Private Image Generation with Dual-Purpose Auxiliary Classifier

Chen Chen[1] Daochang Liu[1] Siqi Ma[2] Surya Nepal[3] Chang Xu[1]

[1]School of Computer Science, Faculty of Engineering, The University of Sydney

[2]The University of New South Wales, Canberra [3]CSIRO, Data61

cche0711@uni.sydney.edu.au, {daochang.liu, c.xu}@sydney.edu.au, siqi.ma@adfa.edu.au, surya.nepal@data61.csiro.au

A. Algorithm

We present the pseudocode of the warm-start steps of our proposed method in Algorithm 1, where non-private discriminators are pre-trained using different sub-sampled datasets. The pre-trained discriminators are then loaded as initialisations for the subsequent training of the private generator. This does not consume any privacy costs since it is only the private generator that we would release after training. Note that Algorithm 1 is kept identical to GS-WGAN [2] for a fair comparison of methodologies.

Algorithm 1 DP-GAN-DPAC Warm-start

```
Input: Dataset $\mathcal{D}$, subsampling rate $\gamma$, warm-start iterations $T_w$, learning rates $\eta_D$ and $\eta_G$, the number of iterations per generator iteration for the discriminator $n_{dis}$, batch size $B$
Output: Non-private discriminator $\boldsymbol{\theta}_D^k$ for $k = 1, 2, ..., K$ ($K = 1/\gamma$)
Subsample (without replacement) the dataset $\mathcal{D}$ into subsets $\{\mathcal{D}_k\}_{k=1}^K$ with rate $\gamma$ where $K = 1/\gamma$;
for $k$ in $\{1, ..., K\}$ in parallel do
    Initialise non-private generator $\boldsymbol{\theta}_G^k$ and non-private discriminator $\boldsymbol{\theta}_D^k$
    for $step$ in $\{1, ..., T_w\}$ do
        for t in $\{1, ..., n_{dis}\}$ do
            Sample batch $\{\boldsymbol{x}_i\}_{i=1}^B \subseteq \mathcal{D}_k$;
            Sample batch $\{\boldsymbol{z}_i\}_{i=1}^B$ with $\boldsymbol{z}_i \sim \mathcal{P}_{\boldsymbol{z}}$;
            Compute mean discriminator gradient $\boldsymbol{g}_D$;
            $\boldsymbol{\theta}_D^k \leftarrow \boldsymbol{\theta}_D^k - \eta_D \cdot \boldsymbol{g}_D$;
        end for
        Compute mean generator gradient $\boldsymbol{g}_G$;
        $\boldsymbol{\theta}_G^k \leftarrow \boldsymbol{\theta}_G^k - \eta_G \cdot \boldsymbol{g}_G$;
    end for
    return Discriminator $\boldsymbol{\theta}_D^k$
end for
```

B. Hyperparameter Settings

For MNIST and Fashion MNIST, We adopt the same hyper-parameter settings for components inherit from GS-WGAN, such as the learning rate and optimiser for the generator and discriminator networks, the number of warm-start iterations $T_w = 2000$, the number of discriminator iteration per update of generator: $n_{dis} = 5$, the sub-sampling rate $\gamma = 1/1000$, noise scale $\sigma = 1.07$, number of private generator iterations $T = 20000$ upon exhausting all privacy budget of $\epsilon = 10$. On top of that, our use of the dual-purpose auxiliary classifier and the deliberate design of its training process introduce some additional hyper-parameters: the proportion of feedback allocated to discriminator as opposed to classifier $\beta = 0.8$, the iteration to launch classifier $t_c = 6000$, and the number of iterations per generator update for the classifier using fake data and real data respectively $n_f = 10$ and $n_r = 10$. The architecture and hyper-parameter settings for the auxiliary classification network is identical to the discriminator, except that after the final linear mapping, we have 10 outputs instead of 1 for the classifier since it conducts multi-class classifications on these two datasets. Thus, to compute the Wasserstein distance between the true class and the fake class, an extra step is to compute the difference between the score of the one output neuron that corresponds to the true class label, and the mean score of the remaining 9 output neurons that correspond to all other classes.

