\nabla_\theta l(\mathbf{x}_{(i)},\theta)$*.*

L PARAMETERS SLICING

The AE can have a complex structure in order to remove the perturbation and reconstruct the input samples effectively. This complex structure results in a high-dimensional parameter space, and leads to an inefficient perturbation transformation. According to the definition of $\boldsymbol{A}_{(i)}$ and $\boldsymbol{M}$, when the dimension of $\boldsymbol{\theta}$ is high, the calculation will incur a significant computation cost.

A solution we employ is to slice the parameter $\boldsymbol{\theta}\in\mathbb{R}^w$ into several parts, e.g., $\boldsymbol{\theta}_1\in\mathbb{R}^{w_1}$, $\boldsymbol{\theta}_2\in\mathbb{R}^{w_2}$ and $\boldsymbol{\theta}_3\in\mathbb{R}^{w_3}$, corresponding to the first, intermediate, and last layers, respectively, as shown in

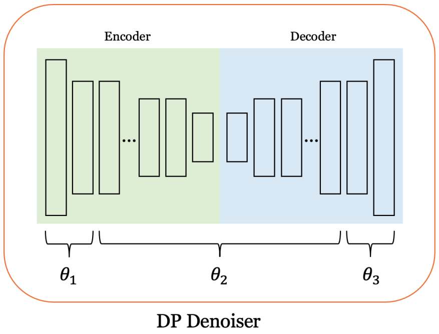


Figure 5: Denoiser with sliced parameters

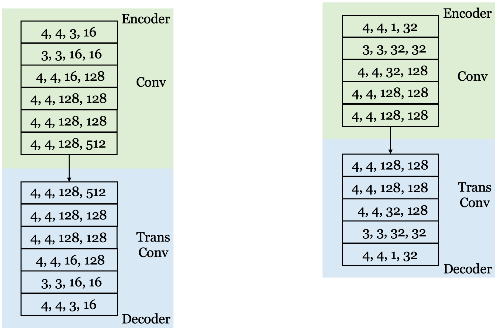
Figure 5. The input perturbation transformation is then applied on $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ and $\boldsymbol{\theta}_3$ separately with each sub-parameter slice having smaller size.

In the training stage, different optimizers work on different parameter slices and update them independently. Because the optimization is separated and each one can be regarded as an independent process, the perturbation transformation in Section **??**, MGM in Section **??** and MMGA in Section 2.3 can be applied to each process respectively. Without loss of generality, we use $\boldsymbol{\theta}$ directly in the following subsections and sections.

M DETAILED EXPERIMENTS

Pre-trained classifiers. Pre-trained classifiers are trained on public datasets. The pre-trained classifier for MNIST is a customized 12 layers deep convolutional network with residual blocks, which achieves 99.3% accuracy on test dataset. The pre-trained classifier for CIFAR-10 is a deep convolutional network transfered from VGG16 (Simonyan & Zisserman, 2014). We replace the top layers of VGG16 with customized fully connected layers and initialize all bottom layers with VGG16 parameters. This classifier achieves 86% accuracy on test dataset. Both classifiers are treated as public in all experiments.

Denoiser. We use convolutional and transposed convolutional layers to build the autoencoder based denoiser for both MNIST and CIFAR10 datasets. Layer normalization (Ba et al., 2016), residual (He et al., 2016) and skip structures are applied to avoid gradient vanishment. Drop out technique is applied to mitigate overfitting. The details of these two denoisers are shown in Figure 6a and Figure 6b.



