Overview of the Appendix

The Appendix is organized as follows:

- Appendix A introduces the general experimental setup.
- Appendix B introduces the details of dynamic sparse training.
- Appendix C shows detailed algorithms, i.e., DDA, ADAPT$_{\text{relax}}$, and ADAPT$_{\text{strict}}$.
- Appendix D shows the BR evolution during training for ADAPT.
- Appendix E shows additional results, including IS and FID of test sets of the main paper.
- Appendix F shows detailed FLOPs comparisons of sparse training methods.

A Experimental setup

In this section, we explain the training details used in our experiments. Our code is mainly based on the original code of ITOP [48] and GAN ticket [8].

A.1 Architecture details

We use ResNet-32 [25] for the CIFAR-10 dataset and ResNet-48 for the STL-10 dataset. See Table 4 and Table 5 for detailed architectures. We apply spectral normalization for all fully-connected layers and convolutional layers of the discriminators.

For BigGAN architecture, we use the implementation used in DiffAugment [82].[2]

A.2 Datasets

We use the training set of CIFAR-10, the unlabeled partition of STL-10, and the training set of TinyImageNet for GAN training. Training images are resized to 32×32, 48×48, 64×64 for CIFAR-10, STL-10, and TinyImageNet datasets, respectively. Augmentation methods for both datasets are random horizontal flip and per-channel normalization.

A.3 Training hyperparameters

SNGAN on the CIFAR-10 and STL-10 datasets. We use a learning rate of 2×10^{-4} for both generators and discriminators. The discriminator is updated five times for every generator update. We adopt Adam optimizer with $\beta_1 = 0$ and $\beta_2 = 0.9$. The batch size of the discriminator and the generator is set to 64 and 128, respectively. Hinge loss is used following [6, 8]. We use exponential moving average (EMA) [78] with $\beta = 0.999$. The generator is trained for a total of 100k iterations.

BigGAN on the CIFAR-10 dataset. We use a learning rate of 2×10^{-4} for both generators and discriminators. The discriminator is updated four times for every generator update. We adopt Adam optimizer with $\beta_1 = 0$ and $\beta_2 = 0.999$. The batch size of both the discriminator and the generator is set to 50. Hinge loss is used following [6, 76]. We use EMA with $\beta = 0.9999$. The generator is trained for a total of 200k iterations.

BigGAN on the TinyImageNet dataset. We use DiffAug [82] to augment the input. The learning rate of the discriminator and the generator are set to 4×10^{-4} and 1×10^{-4}, respectively. The discriminator is updated one time for every generator update. We adopt Adam optimizer with $\beta_1 = 0$ and $\beta_2 = 0.999$. The batch size of both the discriminator and the generator is set to 256. Hinge loss is used following [6, 76]. We use EMA with $\beta = 0.9999$. The generator is trained for a total of 200k iterations.

A.4 Evaluation metric

SNGAN on the CIFAR-10 and the STL-10 datasets. We compute Fréchet inception distance (FID) and Inception score (IS) for 50k generated images every 5000 iterations. Best FID and IS are

[2] `https://github.com/mit-han-lab/data-efficient-gans/tree/master/DiffAugment-biggan-cifar`.

reported. For the CIFAR-10 dataset, we report both FID for the training set and test set, whereas, for the STL-10 dataset, we report the FID of the unlabeled partition.

BigGAN on the CIFAR-10 and the TinyImageNet dataset. We compute Fréchet inception distance (FID) and Inception score (IS) for 10k generated images every 5000 iterations. Best FID and IS are reported.

B Dynamic sparse training details

B.1 How the generator performs DST

In this section, we explain how the generator performs DST below. Note that the generator performs the same for `SDST` and `ADAPT`.