For CelebA, as there are 162770 examples in the training set, we use sub-sampling rate of $\gamma = 1/2543$ to ensure each sub-sample contains 64 examples, which is deliberately chosen as powers of two and can then be easily randomly divided into two batches of $B = 32$ non-overlapping examples during every epoch of each sub-sample. We choose the noise scale to be $\sigma = 0.61135$ since under the current settings of batch size and sub-sampling rate, $\sigma = 0.61135$ allows a privacy cost of 9.9993 to be consumed after exact $T = 20000$ generator iterations, which is just within the privacy budget of $\epsilon = 10$. The warm-start (Appendix A) is conducted for $T_w = 12000$ iterations for obtaining better pre-trained discriminators. Also, since we generate condition on gender, the classifier does binary classification tasks in this scenario. The remaining settings are as follows: $n_{dis} = 10$, $n_f = 10$, $n_r = 10$, $\beta = 0.8$, and $t_c = 1000$.

C. Additional Performance Comparisons for Reversed Utility

In the paper, we have compared the reversed utility (r2g%) of our method with GS-WGAN [2] as it is the backbone that we use to further introduce the dual-purpose auxiliary classifier. Comparisons with GS-WGAN [2] allow clear demonstrations of the effectiveness of the auxiliary classifier holding all remaining designs unchanged.

Here, we present additional performance comparisons for the reversed utility (r2g%) of our method with other two most promising baselines [1, 3] in addition to GS-WGAN [2] in Tab. 1. The re-implementations of all baseline methods are based on their officially released codes. As shown in Tab. 1, our method consistently shows distinct advantage over all baselines. Thus, our generated outputs are most generalisable to the true features that distinguish between their semantic classes. This allows our outputs to be most difficult to tell from their real classes by real-world classifiers.

	MNIST		F-MNIST	
Method ↑	MLP	CNN	MLP	CNN
DataLens [3]	0.39	0.47	0.44	0.48
DPSinkhorn [1]	0.98	0.99	0.79	0.86
GS-WGAN [2]	0.99	0.99	0.85	0.85
Ours	**1.00**	**1.00**	**0.97**	**0.98**

Table 1. Comparing real2gen accuracy ↑ on various datasets.

D. Trade-offs

In addition to comparing performances of each method's final checkpoint when all privacy budgets of $\epsilon = 10$ are consumed, we here present ongoing comparisons of performances (quality, standard utility, and reversed utility) over the baseline under a range of privacy budgets. This demonstrates the consistency of our method's obtaining better privacy-quality and privacy-utility trade-offs. Note that all comparisons for MNIST and Fashion MNIST start from privacy budget of $\epsilon = 5.08$, and from $\epsilon = 3.35$ for CelebA. This is due to the late launch of our auxiliary classifier, which is introduced into the pipeline from iteration $t_c = 6000$ (when $\epsilon = 5.08$) for both MNIST and Fashion MNIST; and from iteration $t_c = 1000$ (when $\epsilon = 3.35$) for CelebA. All prior results are identical to those of GS-WGAN [2] and are thus not included for comparisons.

