In-distribution Public Data Synthesis with Diffusion Models for Differentially Private Image Classification

Supplementary Material

A. Proofs

A.1. Proof of Theorem 3.1.

Following [12], we address this theorem with a finite number of samples $S_{data} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$. As mentioned in Appendix B.3 of [12], p_{data} of 'entire' data and p_s of 'seen' data can be represented by mixtures of Dirac delta distributions:

$$p_{data}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N}\delta(\boldsymbol{x}-\boldsymbol{x}_i),$$

and

$$p_s(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n}\delta(\boldsymbol{x}-\boldsymbol{x}_i),$$

where $\boldsymbol{x} \in \mathcal{X}$ which is discrete sample space, since any pixel of an image can be represented by int between 0 to 255. p_θ can be defined in $\mathcal{X}$, which contains S_{data}. We defined the measure based on the KL divergence without $q(\boldsymbol{x}) = 0$ for the discrete probability, similar to [19], as follows:

$$\tilde{D}_{KL}(p\|q) = \sum_{\substack{\boldsymbol{x}\in\mathcal{X},\\ q(\boldsymbol{x})\neq 0}} p(\boldsymbol{x})\log\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}.$$

$\tilde{D}_{KL}$ becomes the KL divergence when the support of p is included in the support of q [9]. Then, by the definition:

$$\begin{aligned}
\tilde{D}_{KL}(p_\theta\|p_{data}) &= \sum_{\boldsymbol{x}\in S_{data}}\left(p_\theta(\boldsymbol{x})\log\frac{p_\theta(\boldsymbol{x})}{p_{data}(\boldsymbol{x})}\right)\\
&= \sum_{\boldsymbol{x}\in S_s}\left(p_\theta(\boldsymbol{x})\left[\log\frac{p_\theta(\boldsymbol{x})}{p_s(\boldsymbol{x})}+\log\frac{p_s(\boldsymbol{x})}{p_{data}(\boldsymbol{x})}\right]\right)\\
&\quad+\sum_{\boldsymbol{x}\in S_u}\left(p_\theta(\boldsymbol{x})\log\frac{p_\theta(\boldsymbol{x})}{p_{data}(\boldsymbol{x})}\right)\\
&= \tilde{D}_{KL}(p_\theta\|p_s)+\log\frac{N}{n}\sum_{\boldsymbol{x}\in S_s}p_\theta(\boldsymbol{x})\\
&\quad+\sum_{\boldsymbol{x}\in S_u}\left(p_\theta(\boldsymbol{x})\log\frac{p_\theta(\boldsymbol{x})}{p_{data}(\boldsymbol{x})}\right)\\
&= \tilde{D}_{KL}(p_\theta\|p_s)+\log\frac{N}{n}(1-\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x}))\\
&\quad+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log(Np_\theta(\boldsymbol{x}))\\
&= \tilde{D}_{KL}(p_\theta\|p_s)+\log\frac{N}{n}+\log n\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\\
&\quad+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log p_\theta(\boldsymbol{x}).
\end{aligned}$$

Note that as $N \to |\mathcal{X}|$, i.e., $S_{data} \to \mathcal{X}$, the support of p_{data} contains the support of p_θ so that $\tilde{D}_{KL}$ becomes the same as the (reverse) KL divergence. Since $x\log(x)$ is convex, we can use Jensen's inequality. For convenience, let $\bar{p}^u_\theta := \frac{1}{N-n}\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x}) = \frac{1}{N-n}\hat{p}^u_\theta \le \frac{1}{N-n}$. Then,

$$\begin{aligned}
&\tilde{D}_{KL}(p_\theta\|p_{data})-\tilde{D}_{KL}(p_\theta\|p_s)\\
&\quad= \log\frac{N}{n}+\log n\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log p_\theta(\boldsymbol{x})\\
&\quad= \log\frac{N}{n}+(N-n)\bar{p}^u_\theta\log n+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log p_\theta(\boldsymbol{x})\\
&\quad\ge \log\frac{N}{n}+(N-n)\bar{p}^u_\theta\log n+(N-n)\bar{p}^u_\theta\log\bar{p}^u_\theta\\
&\quad= \log\frac{N}{n}+(N-n)\bar{p}^u_\theta\log n\bar{p}^u_\theta\\
&\quad= \log\frac{N}{n}+\hat{p}^u_\theta\log\frac{n}{N-n}\hat{p}^u_\theta.
\end{aligned}$$

