GLeaD: Improving GANs with A Generator-Leading Task
– *Supplementary Material* –

Qingyan Bai[1*] Ceyuan Yang[2] Yinghao Xu[3*] Xihui Liu[4] Yujiu Yang[1†] Yujun Shen[5]

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University [2]Shanghai AI Laboratory
[3]The Chinese University of Hong Kong [4]The University of Hong Kong [5]Ant Group

Overview

This supplementary material is organized as follows. Sec. A introduces the detailed discriminator network structure of GLeaD. Sec. B provides comparisons on the computational between our method and the baseline [1].

A. Discriminator Network Structure

Recall that, our D concludes a backbone D_{enc}, a head predicting realness scores, and a decoder h for predicting representative features f and w. Taking images whose resolution are 256×256 as an instance, the backbone D_{enc} is first employed to extract features from the input image. The very last feature map of 4×4 is sent to the scoring head to extract the realness score while the multi-level feature maps are sent to the decoder h to predict the representative features adequate for G to reconstruct the original images. As described in the submission, the representative features consist of latent codes w and the spatial representations f, which concludes a low-level representation and a high-level representation. Recall that, these spatial representations will be sent to the fixed generator to serve as the basis of the reconstruction and will be modulated by latent codes to predict the final results. We illustrate the architectures of the three aforementioned components of D in Tab. S1, Tab. S2, and Tab. S3, respectively.

B. Computational Costs

We first compute the discriminator parameter amounts of the baseline and our method. As in Tab. S4, our method merely brings 7.4% additional parameters over baseline, which is brought by the proposed lightweight design of h composed of 1×1 convolutions. Then we compare the inference time of the discriminators with a single A6000 GPU. At last, we make comparisons on the training time. We separately train the baseline model [1] and our model with 8 A100 GPUs on LSUN Church and record how much time the training costs. From the numbers in Tab. S4, we

Table S1. Network structure of the backbone D_{enc}. The output size is with order $\{C \times H \times W\}$, where C, H, and W respectively denotes the channel dimension, height and weight of the output.

Stage	Block	Output Size
input	-	$3 \times 256 \times 256$
block_1	[1×1 Conv, 128 2×3×3 Conv, 128 1×1 Conv, 128 Downsample LeakyReLU, 0.2]	$128 \times 128 \times 128$
block_2	[2×3×3 Conv, 256 1×1 Conv, 256 Downsample LeakyReLU, 0.2]	$256 \times 64 \times 64$
block_3	[2×3×3 Conv, 512 1×1 Conv, 512 Downsample LeakyReLU, 0.2]	$512 \times 32 \times 32$
block_4	[2×3×3 Conv, 512 1×1 Conv, 512 Downsample LeakyReLU, 0.2]	$512 \times 16 \times 16$
block_5	[2×3×3 Conv, 512 1×1 Conv, 512 Downsample LeakyReLU, 0.2]	$512 \times 8 \times 8$
block_6	[2×3×3 Conv, 512 1×1 Conv, 512 Downsample LeakyReLU, 0.2]	$512 \times 4 \times 4$

∗ This work was done during an internship at Ant Group.
† Corresponding author.

Table S2. Network structure of the decoder h predicting the low-level spatial representation, the high-level spatial representation and the 512-channel latent codes. Note that h receives multi-level features as inputs due to its feature pyramid architecture [2]. The output size is with order $\{C \times H \times W\}$.

Stage	Block	Output Size
input	–	$512 \times 32 \times 32$ $512 \times 16 \times 16$ $512 \times 8 \times 8$ $512 \times 4 \times 4$
block_1	[1×1 Conv, 512 Upsample]	$512 \times 8 \times 8$
block_2	[1×1 Conv, 512 Upsample]	$512 \times 16 \times 16$
block_3	[1×1 Conv, 512 Upsample]	$512 \times 32 \times 32$
block_4	[1×1 Conv, 3 2×1×1 Conv, 512 Downsample]	$3 \times 32 \times 32$ $512 \times 32 \times 32$ 512

Table S3. Network structure of the head predicting realness scores which are scalars. The output size is with order $\{C \times H \times W\}$.

Stage	Block	Output Size
input	–	$512 \times 4 \times 4$
block_1	[Mbstd, 1 3×3 Conv, 512 LeakyReLU, 0.2 Downsample FC, 512 LeakyReLU, 0.2 FC, 1]	1

Table S4. Computational cost comparisons.

Method	# params	inference time(s)	training time(h)
Baseline	24.00M	0.0184	43.83
GLeaD	25.77M	0.0219	55.78

can conclude that our method improves the synthesis quality without much additional computational burden.

References

[1] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 1

[2] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2