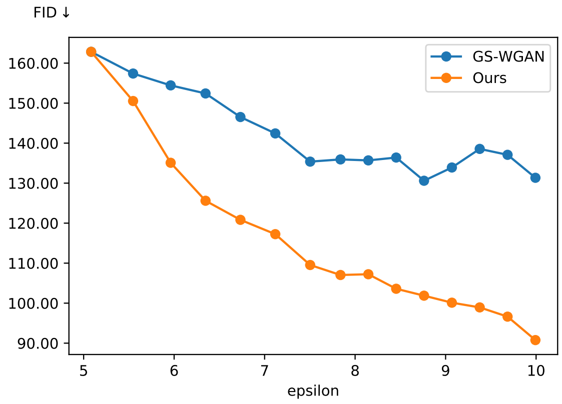

Figure 1. Comparing FID on F-MNIST

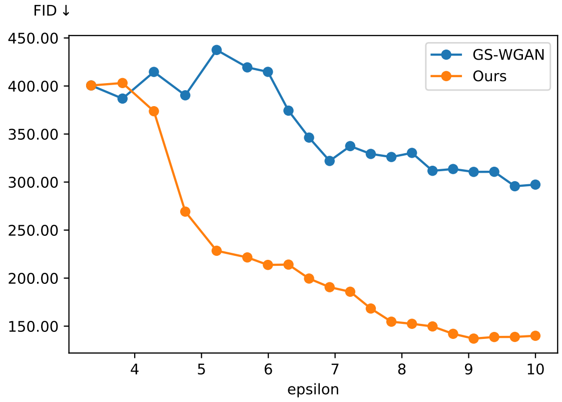

Figure 2. Comparing FID on CelebA

D.1. Privacy vs. Quality

We show in Fig. 1, Fig. 2, and Fig. 3 that for a given privacy budget, our method obtains better quality results (lower FID) than the baseline, thus provides better privacy-quality trade-offs. Comparing to the baseline, our results demonstrate much more rapid accelerations of the decrease of FID from the point that the dual-purpose auxiliary classifier is first introduced into the method design. Besides, the advantage is particularly more obvious and consistent for the more challenging datasets such as Fashion MNIST (Fig. 1) and CelebA (Fig. 2). Image generation on MNIST (Fig. 3) is a relatively easier task that the baseline can also generate comparable high-quality outputs as measured by FID, so we supplement the performance analyses on MNIST by also providing another comparison using Inception Score (IS) as the metric. Fig. 4 shows that soon after the introduction of our auxiliary classifier, the IS stabilises at the highest possible level after an astonishing surge. The advantage over baseline is obvious.