For small $n \le \frac{N}{e+1}$, $\bar{p}^u_\theta\log n\bar{p}^u_\theta$ is a decreasing function in $0 \le \bar{p}^u_\theta \le \frac{1}{N-n}$. Therefore, the lower bound is a decreasing function with respect to $\bar{p}^u_\theta$. Meanwhile,

$$\begin{aligned}
&\log\frac{N}{n}+\log n\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log p_\theta(\boldsymbol{x})\\
&\quad= \log\frac{N}{n}+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log np_\theta(\boldsymbol{x})\\
&\quad\le \log\frac{N}{n}+\sum_{\boldsymbol{x}\in S_u}p_\theta(\boldsymbol{x})\log\frac{n}{N-n}\\
&\quad= \log\frac{N}{n}+(N-n)\bar{p}^u_\theta\log\frac{n}{N-n}\\
&\quad= \log\frac{N}{n}+\hat{p}^u_\theta\log\frac{n}{N-n}.
\end{aligned}$$

For small $n < \frac{N}{2}$, $\log\frac{n}{N-n} < 0$, and the upper bound is a decreasing function with respect to $\bar{p}^u_\theta$. Note that as $n \to N$, $p_{pub} \to p_{data}$, and the upper bound and the lower bound achieve equality.

B. Experimental settings

For experiments, we follow the learning framework outlined in Algorithm 1.

Algorithm 1: Private training with public data

Input: Private dataset $\boldsymbol{D}^{pr}$, public dataset $\boldsymbol{D}^{pub}$; Initial classifier $\boldsymbol{w}$, diffusion model $\boldsymbol{\theta}$, (discriminator ψ); Synthetic dataset size k.
Output: Final private classifier $\boldsymbol{w}_T$.
(Phase I) Synthesis and warm-up training
Train diffusion with public $\boldsymbol{\theta}^* \leftarrow (\boldsymbol{\theta}; \boldsymbol{D}^{pub})$.
if *Generation diversity* **then**
  Train discriminator $\psi^* \leftarrow (\psi; \boldsymbol{\theta}^*, \boldsymbol{D}^{pub})$.
Make synthetic public set $\boldsymbol{D}^{syn} \leftarrow (\boldsymbol{\theta}^*, \psi^*, k)$.
Augmentation diversity
Warm-up classifier $\boldsymbol{w}_0 \leftarrow (\boldsymbol{w}; \boldsymbol{D}^{syn}, aug, opt)$.
(Phase II) Private training
for $t = 0, 1, \ldots, T-1$ **do**
  if *Extended optimization* **then**
    Run extended methods f_g (e.g., Eq. 3)
      $\boldsymbol{w}_{t+1} \leftarrow (\boldsymbol{w}_t; \boldsymbol{D}^{pr}, \boldsymbol{D}^{syn})$.
  else DP-SGD (Eq. 2) $\boldsymbol{w}_{t+1} \leftarrow (\boldsymbol{w}_t; \boldsymbol{D}^{pr})$;
end

B.1. Classification

Private training For private learning, we adopt all the techniques of [4] with WRN-16-4. We employ the techniques such as *augmentation multiplicity* to minimize the averaged loss of various augmentations $\mathcal{L}_i(\boldsymbol{w}) = \mathbb{E}_k[\ell(\boldsymbol{w}, \texttt{aug}^k(\boldsymbol{x}_i))]$, weight standardization [16], and Exponential Moving Average (EMA). We re-implement the JAX official code of [4] in https://github.com/google-deepmind/jax_privacy and extended methods using Pytorch Opacus [22] libraries.

We present the experimental details for CIFAR-10 in Table 12 and CIFAR-100 in Table 13 with their search spaces and best hyperparameter values. All experiments are conducted with DP-SGD with momentum 0 unless otherwise specified. As private learning is hugely affected by the hyperparameter settings, we use a different search space for cold and warm settings. We calculated the noise level σ for training with the hyperparameters in Tables 12 and 13 using Opacus libraries.

Warm-up training We present the experimental details of the warm-up phase with SGD (not DP-SGD) in Table 14. We use SGD with momentum 0.9 as a default setting for warm-up training. For the warm setting, we train epochs until convergence since the number of training data of 2K is less than the warmSyn of 50K samples.