Sparsity distribution at initialization. Following RigL and ITOP [15, 48], only parameters of fully connected and convolutional layers will be pruned. At initialization, we use the commonly adopted *Erdős-Rényi-Kernel* (ERK) strategy [15, 13, 48] to allocate higher sparsity to larger layers. Specifically, the sparsity of convolutional layers l is scaled with $1-\frac{n^{l-1}+n^l+w^l+h^l}{n^{l-1}n^l w^l h^l}$, where n^l denotes the number of channels of layer l while w^l and h^l are the widths and the height of the corresponding kernel in that layer. For fully connected layers, *Erdős-Rényi* (ER) strategy is used, where the sparsity is scaled with $1-\frac{n^{l-1}+n^l}{n^{l-1}n^l}$.

Update schedule. The update schedule controls how many connections are adjusted per DST operation. It can be specified by the number of training iterations between sparse connectivity updates ΔT_G, the initial fraction of connections adjusted γ, and decaying schedule $f_{\text{decay}}(\gamma, T)$ for γ.

Drop and grow. After ΔT_G training iterations, we update the mask $\boldsymbol{m}_G$ by dropping/pruning $f_{\text{decay}}(\gamma, T)\,|\boldsymbol{\theta}_G|\,d_G$ number of connections with the lowest magnitude, where $|\boldsymbol{\theta}_G|$, d_G are the number of parameters and target density for the generator, $f_{\text{decay}}(\gamma, T)$ is the update schedule. Right after the connection drop, we regrow the same amount of connections.

For the growing criterion, we test both random growth ◆ `SET` [56, 48] and gradient-based growth ● `RigL` [15]. Concretely, gradient-based methods find newly-activated connections θ with the highest gradient magnitude $\left|\frac{\partial \mathcal{L}}{\partial \theta}\right|$, while random-based methods explore connections in a random fashion. All the newly-activated connections are set to 0. One thing that should be noticed is that while previous works consider layer-wise connections drop and growth, we grow and drop connections globally as it grants more flexibility to the DST method.

EMA for sparse GAN. EMA [78] is well-known for its ability to alleviate the non-convergence of GAN. We also implement EMA for sparse GAN training. Specifically, we zero out the moving average of dropped weights whenever there is a mask change.

B.2 DST hyperparameters for the generator

We use the same hyper-parameters for `SDST` and `ADAPT`. The initial γ is set to 0.5, and we use a cosine annealing function f_{decay} following RigL and ITOP.

SNGAN on the CIFAR-10 and the STL-10 datasets. The connection update frequency of the generator ΔT_G is set to 500 and 1000 for the CIFAR-10 dataset and STL-10 dataset, respectively.

BigGAN on the CIFAR-10 and the TinyImageNet dataset. The connection update frequency of the generator ΔT_G is set to be 1000.

B.3 Density dynamic adjust (DDA) hyper-parameters

In this section, we provide hyper-parameters used in subsection 5.3. We set $d_D = 2000$, $\Delta T_D = 0.05$, $[B_-, B_+] = [0.5, 0.65]$. Time-averaged BR over 1000 iterations is used as the indicator.

B.4 DST hyperparameters for the discriminator in ADAPT

We use a constant BR interval $[B_-, B_+] = [0.45, 0.55]$ for SNGAN experiments and BigGAN on the CIFAR-10 dataset. We set the BR interval $[B_-, B_+] = [0.3, 0.4]$ for BigGAN on the TinyImageNet

Table 4: ResNet architecture for CIFAR-10.

(a) Generator	(b) Discriminator
$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$	image $x \in [-1, 1]^{32\times32\times3}$
dense, $4 \times 4 \times 256$	ResBlock down 128
ResBlock up 256	ResBlock down 128
ResBlock up 256	ResBlock down 128
ResBlock up 256	ResBlock down 128
BN, ReLU, 3×3 conv, Tanh	ReLU 0.1
	Global sum pooling
	dense $\rightarrow$ 1

Table 5: ResNet architecture for STL-10.