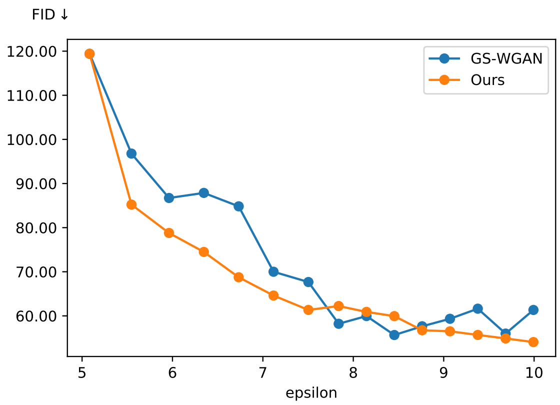

Figure 3. Comparing FID on MNIST

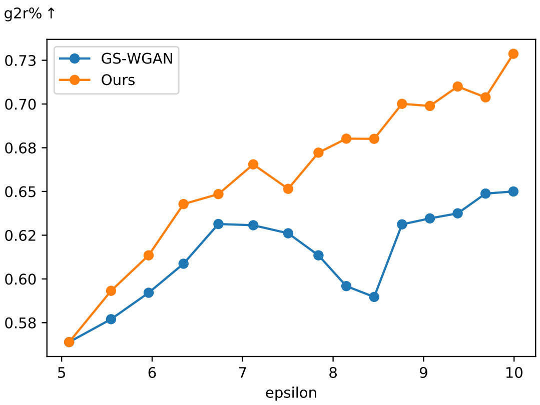

Figure 5. Comparing g2r% on F-MNIST

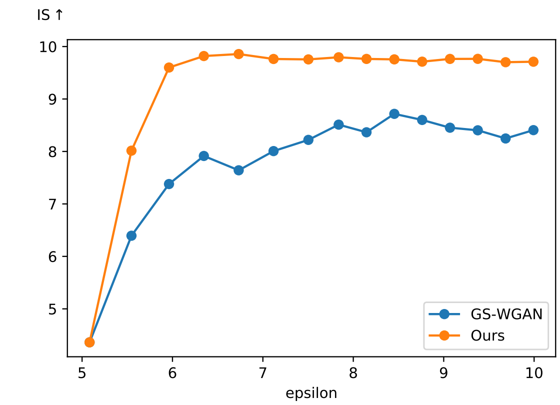

Figure 4. Comparing IS on MNIST

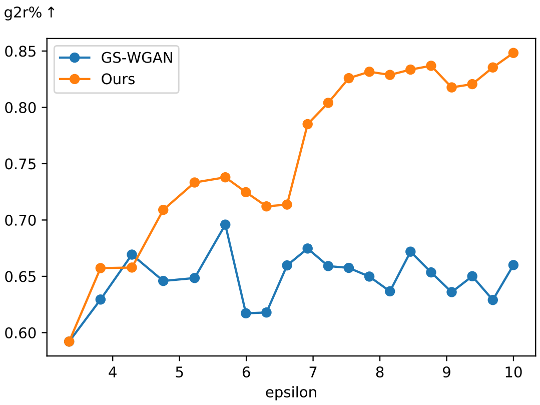

Figure 6. Comparing g2r% on CelebA

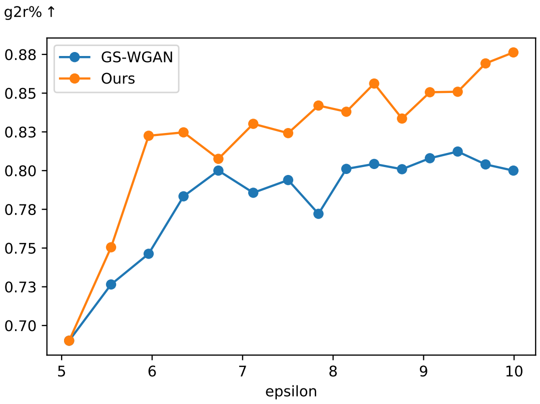

Figure 7. Comparing g2r% on MNIST

D.2. Privacy vs. Standard Utility

We show in Fig. 5, Fig. 6, and Fig. 7 that for a given privacy budget, our method obtains better **standard** utility results (higher g2r%) than the baseline, thus provides better privacy-utility (standard) trade-offs. For Fashion MNIST (Fig. 5), both methods illustrate increasing trend of the g2r%, but our method shows more rapid and consistent rises, compared to the baseline's slow and fluctuating increases. The results on CelebA (Fig. 6) are even more obvious, where the baseline struggles to generate semantically meaningful outputs that could train a binary classifier to demonstrate significant advantage over random guesses. Our results are consistently better and reaches a g2r% of 0.85 upon exhausting all privacy budgets of $\epsilon = 10$. Finally, Fig. 7 shows our performance on g2r% is also distinctive for easier generation task on MNIST.

D.3. Privacy vs. Reversed Utility

We also show in Fig. 8, Fig. 9, and Fig. 10 that for a given privacy budget, our method obtains better **reversed** utility results (higher r2g%) than the baseline, thus provides better privacy-utility (reversed) trade-offs. It is worth noting that our method could reach and subsequently stay at a real2gen accuracy (r2g%) of 1.00 or fractionally lower than 1.00 soon after the introduction of auxiliary dual-purpose