Pre-trained model We use pre-trained Vision Transformers such as DeiT, and CrossViT, with the ghost clipping methods proposed in [3] and their GitHub code from https://github.com/woodyx218/private_vision. For the cold setting, we trained the models with 5 epochs with Adam optimizer with a learning rate of 0.002. We used a batch size of 1K following the default settings in GitHub. We tested various ranges of model sizes. For the warm settings, we took a grid search on the learning rate of $\{0.0005, 0.001, 0.002\}$ on both the warm-up phase and the private training phase.

B.2. Diffusion synthesis

EDM settings We implemented EDM [12] from their official GitHub code from https://github.com/NVlabs/edm. We trained the EDM model using the base settings as reported on the official GitHub repository. For the CIFAR-10 dataset, we utilized a batch size of 512 images, distributed among four NVIDIA GeForce RTX 3090 GPUs, while maintaining the other setting as the default setting. Specifically, we use a learning rate of 10^{-3}, an EMA coefficient of 0.5, duration of 200. The detailed settings are reported in Table 7 of [12]. We sampled images with $\sigma_{min} = 0.002, \sigma_{max} = 80, \rho = 7, S_{churn} = 0, S_{min} = 0, S_{max} = \infty, S_{noise} = 1$, and a step size of 18, as the default setting. For the CIFAR-100 dataset, we employed a batch size of 1024 images, distributed among four NVIDIA A100 GPUs, while maintaining the same settings as those used for the CIFAR-10 dataset. The training took about 3 days with four NVIDIA GeForce RTX 3090 GPUs. Sampling with a step size of 18 and a batch size of 500, took less than 30 seconds per batch when using a single NVIDIA GeForce RTX 3090 GPU. Therefore, it took about one hour to sample 50K images.

In summary, we require about 10-15 times for training diffusion and 0.2-0.4 times for sampling 50K images compared to training our classifiers of [4].

DG settings We implemented DG [13] from their official GitHub code from https://github.com/alsdudrla10/DG. We trained both the classifier and discriminator for DG from scratch with synthetic data from the 2K public data. We avoided using pre-trained models from the CIFAR or Imagenet datasets to solely investigate the effects by using in-distribution data.

C. Additional Notes

C.1. DP-SGD

The noise level σ of DP-SGD is determined by the total steps, sampling probability, and privacy budget (ε, δ) as follows:

Table 12. Hyperparameters for CIFAR-10.

Setup	Hyper-parameter	Search space	Best values				
Warm	ε	$\{1, 2, 3, 4, 6\}$	1	2	3	4	6
	δ	$\{10^{-5}\}$	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}
	Multiplicity K	$\{16\}$	16	16	16	16	16
	Batch size	$\{4096\}$	4096	4096	4096	4096	4096
	Clipping norm C	$\{1\}$	1	1	1	1	1
	Epochs	$\{15, 20, 30, 40\}$	15	30	30	30	20
	Learning rate η	$\{0.1, 0.5, 1, 2, 4\}$	0.5	0.5	1	1	2
+ Extended	Public batch size	$\{32, 64, 128\}$	64	64	64	64	64

Table 13. Hyperparameters for CIFAR-100.

Setup	Hyper-parameter	Search space	Best values			
Warm	ε	$\{1, 2, 6, 10\}$	1	2	6	10
	δ	$\{10^{-5}\}$	10^{-5}	10^{-5}	10^{-5}	10^{-5}
	Multiplicity K	$\{16\}$	16	16	16	16
	Batch size	$\{4096\}$	4096	4096	4096	4096
	Clipping norm C	$\{1\}$	1	1	1	1
	Epochs	$\{25, 50, 75\}$	25	50	75	75
	Learning rate η	$\{0.1, 0.5, 1, 2, 4\}$	0.5	0.5	1	1
+ Extended	Public batch size	$\{32, 64, 128\}$	64	64	64	64

Table 14. Hyperparameters for warm-up phase on CIFAR-10 and CIFAR-100.

Hyper-parameter	Search space	CIFAR-10	CIFAR-100
Batch size	$\{64, 100, 128, 256\}$	64	64
Epochs	$\{50, 100, 200\}$	100	200
Learning rate η	$\{0.05, 0.1, 0.2, 0.3\}$	0.1	0.1
Momentum	$\{0.9\}$	0.9	0.9
Learning rate decay	{Cosine}	Cosine	Cosine
Weight decay	$\{5 \times 10^{-4}\}$	5×10^{-4}	5×10^{-4}
Radius ρ for SAM	$\{0.05, 0.1, 0.2\}$	0.1	0.1