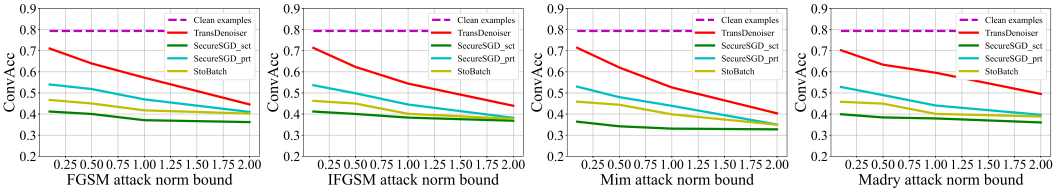
(a) Detailed architecture of denoiser for CIFAR10

(b) Detailed architecture of denoiser for MNIST

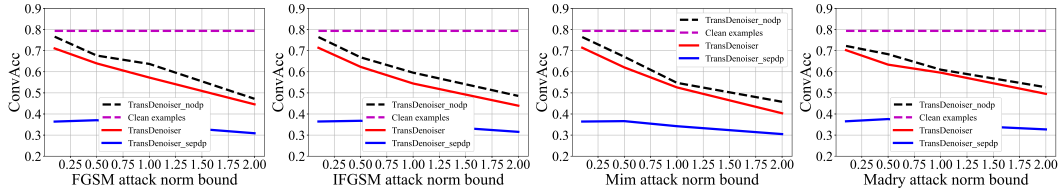
Figure 6: Denoiser for two datasets, where each box denotes the layer, the digits in each box denotes the shape of convolutional filter, "Conv" denotes and "Trans Conv" denotes the boxes are convolutional and transposed convolutional layers respectively.

Implementation details. All models are implemented using Tensorflow 1.14 and trained with a system equipped with Nvidia V-100 GPU. We use the open source code of SecureSGD (`https://github.com/haiphanNJIT/SecureSGD`) and StoBatch (`https://github.com/haiphanNJIT/StoBatch`) to conduct experiments, and we fix an error in SecureSGD code where variance is used incorrectly as standard deviation. Our code is available at `https://anonymous.4open.science/r/14d3ec7c-ab5b-4c76-be92-1f3b0454563a/`.

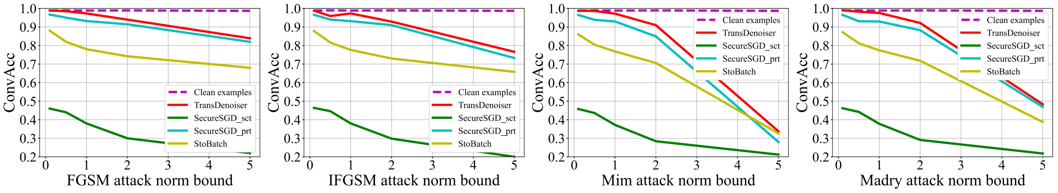
Empirical defense. We show ConvAcc against four different adversarial attacks: FGSM, I-FGSM, Momentum Iterative Method (MIM) and MadryEtAl. As can be seen in Figure 7, the results for MIM and MadryEtAl are similar to those shown in Section 3 (Figure 3). TransDenoiserdominates all baselines and ablation cases over four different attacks and different attack norm bound.



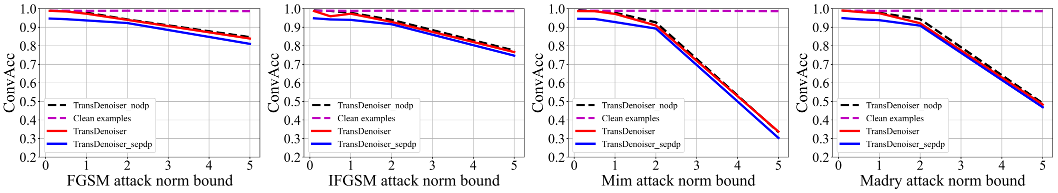
(a) TransDenoiser and baselines on CIFAR10



(b) TransDenoiser and ablation cases on CIFAR10



(c) TransDenoiser and baselines on MNIST



(d) TransDenoiser and ablation cases on MNIST

Figure 7: More comparison among TransDenoiser, baselines and ablation cases for conventional accuracy vs. l_2 radii on two datasets. The input perturbation scale on CIFAR10 = 0.1, on MNIST = 0.25, the overall gradient perturbation scale = 2.0 ($\geq$ 2.0 for TransDenoiser), and guarantee $(1.0, 1e{-}5)$-DP for private models.