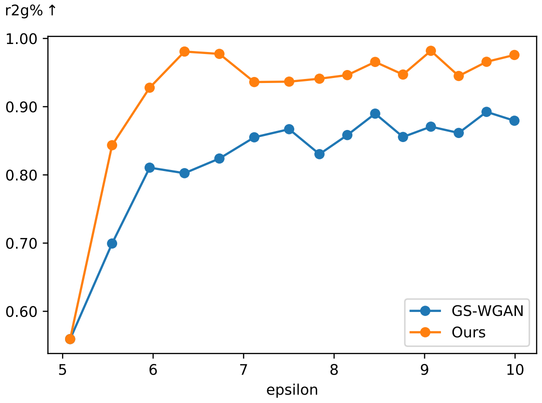

Figure 8. Comparing r2g% on F-MNIST

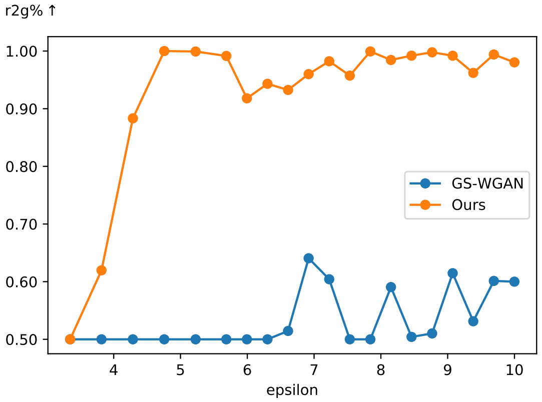

Figure 9. Comparing r2g% on CelebA

			gen2real ↑		real2gen ↑	
Method	IS ↑	FID ↓	MLP	CNN	MLP	CNN
Baseline	9.23	61.34	0.79	0.80	0.99	0.99
w/o g2r	9.73	58.34	0.84	0.84	**1.00**	**1.00**
w/o r2g	9.84	58.97	0.84	0.84	**1.00**	**1.00**
w/o seq	7.92	66.61	0.77	0.73	0.99	0.98
w/o init	**9.91**	64.84	0.82	0.81	**1.00**	**1.00**
Full	9.71	**54.06**	**0.85**	**0.88**	**1.00**	**1.00**

Table 2. Ablation studies on MNIST. Each sub-component of our final design has demonstrated its expected effectiveness.

classifier for all three tested datasets. On the other hand, the baseline could only approach our performance on the relatively easier MNIST dataset (Fig. 10), while the gap is biggest for the most challenging CelebA dataset (Fig. 9). These have demonstrated the auxiliary classifier's significant contribution on the generated outputs' generalisability.

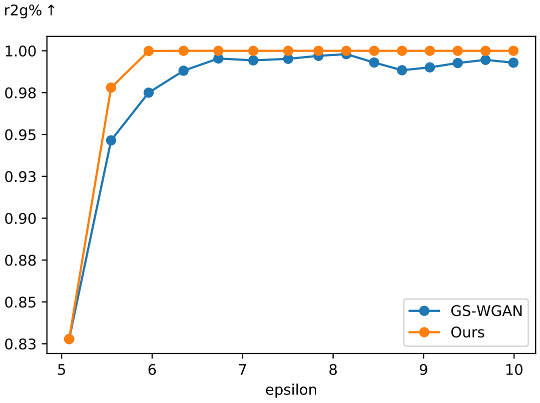

Figure 10. Comparing r2g% on MNIST

			gen2real ↑		real2gen ↑	
Method	IS ↑	FID ↓	MLP	CNN	MLP	CNN
Baseline	1.85	297.35	0.68	0.66	0.66	0.60
w/o g2r	1.80	136.68	0.78	0.82	**0.99**	0.97
w/o r2g	1.77	135.29	0.78	0.83	**0.99**	0.95
w/o seq	1.77	325.93	0.57	0.54	0.80	0.71
w/o init	**1.96**	**133.00**	0.70	0.74	**0.99**	**0.98**
Full	1.90	139.99	**0.80**	**0.85**	**0.99**	**0.98**

Table 3. Ablation studies on CelebA. Each sub-component of our final design has demonstrated its expected effectiveness.

E. Ablation Studies on MNIST and CelebA

In addition to the ablation studies on Fashion MNIST in the paper, here we present the studies on MNIST (Tab. 2) and CelebA (Tab. 3). Each sub-component of our method is designed for a clear purpose, *i.e.*, to improve on quality, or standard utility, or reversed utility. The ablation studies on all three datasets have shown reasonable results to prove their effectiveness.

For example, the auxiliary classifier is designed in a dual-purpose way that incorporates both gen2real and real2gen components. This is because we find learning from both real and fake data sources would enable us to feedback the generator with learning signals about the bilateral transferability during training. This could further improve on the g2r% and r2g% performances compared to the single-purpose design that without either gen2real or real2gen component. The effectiveness of the dual-purpose design is proved in Tab. 2 and Tab. 3 where the full design outperforms the "w/o g2r" and "w/o r2g" designs in terms of g2r% and r2g% performances. The only "exception" is that on MNIST, all three versions achieve the same r2g% of 1.00 using either MLP or CNN as the classifier, due to the simplicity of the task. It is also noteworthy that both "w/o g2r" and "w/o r2g" designs outperforms the baselines in terms of g2r% and r2g% for quite a margin, especially on the more challenging CelebA dataset. This proves the effectiveness of the auxiliary classifier in general.