Proposition C.1 *(Abadi et al. [1]). There exist constant c_1 and c_2 so that given total steps T and sampling probability q, for any $\varepsilon < c_1 q^2 T$, DP-SGD guarantees (ε, δ)-DP, for any $\delta > 0$ if we choose*

$$\sigma \geq c_2 \frac{q\sqrt{T \log(1/\delta)}}{\varepsilon}. \tag{4}$$

C.2. Deep Analysis for Toy Example in Section 3.2

For the spiral dataset, the radius increases proportionally with the angle and at each location, the points have a probability distribution that decreases proportionally with the cumulative sum. In other words, points closer to the origin have higher probabilities. With this dataset, we construct a simple diffusion model with time step 20 and two diffusion blocks, containing a linear layer with 64 units. We train the model 10K epochs with a learning rate of 0.001 using Adam optimizer. For classifiers, we train 2-layer neural networks with ReLU, optimized using BCELoss and a learning rate of 0.1 with SGD until convergence.

With the same examples of the spiral dataset, we plot the probability density functions along the x-axis in Figure 6, with 100% (sufficient), and 10% and 4% (insufficient) cases. Specifically, the diffusion model effectively approximated the distribution, regardless of the number of training data in region A. However, in region C, where the true density is high, the generated distribution shows a higher prob-

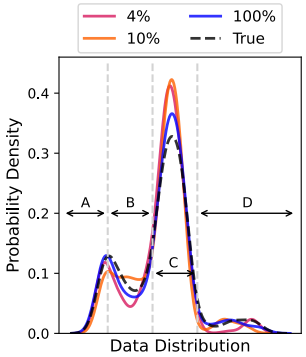


Figure 6. Probability density of toy experiment.

ability when the number of training data is limited. This indicates that when the number of training data is small, the model tends to memorize the training dataset. Conversely, in regions B and D, where the true density is low, the probability distribution of generated samples is lower than the true distribution, even if the true density is low. This indicates that when the number of samples is small, the model tends to ignore the tail distribution. As the number of in-distribution data is small, we should be aware of using synthetic data in terms of data memorization and ignore the tail part.

GAN mode collapse The diversity problem is more significant in other generative models, such as generative adversarial networks (GAN) [7]. As GAN can generative images with high fidelity, however, GAN suffers from training instability to control both generator and discriminatory, sometimes generating only a small portion of data repeatedly called mode collapse. The mode collapse happens when the convergence speed of the discrimination is faster than that of the generator, which induces the generator to generate the same images which can confuse the discriminator without generating similar images to real data. Figure 7 illustrates that the mode collapse happens with a smaller number of samples on Ring data. Thus, we mainly focus on diffusion models in this paper.

C.3. Notes on Discriminator Guidance [13]

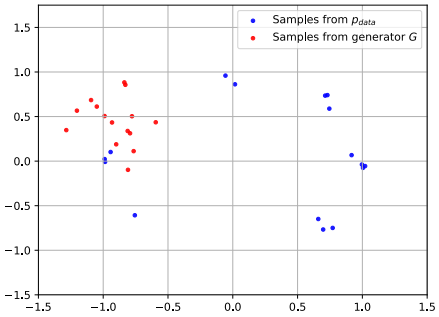
We use Discriminator Guidance (DG) [13] as a tool to understand the important factor of diffusion for performance gain without privacy concerns. By adjusting the weight of DG w_d, we determined that diversity is the key factor for sparse data. Previously, to control the trade-off between fidelity and diversity in diffusion models, Dhariwal and Nichol [5] suggested using the classifier to diffusion networks. The classifier, which is trained on noisy images during diffusion steps and their labels, can force the model to generate certain types of images based on their labels. To push further, DG [13] adopted another network called discriminator. The discriminator is trained to decide whether the images during the diffusion process are generated from real data or not. Thus, similar to the discriminator in GAN, the model can force the diffusion model to generate more similar images to the real datasets. Both studies enable the users to control the level of fidelity and diversity in diffusion sampling, where the optimal FID is obtained with a moderate level of fidelity and diversity. Instead, we focus on a high level of weight in the discriminator to generate images with high diversity, rather than repeating typical images with high fidelity.