(a) Generator	(b) Discriminator
$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$	image $x \in [-1, 1]^{48\times48\times3}$
dense, $6 \times 6 \times 512$	ResBlock down 64
ResBlock up 256	ResBlock down 128
ResBlock up 128	ResBlock down 256
ResBlock up 64	ResBlock down 512
BN, ReLU, 3×3 conv, Tanh	ResBlock down 1024
	ReLU 0.1
	Global avg pooling
	dense $\rightarrow$ 1

since it uses DiffAug. Time-averaged BR over 1000 iterations is used as the indicator. Density increment Δd is set to be 0.05, 0.025, and 0.05 for SNGAN (CIFAR-10), SNGAN (STL-10), and BigGAN (CIFAR-10), respectively. We use the same setting used in subsection B.2 for the generator.

Hyper-parameters for `ADAPT`$_{\text{relax}}$. The density update frequency of the discriminator ΔT_D is 1000, 2000, 5000, and 10000 iterations for SNGAN (CIFAR-10), SNGAN (STL-10), BigGAN (CIFAR-10), and BigGAN (TinyImageNet), respectively.

Hyper-parameters for `ADAPT`$_{\text{strict}}$. The density/connections update frequency of the discriminator ΔT_D is 2000, 2000, 5000, and 10000 iterations for SNGAN (CIFAR-10), SNGAN (STL-10), BigGAN (CIFAR-10), and BigGAN (TinyImageNet), respectively.

Note that we compute BR for every iteration to visualize the BR evolution, whereas one should note that such computational cost can be greatly decreased if BR is computed every few iterations.

C Algorithms

In this section, we present the detailed algorithms for `DDA`, `ADAPT`$_{\text{relax}}$, and `ADAPT`$_{\text{strict}}$.

C.1 Dynamic adjust algorithm

We first present `DDA` in Algorithm 1.

Algorithm 1 Dynamic density adjust (`DDA`) for the discriminator.

Require: Generator G, discriminator D, BR upper bound B_+ and lower bound B_-, DA interval ΔT_D, density increment Δd, current training iteration t.

```
if t mod ΔT_D == 0 then
    Compute time-averaged BR with Equation 3
    if BR > B_+ then
        Increase the density of discriminator from d_D to d_D + Δd.
    else if BR < B_- then
        Decrease the density of discriminator from d_D to d_D − Δd.
    end if
end if
```

C.2 Relaxed balanced dynamic sparse training algorithm

Details of `ADAPT`$_{\text{relax}}$ algorithm is presented in Algorithm 2.

C.3 Strict balanced dynamic sparse training algorithm

Details of `ADAPT`$_{\text{strict}}$ algorithm is presented in Algorithm 3.

Algorithm 2 Relaxed balanced dynamic sparse training ($\texttt{ADAPT}_{\text{relax}}$) for GANs.

Require: Generator G, discriminator D, total number of training iterations T, number of training steps for discriminator in each iteration N, discriminator adjustment interval ΔT_D, DST interval for the generator ΔT_G, density increment Δd, target generator density d_G, BR upper bound B_+ and lower bound B_-.

Set initial discriminator density $d_D = d_G$
for t in $[1, \cdots, T]$ **do**
  for n in $[1, \cdots, N]$ **do**
    Compute the loss of discriminator $\mathcal{L}_D(\boldsymbol{\theta}_D)$
    $\mathcal{L}_D(\boldsymbol{\theta}_D).backward()$
  end for
  if $t \mod \Delta T_D == 0$ **then**
    Compute the loss of generator $\mathcal{L}_G(\boldsymbol{\theta}_G)$
    $\mathcal{L}_G(\boldsymbol{\theta}_G).backward()$
    Compute time-averaged BR with Equation 3
    if BR $> B_+$ **then**
      Increase the density of discriminator from d_D to $\min(100\%, d_D + \Delta d)$.
    else if BR $< B_-$ **then**
      Decrease the density of discriminator from d_D to $\max(0\%, d_D - \Delta d)$.
    end if
  end if
  if $t \mod \Delta T_G == 0$ **then**
    Apply DST to G
  end if
end for

Algorithm 3 Strict balanced dynamic sparse training ($\texttt{ADAPT}_{\text{strict}}$) for GANs.

