A Algorithm: DP-SGD

Algorithm 2 DP-SGD from Abadi et al. [2]

```
Require: Training data $x_1, ..., x_N$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate $\eta_t$, noise scale $\sigma$, group size $B$, gradient norm bound $C$.
  Initialize $\theta_0$ randomly
  for $t \in [T]$ do
      Take a random sample $B_t$ with sampling probability $B/N$
      for each $i \in B_t$ do
          Compute $\mathbf{g}_t(x_i) \leftarrow \nabla_\theta \mathcal{L}(\theta_t, x_i)$
          $\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)/\max(1, \frac{||\mathbf{g_t(x_i)}||_2}{C})$
      end for
      $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{B}\sum_i(\bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))$
      $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$
  end for
Ensure: $\theta_T$ and and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method
```

B Additional Experimental Results

B.1 How useful is the synthetic data?

Let $\mathcal{S}_1$ and $\mathcal{S}_2$ denote equally sized disjoint subsets of the private dataset, and $\mathcal{D}$ an equally sized set of synthetic samples generated via diffusion, as described in Section 3.1.

For this experiment, we consider three different training methods.

- **Method A**: Fine-tune on $\mathcal{S}_1$ using DP-MIX$_{\text{SELF}}$.
- **Method B**: Fine-tune on $\mathcal{S}_1$ using DP-MIX$_{\text{DIFF}}$ with mixup images sampled from $\mathcal{D}$.
- **Method C**: Same as Method B, but replace set $\mathcal{D}$ with set $\mathcal{S}_2$. Note, we assume that $\mathcal{S}_2$ is publicly available, so accessing it during training does not incur a privacy cost.

The intuition behind this experiment is that method C provides an upper bound for both A and B, since, in the best case, the distribution of the synthetic data would exactly match that of the private data. The experimental results, presented in Table 10, validate this intuition.

For CIFAR-100, Caltech256, SUN397, and Oxford Pet, method C outperforms method A. Consistent with this, method B, our proposed DP-MIX$_{\text{DIFF}}$, also provides a performance boost. On the other hand, for CIFAR-10 and EuroSAT, method C, despite directly using private data for mixup, does not meaningfully improve performance. Similarly, method B also does not improve performance. For EuroSAT, method B slightly decreases performance, due to the large domain gap and bigger fid value between the synthetic and private data.

Table 10: **Test Accuracy (%) using different training methods** with Vit-B-16 and ConvNext on various datasets after fine-tuning . We set the $\varepsilon = 2$ and $\delta = 10^{-5}$.

	Vit-B-16			**ConvNext**		
Dataset	**A**	**B**	**C**	**A**	**B**	**C**
CIFAR-10	96.9 (.1)	96.9 (.0)	97.0 (.1)	96.2 (.1)	96.1 (.1)	96.2 (.1)
CIFAR-100	81.3 (.3)	82.0 (.2)	82.1 (.1)	76.7 (.3)	78.0 (.1)	78.4 (.1)
EuroSAT	93.7 (.1)	91.4 (.1)	93.9 (.2)	93.8 (.2)	92.3 (.1)	93.9 (.2)
Caltech 256	76.0 (.4)	81.9 (.1)	82.3 (.2)	76.4 (.3)	81.1 (.1)	81.6 (.3)
SUN397	70.8 (.2)	72.4 (.1)	73.8 (.1)	70.1 (.2)	72.3 (.1)	73.6 (.2)
Oxford Pet	65.1 (.3)	68.3 (.1)	68.3 (.1)	64.6 (.2)	66.0 (.1)	68.1 (.1)

B.2 Reproducing De et al.[9] and DP-MIX$_{\text{SELF}}$ in JAX

For completeness, we train WRN-16-4 on CIFAR10 using Self-Aug and our proposed DP-MIX$_{\text{SELF}}$ using [9]'s official JAX code (`https://github.com/deepmind/jax_privacy`). As in [9], we repeat each experiment 5 times and report median test accuracy in Table 11. The results are consistent with those presented in the rest of our paper — DP-MIX$_{\text{SELF}}$ outperforms Self-Aug for different privacy budgets.