Generation diversity and augmentation diversity We demonstrate the additional experiments on both using generation diversity by DG and augmentation diversity. Figure 8 shows CAS and private classification accuries (ε=2) for $w_d = [0, 3, 10, 20]$ for DG with or without traditional augmentation on CIFAR-10. The combination of DG(w_d=10)+Aug performs best (85.93%), surpassing EDM+Aug (85.48% in Table 6). Yet, we conclude that the gain is marginal than DG alone (2.53%p). Due to DG's extra complexity, we rather use EDM alone for experiments while using DG for investigating the important factors in generation. We believe that EDM can cover the data diversity (more than DDPM) in ID public synthesis. Note that data diversity remains a critical factor in enhancing performance when the model is not trained on specific in-distribution public data, as demonstrated in Table 10.

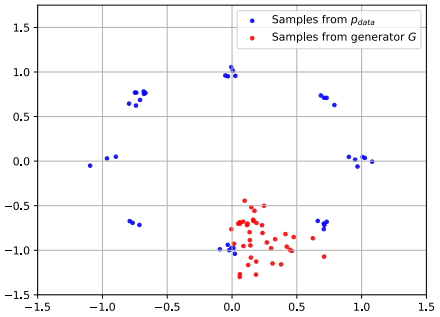
C.4. Training Dynamics

For a detailed analysis of the learning dynamics of Section 5.3, we show additional results of the private training phase. We measure the gradient norm and loss of private and synthetic data, as shown in Figure 9. As DP-SGD focuses on private data, the loss of private data decreases constantly, while the loss of public data is increasing. On the other hand, DOPE-SGD minimizes the loss of both synthetic and private data. However, in a certain range of training, DOPE-SGD occasionally fails to converge and thus diverge, where their gradient norms explode and the accuracy plummets to 0. As private training usually uses a larger learning rate than standard training, explicitly minimizing the public gradient might be dangerous as the model can be overfitted to training data and stuck into sharp minima during training.

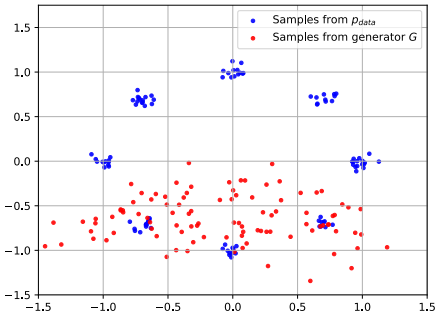
Interestingly, Figure 9d illustrates the private gradient norm without clipping. Both DOPE-SGD and DP-SGD show diverging gradient norms during training. This indicates that private training does not decrease the gradient norm, but rather decreases the clipped gradient norm.



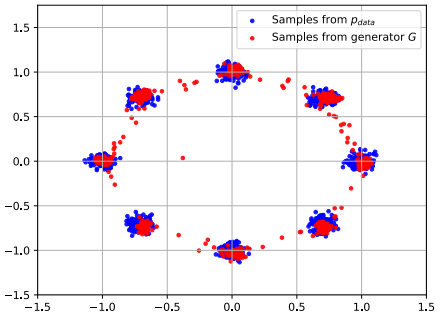
(a) GAN trained with 160 samples.



(b) GAN trained with 400 samples.



(c) GAN trained with 1K samples.



(d) GAN trained with 10K samples.

Figure 7. Mode collapse of GAN with a different number of samples on Ring data.

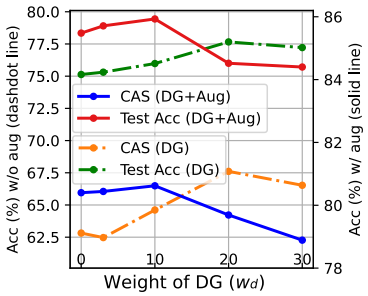


Figure 8. CAS and test accuracies by different synthetic images on the guidance weight w_d.