Require: Generator G, discriminator D, total number of training iterations T, number of training steps for discriminator in each iteration N, given maximal density of discriminator d_{max}, discriminator adjustment interval ΔT_D, DST interval for the generator ΔT_G, density increment Δd, target generator density d_G, BR upper bound B_+ and lower bound B_-.

Set initial discriminator density $d_D = d_G$
for t in $[1, \cdots, T]$ **do**
  for n in $[1, \cdots, N]$ **do**
    Compute the loss of discriminator $\mathcal{L}_D(\boldsymbol{\theta}_D)$
    $\mathcal{L}_D(\boldsymbol{\theta}_D).backward()$
  end for
  if $t \mod \Delta T_D == 0$ **then**
    Compute the loss of generator $\mathcal{L}_G(\boldsymbol{\theta}_G)$
    $\mathcal{L}_G(\boldsymbol{\theta}_G).backward()$
    Compute time-averaged BR with Equation 3
    if BR $> B_+$ and $d_D < d_{\text{max}}$ **then**
      Increase the density of discriminator from d_D to $\min(d_{\text{max}}, d_D + \Delta d)$.
    else if BR $> B_+$ and $d_D == d_{\text{max}}$ **then**
      Apply DST to D
    else if BR $< B_-$ **then**
      Decrease the density of discriminator from d_D to $\max(0\%, d_D - \Delta d)$.
    end if
  end if
  if $t \mod \Delta T_G == 0$ **then**
    Apply DST to G
  end if
end for

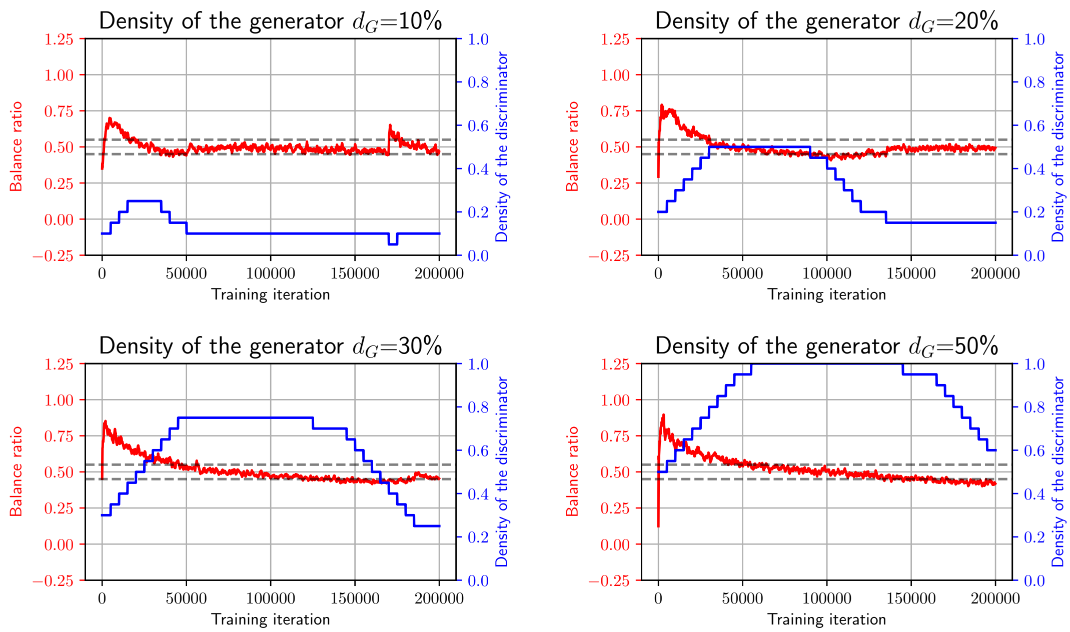

Figure 5: Balance ratio and discriminator density evolution during training for $\texttt{ADAPT}_{\text{relax}}$ on BigGAN (CIFAR-10). Dashed lines represent BR values of 0.45 and 0.55.