Table 11: **Performance comparison on JAX.**

Method	$\varepsilon = 1$	$\varepsilon = 8$
Self-Aug	56.3 (.3)	79.4 (.1)
DP-MIX$_{\text{SELF}}$	57.1 (.4)	80.0 (.2)

B.3 Pure-DP-MIX$_{\text{DIFF}}$

To demonstrate the influence of K_{BASE} in our method, we set $K_{\text{BASE}} = 0$ and call it Pure-DP-MIX$_{\text{DIFF}}$. In effect Pure-DP-MIX$_{\text{DIFF}}$ is simply mixing up the synthetic examples themselves. We test it on CIFAR-100 and represent it in Table 12. We can see that Pure-DP-MIX$_{\text{DIFF}}$ offers much worse performance than both DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$, although it still offers better performance than Self-Aug due to the beneficial effects of mixup. More generally, we think that Pure-DP-MIX$_{\text{DIFF}}$ will tend to worsen an overfitting problem whenever there is a large domain gap between the original training data and the diffusion samples. DP-MIX$_{\text{DIFF}}$ does not suffer from this problem because it ensures that (augmented versions) of the original training data samples are seen during training.

Table 12: **Test Accuracy (%) of Pure-DP-MIX$_{\text{DIFF}}$ ($K_{\text{BASE}} = 0$) on CIFAR-100 with Vit-B-16 model.** We set $\delta = 10^{-5}$ and $\varepsilon = 1$. We can observe that Pure-DP-MIX$_{\text{DIFF}}$ does not improve performance, which shows the necessitate of using base augmentations ($K_{\text{BASE}} > 0$).

Method	**Test accuracy**
Selg-Aug	79.3 (.2)
DP-MIX$_{\text{SELF}}$	81.8 (.2)
DP-MIX$_{\text{DIFF}}$	**82.0** (.1)
Pure-DP-MIX$_{\text{DIFF}}$	80.9 (.2)

B.4 Running time

We provide the running time for different methods in Table 13. All experimental runs utilized a single A100 GPU and were based on the same task of finetuning the Vit-B-16 model on the Caltech256 dataset for 10 epochs. Due to additional augmentation steps, the training time of our methods is longer than prior work.

Table 13: **Running time** for different methods of the same task(fine-tuning Vit-B-16 on Caltech256 for 10 epochs). We use one A100 GPU for each training method.

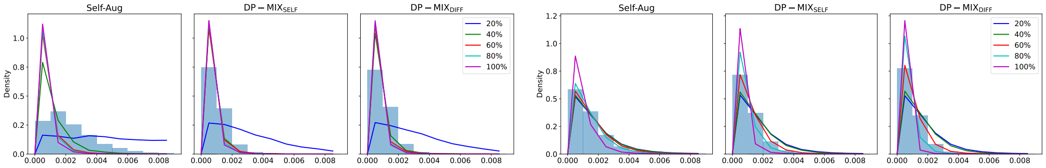
Method	**Self-Aug**	**DP-MIX$_{\text{SELF}}$**	**DP-MIX$_{\text{DIFF}}$**
Running time	2h 12min	7h 33min	7h 40 min

B.5 Effect of Mixup on Gradients

We study what happens to gradients and parameter updates during training for our methods versus Self-Aug. Fig. 2 plots the per-parameter gradient magnitude averaged over samples at each epoch (prior to clipping and noise adding). The histogram shows the data averaged over all training epochs and the X% color lines show that data only for the epoch at X% of the total training process. There are 10 epochs for this experiment – for example, the line for 20% shows the data for epoch 2 (out of 10).

The figure shows more concentrated values for our methods compared to the Self-Aug baseline, which suggests more stable training and faster convergence. Standard deviations for CIFAR-10 with Self-Aug, DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$ are: $2.16 \cdot 10^{-3}$, $9.76 \cdot 10^{-4}$ and $9.59 \cdot 10^{-4}$, respectively.

For Caltech256 they are: $1.43 \cdot 10^{-3}$, $1.07 \cdot 10^{-3}$ and $9.32 \cdot 10^{-4}$, respectively. This is consistent with experimental results of test accuracies for each method.