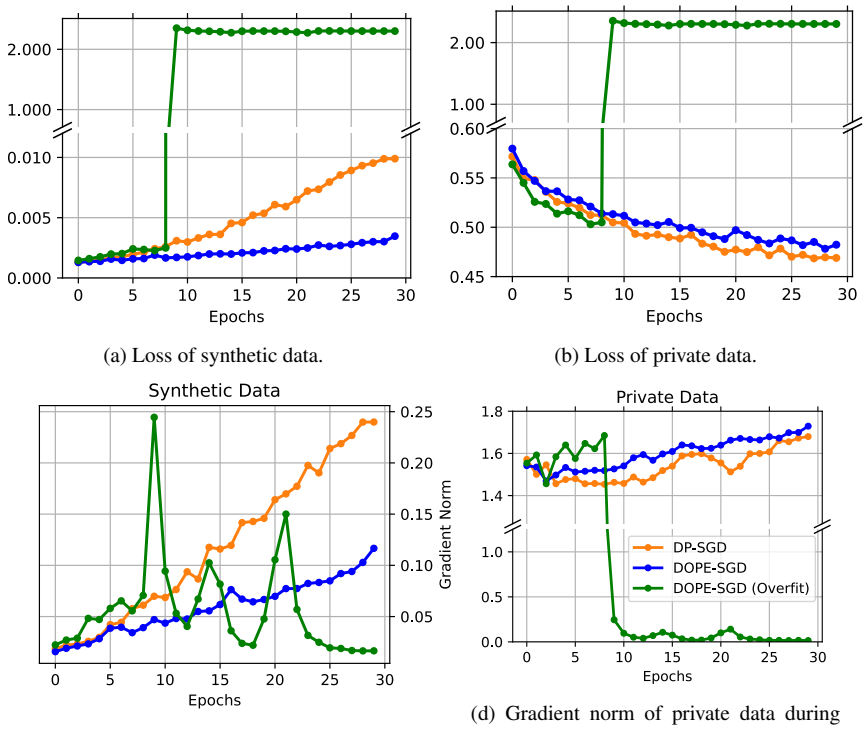
D. Ablation study

D.1. Synthetic Public Data using Text-to-Image Diffusion Models

To mitigate the drawbacks of the proposed method concerning training diffusion models, we conduct experiments with the pre-trained Stable Diffusion (SD) [6, 17], without training diffusion models. The text-to-image diffusion model is trained on a larger public LAION dataset [18]. Bao et al. [2] already utilized that using pre-trained diffusion models does not incur privacy leakage for differentially private classification tasks.

In Table 10, we examine various datasets including EuroSAT [10], which focuses on land use and land cover classification; PathMNIST [21], representing colon pathology datasets; and [11] for the case of sensitive attributes, which includes human face images depicting seven distinct race groups: White, Black, Indian, East Asian, Southeast Asian, Middle Eastern, and Latino. Without any additional fine-tuning, we provide the text prompt 'a color photo of the face of {*group name*} race for the FairFace datasets, and 'a photo of a {*class name*}.' for the other datasets to generate synthetic datasets. The data samples are resized to the size of 32×32 for training. As a result, Table 10 illustrates the strength of diffusion and diversity for warm-up training.

While we highlight the effectiveness of synthesis on a small fraction of public data, someone might argue that using ID public data might be dangerous for privacy. Thus, we conduct experiments with the Stable Diffusion without utilizing any ID public data. We compare test accuracies after private training, whether pre-trained on synthetic data or not. The results presented in Table 15 indicate that warm-up training using pre-trained diffusion models can notably enhance classification performance. This gives strong evi-



(a) Loss of synthetic data.

(b) Loss of private data.

(c) Gradient norm of synthetic data during training. training **without clipping**.

(d) Gradient norm of private data during

Figure 9. Learning dynamics during private training of various optimization methods.

Table 15. Comparison of using synthetic data from pre-trained text-to-image diffusion models, without using in-distribution public data.

Dataset	Syn	Test Acc (%)		
		$\varepsilon = 2$	$\varepsilon = 6$	$\varepsilon = 10$
CIFAR-100	✗	18.19	33.61	39.09
	✓	**52.57**	**58.94**	**60.85**
FairFace	✗	39.09	44.39	45.34
	✓	**48.21**	**53.32**	**54.75**

dence that private training, which suffers from the lack of training data samples, can be improved using diffusion synthesis. Note that the accuracies of synthetic images after the warm-up ($\varepsilon = 0$) are 29.25% for CIFAR-100 and 28.46% for FairFace. The resulting images are presented in Figure 11 of Appendix E.

D.2. Effect of Generated Data Size

To analyze the effect of the generated data size, we train models with 5K, 20K, and 40K synthetic samples in the warm-up phase until convergence for each model. The results in Table 16 indicate that data size remains a critical factor, even when applying the identical EDM model. Note that the 40K sample size is equal in the experiments of [15].

Table 16. Performance by the different number of synthetic data on CIFAR-10.

Generated	Test Acc (%)		
	$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 6$
5K	77.50±0.03	77.97±0.02	78.22±0.31
20K	83.21±0.12	84.05±0.04	84.16±0.08
40K	85.11±0.05	85.66±0.07	86.04±0.01

Table 17. Performance by using different amounts of public data of CIFAR-10.