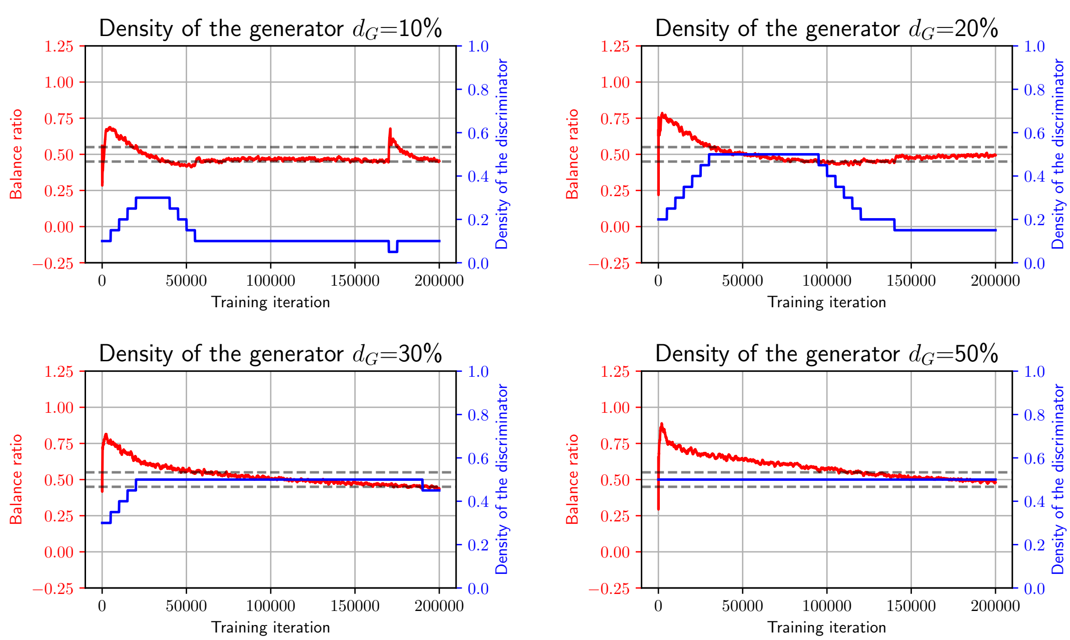

Figure 6: Balance ratio and discriminator density evolution during training for $\texttt{ADAPT}_{\text{strict}}$ on BigGAN (CIFAR-10). Dashed lines represent BR values of 0.45 and 0.55.

D ADAPT balance ratio evolution

In this section, we show that ADAPT methods are able to maintain a BR throughout training. We show the time evolution of BR and discriminator density for BigGAN on the CIFAR-10 dataset.

Results of $\texttt{ADAPT}_{\text{relax}}$ and $\texttt{ADAPT}_{\text{strict}}$ are shown in Figure 5 and Figure 6. It clearly illustrates the ability of ADAPT to keep the BR controlled during GAN training.

Table 6: FID (↓) of different sparse training methods along with post-hoc pruning baseline **with no constraint on the density of the discriminator**. Best results are in **bold**; second-best results are underlined.

Dataset	CIFAR-10 (SNGAN)				STL-10 (SNGAN)				CIFAR-10 (BigGAN)			
Generator density	10%	20%	30%	50%	10%	20%	30%	50%	10%	20%	30%	50%
(Dense Baseline)	10.74				29.71				8.11			
Post-hoc pruning	20.89	14.07	12.99	11.90	57.28	37.12	31.98	**29.70**	15.44	10.84	9.65	8.77
STATIC-Balance	26.75	19.04	15.05	12.24	48.18	44.67	41.73	37.68	16.98	12.81	10.33	8.47
STATIC-Strong	26.79	19.65	14.38	11.91	52.48	43.85	42.06	37.47	23.48	14.26	11.19	8.64
◆ SDST-Balance-SET	26.23	17.79	13.21	11.79	56.41	46.58	39.93	30.37	12.41	9.87	9.13	**8.01**
◆ SDST-Strong-SET	16.49	13.36	**11.68**	10.68	67.37	49.96	37.99	31.08	18.94	9.64	8.75	8.36
● SDST-Balance-RigL	27.06	16.36	14.00	12.28	43.08	33.90	31.83	30.30	12.45	9.42	8.86	8.03
● SDST-Strong-RigL	17.02	13.86	12.51	11.35	53.65	33.25	**31.41**	30.18	10.58	9.11	8.69	8.33
ADAPT$_{\text{relax}}$ (Ours)	**14.19**	**13.19**	12.38	**10.60**	**35.98**	**33.06**	31.71	29.96	**10.19**	**8.56**	**8.36**	8.22