In addition, the sequential training strategy for integrating the two data sources is extremely crucial to our method's success. This is because real and fake data are from very different distributions and have quite distinct features for each class label. Thus, mixing the losses from real and fake sources into the same equation would result in noisy gradients that confuse the classifier during its updates. As shown in Tab. 2 and Tab. 3, all metrics including the quality ones and the utility ones experience a dramatic downgrade in performance without this sequential training design.

Finally, as shown in Tab. 2 and Tab. 3, without the re-initialisation steps, the g2r% performance decreases significantly for both datasets, while all remaining metrics: IS, FID, and r2g% show comparable performances and are not clearly impacted by the ablation of re-initialisation component. This is reasonable because the re-initialisation of the auxiliary classifier after each generator update is designed to improve on the standard utility measured by g2r%, since re-initialisation mimics the gen2real evaluation process, and allows the classifier to be trained from scratch using the fake data synthesised by the specific version of generator during each iteration. Thus, the classifier is a better representative of the specific generator, and as a result, returns better feedback back to the generator to improve on its g2r% performance.

F. Additional experiments on CIFAR-10

We also conducted preliminary experiments on CIFAR-10. The results (Tab. 4) were far from optimal, but had already shown clear improvements on the baseline [2]. Further progresses might be made by employing more recent GAN architectures. The current choices of DCGAN & BigGAN were inherited from previous work [2] for fair comparisons.

			gen2real ↑		real2gen ↑	
Method	IS ↑	FID ↓	MLP	CNN	MLP	CNN
GS-WGAN [2]	1.01	435.83	0.13	0.11	0.10	0.10
Ours	**1.73**	**236.10**	**0.16**	**0.16**	**0.55**	**0.59**

Table 4. CIFAR-10 Preliminary Results.

G. Effect of auxiliary classifier on DP mechanism

We plotted the average gradient norm (before clipping) for both our method and the baseline [2] that does not have the auxiliary classifier.

$$\mathcal{M}_{\sigma,\zeta}(\boldsymbol{g}) = \text{Clip}(\boldsymbol{g}, \zeta) + \mathcal{N}(0, \sigma^2\zeta^2\boldsymbol{I}^2). \tag{1}$$

As in Eq. (1), the DP mechanism has two components that greatly destroy gradient information during GAN training: clipping and adding random noise. The clip bound ζ was analytically determined to be 1 due to the use of WGAN-GP softly regularise the gradient norms to be within one. However, without auxiliary classifier, the optimised gradients from the discriminator converged very slow and were consistently having norms above 1 as shown in Fig. 11. With the auxiliary classifier, the combined gradients from both networks converged much faster, and were consistently having a smaller norm (mostly below 1), therefore clipping destroyed significantly less gradient information.

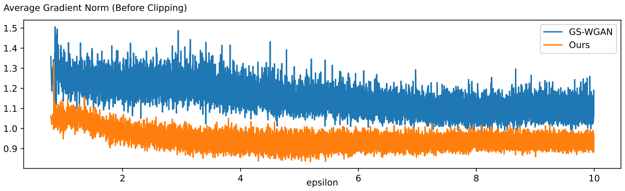

Figure 11. Average gradient norm (before clipping) for each privacy costs during the GAN training process on FashionMNIST.

References

[1] Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. Don't generate me: Training differentially private generative models with Sinkhorn divergence. *NeurIPS*, pages 12480–12492, 2021. 2

[2] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. GS-WGAN: A gradient-sanitized approach for learning differentially private generators. *NeurIPS*, pages 12673–12684, 2020. 1, 2, 5

[3] Boxin Wang, Fan Wu, Yunhui Long, Luka Rimanic, Ce Zhang, and Bo Li. DataLens: Scalable privacy preserving training via gradient compression and aggregation. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2146–2168, 2021. 2