Public Size	Method	Test Acc (%)		
		$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 6$
500 (1%)	[15]	68.9	72.1	77.1
	Ours	**78.32**	**80.99**	**82.15**
2K (4%)	[15]	75.1	77.9	80.0
	Ours	**85.48**	**86.49**	**87.06**

D.3. Different Amount of Public Data

To analyze the effects of varying amounts of public data on synthesis, we reproduce the CIFAR-10 results using only 500 public samples (1% of the training data), with each class comprising only 50 data points. We keep all other hyperparameters the same, including diffusion synthesis,

Table 18. Performance comparison of trained on the public and augmented data on CIFAR-10.

	$\varepsilon = 0$	$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 6$
Warm (w/ Aug and Opt)	69.63	75.44	77.22	77.99
Ours	**80.72**	**85.48**	**86.49**	**87.06**

Table 19. Test accuracy (%) of private classification using pre-trained models.

Datasets	Privacy budget		$\varepsilon = 1$			$\varepsilon = 4$		
	Architecture	Pre-trained	Cold	Warm	Ours	Cold	Warm	Ours
CIFAR -100	CrossViT small 240 (26.3M)	✓	66.88	74.70	**78.91**	72.91	78.80	**81.75**
	CrossViT 18 240 (42.6M)	✓	71.27	78.60	**81.39**	76.22	81.73	**83.75**
	DeiT base patch16 224 (85.8M)	✓	63.84	81.55	**81.62**	72.84	**84.60**	84.43
	CrossViT base 240 (103.9M)	✓	71.08	77.30	**79.65**	76.63	80.43	**82.63**

Table 20. Test accuracy of adversarial training on CIFAR-10.

Datasets	Architecture	Public	Synthesis	Methods	$\varepsilon = 2$
CIFAR -10	WRN-16-4 (2.74M)	✓	✓	Adversarial training $(8/255)$ in warm-up	77.25
		✓	✓	Adversarial training $(2/255)$ in warm-up	81.55

warm-up training, and private learning. The results presented in Table 17 demonstrate that our method consistently surpasses the previous method even when assuming a small data size.

D.4. Results of Augmentation and Generalization without Diffusion Synthesis

The results on the augmented public data and SAM, without using EDM synthesis, are presented in Table 18. The table illustrates the importance of diffusion synthesis alone without relying on augmentation and generalization.

D.5. Different Privacy Budget ε on Pre-trained CIFAR-100

We additionally append the experimental results of CIFAR-100 with pre-trained models on ImageNet, in Table 19 with privacy budget of $\varepsilon \in \{1, 4\}$.

D.6. Additional Optimization during Warm-up

We try to use other techniques for private learning to improve classification performance. The classification results with $(2, 10^{-5})$-DP are presented in Table 20 using adversarial training [14]. Adversarial training aims to make decision boundaries smooth in terms of input space. Thus, we hypothesize that using adversarial training in the warm-up training can make the private training easier. Furthermore, the importance of generative models is well under-studied in adversarial training [8, 20]. Similar to DP training, as the task of adversarial training is harder than standard training, Wang et al. [20] observed that utilizing the synthetic data with EDM can improve the generalization performance and prevent robust overfitting without extra data samples. Therefore, we adopt to use of adversarial training on the top of synthetic data in the warm-up phase. However, adversarial training methods are designed to reduce the accuracy of PGD-10 [14], and the standard accuracy is significantly dropped. Therefore, the private classification results are also decreased.

E. Generated images

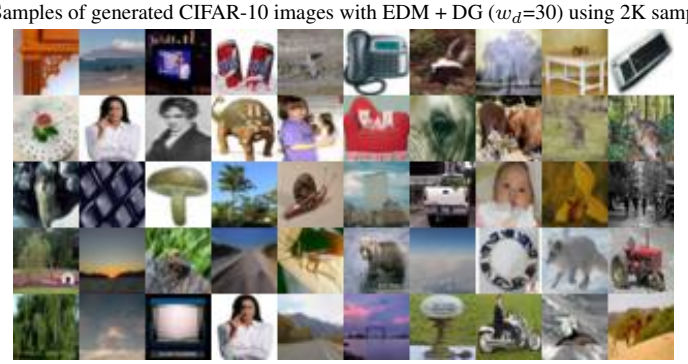
We illustrate the samples of generated images using ID public data. Figure 10a demonstrates the memorization within CIFAR-10, by random sampling synthetic images in the first row and choosing the nearest samples in the synthetic dataset. Figure 10b and Figure 10c are obtained from generated images of CIFAR-10, and Figure 10d is obtained from generated images of CIFAR-100.