E More experiment results

E.1 IS and FID for the CIFAR-10 dataset

In this section, we present corresponding IS scores results for Table 1 and Table 2. The results are shown in Table 8 and Table 9, respectively. We also include FID results of CIFAR-10 test set in Table 10.

E.2 Naively applying DST to both the generator and the discriminator

In this section, we follow STU-GAN to compare the baseline where applying DST on both generators and discriminators. We name it `DST-bothGD`.

We test on SNGAN (CIFAR-10) with $\Delta T_D = 1000$, $\Delta T_G = 500$, and $\gamma = 0.5$. Note that we use the balance strategy where $d_G = d_D$. The reason is that the strong strategy uses a dense discriminator, and it does not make sense to apply DST to a dense network.

We show the results in Table 7. It shows that it generates unstable results and consistenly performs worse than `SDST-Strong`. So we do not compare such baseline in the main body of the paper.

E.3 Post-hoc pruning baseline

In this section, we compare different sparse training methods with post-hoc magnitude pruning [61] baseline. Magnitude pruning involves first training a dense generator, then pruning its weights globally based on their magnitudes. The pruned generator is then fine-tuned with the dense discriminator. We perform additional fine-tuning for 50% of the original total iterations. Results are presented in Table 6.

Our experimental results clearly demonstrate the advantages of dynamic sparse training over post-hoc magnitude pruning. The latter typically requires around 150% normalized training FLOPs, while DST methods constantly achieve comparable or better performance with significantly reduced computational cost.

F A detailed comparison of training costs

In this section, we include the detailed computational cost of all sparse training methods. More specifically, we take into account the density redistribution over different layers in this section. Also, we make an assumption that the computational overhead introduced by computing BR can be neglected.[3]

Here we provide training costs for the **strict** setting in Table 12.

[3]This is true if we compute BR less frequently.

Table 7: FID (↓) of different sparse training methods on CIFAR-10 datasets with no constraint on the density of the discriminator. Best results are in **bold**; second-best results are <u>underlined</u>.

Dataset	CIFAR-10			
Generator density	10%	20 %	30 %	50 %
(Dense Baseline)	10.74			
Static-Balance	26.75	19.04	15.05	12.24
Static-Strong	26.79	19.65	14.38	11.91
◆ DST-bothGD-SET	20.57	14.90	12.58	11.86
● DST-bothGD-RigL	31.95	17.99	13.24	12.47
◆ SDST-Balance-SET	26.23	17.79	13.21	11.79
◆ SDST-Strong-SET	<u>16.49</u>	<u>13.36</u>	**11.68**	<u>10.68</u>
● SDST-Balance-RigL	27.06	16.36	14.00	12.28
● SDST-Strong-RigL	17.02	13.86	12.51	11.35
$\text{ADAPT}_{\text{relax}}$ (Ours)	**14.19**	**13.19**	<u>12.38</u>	**10.60**

Table 8: IS (higher is better) of different sparse training methods. There is no constraint on the density of the discriminator, i.e., $d_{\text{max}} = 100\%$.