(a) CIFAR-10 (b) Caltech256

Figure 2: **Per-parameter gradient magnitude before clipping and adding noise** on CIFAR-10(a) and Caltech256(b) with fine-tuning Vit-B-16 model with $\varepsilon = 1$ and $\delta = 10^{-5}$. The different curves represent different training stages. Values for our proposed DP-MIX$_{\text{SELF}}$ and DP-MIX$_{\text{DIFF}}$ are more concentrated suggesting more stable training and faster convergence.

C Additional Experimental Details

Datasets We use the following datasets:

- **CIFAR-10** is a widely-utilized dataset in the field of computer vision, serving as a standard for evaluating image recognition algorithms. Collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton [21], this dataset is a crucial tool for machine learning research. The dataset consists of 60,000 color images, each sized at 32×32 pixels, and categorized into 10 different classes like cats, dogs, airplanes, etc. We use 50,000 data points for training, and 10,000 for the test set.
- **CIFAR-100** is a well-regarded dataset in the domain of computer vision, typically used for benchmarking image classification algorithms. This dataset, also collected by Krizhevsky et al. [21], is a key resource in machine learning studies. CIFAR-100 comprises 60,000 color images, each of 32x32 pixel resolution. What distinguishes it from CIFAR-10 is the higher level of categorization complexity; the images are sorted into 100 distinct classes instead of 10. We use 50,000 data points for training, and 10,000 for the test set.
- **EuroSAT** [18] is a benchmark dataset for deep learning and image classification tasks. This dataset is composed of Sentinel-2 satellite images spanning 13 spectral bands and divided into ten distinct classes. It has 27,000 labeled color images which size is 64×64. We use 21600 as the training set and 5400 as a test set.
- **Caltech 256** [15]. Caltech is commonly used for image classification tasks and comprises of 30607 RGB images of 257 different object categories. For our experiments, we designated 80% of these images for training and the remaining 20% for testing.
- **SUN397** The Scene UNderstanding (SUN) [42, 43] database contains 108,754 RGB images from 397 classes. For our experiments, we use 80% of these images for training and the remaining 20% for testing.
- **Oxford Pet** [28] contains 37 classes of cats and dogs and we use 3680 images for training and the rest 3669 images for testing.

For all our experiments, we maintained the clip norm at $C = 1$, with the exception of Mix-ghost clipping where we used $C = 0.05$ as required by the original paper [5] and its implementation[4]. The noise level was automatically calculated by Opacus based on the batch size, target ε, and δ, as well as the number of training epochs.

Implementation details of Section 3. In this experiment, we set $K_{\text{BASE}} = 16$ and adjust the hyperparameters according to the recommendations in the original paper [9]. To facilitate the implementation in PyTorch and enable microbatch processing, we adapt our code based on two existing code bases [5].

Implementation details of Section 5.1. In training our models from scratch, we adhere to a K_{BASE} value of 16 and adjust the other hyperparameters in accordance with the guidelines provided in the

---

[4] `https://github.com/JialinMao/private_CNN`

[5] `https://github.com/facebookresearch/tan` and `https://github.com/ChrisWaites/pyvacy`

original papers [9, 5]. For our fine-tuning experiments, we maintain the same K_{BASE} value and perform hyperparameter searches in each case to ensure we utilize the optimal learning rate. Importantly, we do not incorporate any learning rate schedules, as per the suggestions of other research papers [5, 9] that indicate such schedules do not yield performance improvements.

Implementation details of Section 5.2 and Appendix B. In order to generate synthetic samples, we feed the text prompt '`a photo of a <class name>`' to the diffusion model [13]. For each dataset, the number of synthetic samples we generate is equal to the number of real images in the dataset.

Source Code. Readers can find further experimental and implementation details in our open-source code at `https://github.com/wenxuan-Bao/DP-Mix`.

D Ethical Considerations & Broader Impacts

In this paper, we propose an approach to use multi-sample data augmentation techniques such as mixup to improve the privacy-utility tradeoff of differentially-private image classification. Since differential privacy offers strong guarantees, its deployment when training machine learning has the potential to substantially reduce harm to individuals' privacy. However, some researchers have observed that although the guarantee of differential privacy is a worst-case guarantee, the privacy obtained is not necessarily uniformly spread across all individuals and groups. Further, there is research suggesting that differential privacy may (in some cases) increase bias and unfairness, although these findings are disputed by other research.