Furthermore, we present samples of generated images utilizing pre-trained text-to-image diffusion models in Figure 11, which is discussed in Appendix D.

(a) Illustration of memorization within generated CIFAR-10 images.

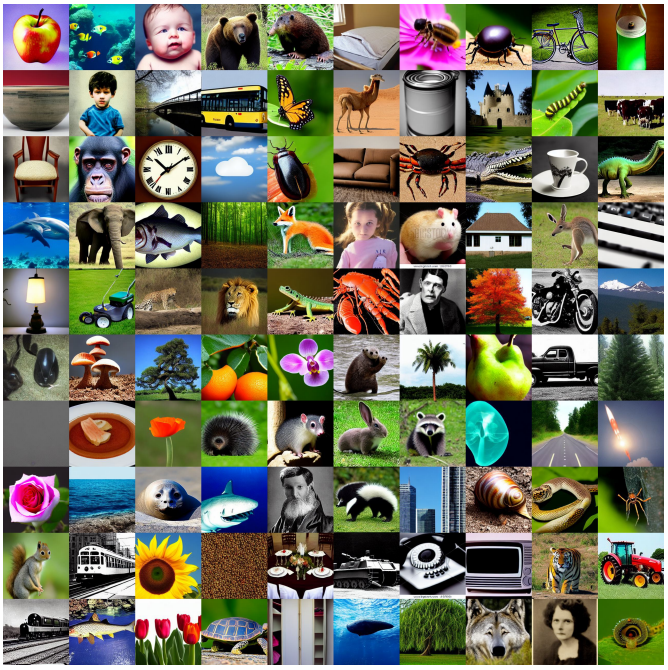
(b) Samples of generated CIFAR-10 images with EDM using 2K samples.

(c) Samples of generated CIFAR-10 images with EDM + DG (w_d=30) using 2K samples.



(d) Samples of generated CIFAR-100 images with EDM using 2K samples.

Figure 10. Samples of generated images using in-distribution public data.



(a) Illustration of generated CIFAR-100 images. Each image represents one distinct class.

(b) Illustration of generated FairFace images. Each row illustrates the sampled images for each race.

Figure 11. Samples of generated images using pre-trained text-to-image diffusion models.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016. 3

[2] Wenxuan Bao, Francesco Pittaluga, Vincent Bindschaedler, et al. Dp-mix: Mixup-based data augmentation for differentially private learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 5

[3] Zhiqi Bu, Jialin Mao, and Shiyun Xu. Scalable and efficient training of large convolutional neural networks with differential privacy. In *Advances in Neural Information Processing Systems*, 2022. 2

[4] Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022. 2

[5] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 4

[6] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit Haim Bermano, Gal Chechik, and Daniel Cohen-or. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *The Eleventh International Conference on Learning Representations*, 2022. 5

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 4

[8] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. Improving robustness using generated data. *Advances in Neural Information Processing Systems*, 34:4218–4233, 2021. 7

[9] Venkatesan Guruswami. Lecture notes in information theory and its applications in theory of computation. http://https://www.cs.cmu.edu/~venkatg/teaching/ITCS-spr2013/, 2013. 1

[10] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019. 5

[11] Kimmo Karkkainen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1548–1558, 2021. 5

[12] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022. 1, 2

[13] Dongjun Kim, Yeongmin Kim, Se Jung Kwon, Wanmo Kang, and Il-Chul Moon. Refining generative process with discriminator guidance in score-based diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 16567–16598. PMLR, 2023. 2, 4

[14] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 7

[15] Milad Nasr, Saeed Mahloujifar, Xinyu Tang, Prateek Mittal, and Amir Houmansadr. Effectively using public data in privacy preserving machine learning. In *Proceedings of the 40th International Conference on Machine Learning*, pages 25718–25732. PMLR, 2023. 6

[16] Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan Yuille. Micro-batch training with batch-channel normalization and weight standardization. *arXiv preprint arXiv:1903.10520*, 2019. 2

[17] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 5

[18] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022. 5

[19] Emanuel Todorov. Linearly-solvable markov decision problems. *Advances in neural information processing systems*, 19, 2006. 1

[20] Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. Better diffusion models further improve adversarial training. In *Proceedings of the 40th International Conference on Machine Learning*, pages 36246–36263. PMLR, 2023. 7

[21] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023. 5

[22] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021. 2