Dataset	SNGAN(CIFAR-10)				SNGAN(STL-10)				BigGAN(CIFAR-10)				BigGAN(TinyImageNet)			
Generator density	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %
(Dense Baseline)	8.48				9.16				8.99				14.65			
Static-Balance	7.24	7.83	8.06	8.38	7.94	8.19	8.44	8.69	7.99	8.24	8.68	8.90	10.65	12.28	13.41	13.57
Static-Strong	7.52	8.03	8.32	8.45	7.70	8.22	8.35	8.70	7.75	8.13	8.52	8.99	10.45	12.56	13.61	13.73
◆ SDST-Balance-SET	7.28	7.89	8.22	8.38	8.43	8.92	9.26	9.31	8.62	8.67	8.82	8.98	11.75	12.60	12.30	12.21
◆ SDST-Strong-SET	8.37	8.54	8.57	8.60	7.65	8.53	9.39	9.21	8.16	8.78	8.85	9.06	12.75	12.84	12.46	13.73
● SDST-Balance-RigL	7.19	7.94	8.18	8.34	8.98	9.07	9.12	9.28	8.64	8.71	8.91	8.93	12.67	13.32	13.18	13.61
● SDST-Strong-RigL	8.32	8.52	8.59	8.57	8.15	9.10	9.16	9.17	8.65	8.72	8.97	9.00	13.32	13.35	13.60	14.47
$\text{ADAPT}_{\text{relax}}$ (Ours)	8.42	8.44	8.54	8.60	9.08	9.29	9.06	9.26	8.74	9.07	8.98	9.00	13.09	13.57	13.68	15.77

Table 9: IS (higher is better) of different sparse training methods. The density of the discriminator is constrained to be lower than $d_{\text{max}} = 50\%$.

Dataset	SNGAN(CIFAR-10)				SNGAN(STL-10)				BigGAN(CIFAR-10)				BigGAN(TinyImageNet)			
Generator density	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %
(Dense Baseline)	8.48				9.16				8.99				14.65			
Static-Balance	7.24	7.83	8.06	8.38	7.94	8.19	8.44	8.69	7.99	8.24	8.68	8.90	10.65	12.28	13.41	13.57
Static-Strong	7.85	8.14	8.31	8.38	7.89	8.22	8.38	8.69	7.75	8.03	8.52	8.90	9.99	11.61	13.77	13.57
◆ SDST-Balance-SET	7.28	7.89	8.22	8.38	8.43	8.92	9.26	9.31	8.62	8.67	8.82	8.98	11.75	12.60	12.30	12.21
◆ SDST-Strong-SET	8.33	8.53	8.40	8.38	8.50	8.77	9.46	9.26	8.55	8.77	8.84	8.98	12.00	12.87	12.16	12.21
● SDST-Balance-RigL	7.19	7.94	8.18	8.34	8.98	9.07	9.12	9.28	8.64	8.71	8.91	8.93	12.67	13.32	13.18	13.61
● SDST-Strong-RigL	8.24	8.48	8.37	8.34	8.28	9.05	9.11	9.28	8.61	8.83	8.84	8.93	12.04	12.66	13.57	13.61
$\text{ADAPT}_{\text{strict}}$ (Ours)	8.27	8.36	8.48	8.47	8.98	9.17	9.20	9.19	8.90	8.89	8.92	9.10	13.85	13.61	14.05	14.40

Table 10: FID of test set (↓) of different sparse training methods on SNGAN (CIFAR-10) dataset. Best results are in **bold**; second-best results are <u>underlined</u>.

Maximal discriminator density d_{max}	100 %				50 %			
Generator density	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %
(Dense Baseline)	13.32							
Static-Balance	29.56	21.79	17.80	14.94	29.56	21.79	17.80	14.94
Static-Strong	29.50	22.45	17.12	14.58	24.62	19.43	16.32	14.94
◆ SDST-Balance-SET	28.84	20.31	15.95	14.35	28.84	20.31	15.95	**14.35**
◆ SDST-Strong-SET	<u>19.16</u>	<u>16.12</u>	**14.45**	<u>13.50</u>	18.38	**15.33**	**14.78**	**14.35**
● SDST-Balance-RigL	29.77	19.02	16.68	15.05	29.77	19.02	16.68	15.05
● SDST-Strong-RigL	19.72	16.50	15.20	14.09	<u>17.92</u>	<u>15.51</u>	15.52	15.05
$\text{ADAPT}_{\text{relax}}$ (Ours)	**16.82**	**15.85**	<u>15.14</u>	**13.37**	-	-	-	-
$\text{ADAPT}_{\text{strict}}$ (Ours)	-	-	-	-	**17.19**	15.57	<u>14.92</u>	<u>14.80</u>

Table 11: FID of test set (↓) of different sparse training methods on BigGAN (CIFAR-10) dataset. Best results are in **bold**; second-best results are underlined.

Maximal discriminator density $d_{\max}$	**100 %**				**50 %**			
Generator density	10%	20 %	30 %	50 %	10%	20 %	30 %	50 %
(Dense Baseline)	10.36							
Static-Balance	19.58	15.63	13.21	10.92	19.58	15.63	13.21	10.92
Static-Strong	26.08	15.82	13.47	10.95	22.04	16.39	13.73	10.92
◆ SDST-Balance-SET	14.90	12.77	11.82	**10.68**	14.90	12.77	11.82	10.68
◆ SDST-Strong-SET	21.63	11.92	11.27	10.75	14.53	11.83	10.96	10.68
● SDST-Balance-RigL	14.86	12.03	11.30	**10.68**	14.86	12.03	11.30	10.68
● SDST-Strong-RigL	13.35	11.58	11.00	10.88	12.59	12.03	**10.89**	10.68
ADAPT$_{\text{relax}}$ (Ours)	**12.71**	**11.02**	**10.62**	10.80	-	-	-	-
ADAPT$_{\text{strict}}$ (Ours)	-	-	-	-	**11.83**	**11.22**	10.92	**10.33**

Table 12: Normalized training FLOPs (↓) of different sparse training methods. **The density of the discriminator is constrained to be lower than 50%.**

Dataset	**CIFAR-10 (SNGAN)**				**STL-10 (SNGAN)**				**CIFAR-10 (BigGAN)**				**TinyImageNet (BigGAN)**			
Generator density	10%	20%	30%	50%	10%	20%	30%	50%	10%	20%	30%	50%	10%	20%	30%	50%
(Dense Baseline)	100% (2.67×10^{17})				100% (3.94×10^{17})				100% (6.81×10^{17})				100% (9.85×10^{17})			
Static-Balance	8.97%	17.08%	26.25%	47.25%	27.30%	47.14%	59.22%	73.35%	9.79%	19.02%	28.66%	49.03%	23.25%	44.87%	60.91%	79.29%
Static-Strong	30.89%	33.58%	37.17%	47.25%	70.65%	71.48%	72.14%	73.35%	42.66%	43.69%	45.10%	49.03%	41.52%	55.03%	66.29%	79.29%
◆ SDST-Balance-SET	9.78%	18.91%	28.35%	48.44%	27.55%	47.60%	60.17%	75.38%	10.35%	20.12%	29.96%	49.82%	21.13%	37.06%	48.83%	65.58%
◆ SDST-Strong-SET	31.87%	35.51%	39.53%	48.44%	70.95%	71.97%	73.07%	75.38%	43.25%	44.80%	46.42%	49.82%	39.28%	47.31%	54.11%	65.58%
● SDST-Balance-RigL	10.71%	17.43%	25.66%	43.56%	29.51%	50.41%	63.34%	79.03%	9.92%	19.30%	28.90%	48.31%	24.97%	43.86%	57.26%	76.75%
● SDST-Strong-RigL	31.22%	33.93%	36.63%	43.56%	72.95%	75.05%	76.42%	79.03%	42.80%	44.08%	45.37%	48.31%	43.76%	53.71%	63.05%	76.75%
ADAPT$_{\text{strict}}$ (Ours)	24.23%	27.55%	31.70%	37.83%	50.91%	70.18%	75.99%	80.68%	10.32%	23.69%	31.54%	33.83%	34.42%	51.68%	62.34%	77.46%