Rebuttal for "Improving Few-shot Image Generation by Structural Discrimination and Textural Modulation"

Anonymous Author(s)
Submission Id: 316

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability; • **Computing methodologies** → **Computer vision representations**; **Image representations**; **Computer vision representations**; *Neural networks*; Learning .

ACM Reference Format:
Anonymous Author(s). 2023. Rebuttal for "Improving Few-shot Image Generation by Structural Discrimination and Textural Modulation". In *Proceedings of Proceeding of the 28th ACM International Conference on Multimedia (MM '23).* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

To Reviewer wMYJ

Q1: Lacking of limitations or failure cases of the proposed method.

Reply: Thanks. Following your valuable suggestion, here we provide the limitations of our proposed method, which will also be presented in our revised paper.

Limitations. Despite achieving substantial improvements on all evaluated datasets, there remain areas for further improvement in our proposed model Specifically, the model's performance might suffer when generalizing to datasets with significant class variances, such as ImageNet [1]. Moreover, the cross-domain generation capability is still suboptimal, particularly when the domain gap is substantial, like transferring from the human face domain to natural flowers. Finally, the synthesis quality of our model on extremely limited data amounts, such as ont-shot generation tasks, can be further enhanced. These limitations might be approached in the following tow ways: 1) Incorporating various data augmentation techniques (*e.g.,* adaptive data augmentation (ADA) in [6] and differentiable augmentation from [7]) to enlarge the sample amount of one-shot generation tasks. 2) Exploring additional modules to capture more internal distributional information for the generation tasks. Despite these limitations, our model offers promising alternatives to enhance few-shot image generation and downstream classification problems. In future works, we intend to investigate more general approaches for addressing these limitations.

Q2: Include more baselines (*e.g.,* WaveGAN) in the qualitative comparison.

Reply: Thanks. Following your valuable suggestion, we have included additional qualitative results for visual comparison between WaveGAN and our proposed model. These results can be accessed through the provided anonymous link. The new qualitative results demonstrate that the images generated by our proposed model exhibit superior fine-grained semantic details, overall structures, and authenticity compared to those generated by WaveGAN. The inclusion of these qualitative results, in conjunction with the results presented in Fig. 1 and Fig. 4 in our main paper, further highlight the significant improvements in generation quality achieved by our proposed techniques. We plan to include these results as an appendix in the revised version of our manuscript to provide a more comprehensive and conclusive illustration and comparison of our approach.

Q3: Lacking of descriptions of the network structures (*e.g.,* the decoder of the proposed method).

Reply: Thanks. In our implementation, our encoder consists of five convolutional blocks and four wavelet transformation blocks following WaveGAN. Each convolutional block contains one convolution layer, followed by batch normalization and Leaky-Relu activation. Our decoder is symmetrical with four upsampling blocks and one output convolutional layer. Each upsampling block includes upsampling operation followed by one convolutional block. Regarding the discriminator, besides four residual blocks and two fully connected layers like LofGAN and WaveGAN, we include two lightweight convolutional layers to respectively extract Laplacian and frequency representations. Additionally, the network details are provided in this anonymous link (Algorithm 1, Algorithm 2, Algorithm 3 and Algorithm 4). Moreover, these descriptions of our network architectures will be included in the appendix of our revised paper to improve the clarity and reproducibility of our proposed method. Furthermore, our code and pre-trained models will be made publicly available to enable interested readers to reproduce our results.

Hope that the above discussions could address your concerns, please let us know if you have any further questions. Thanks for your effort and constructive suggestions again.

To Reviewer dB6U

Q1: It is important to provide quantitative results of the cross-domain generation results with all dataset combinations.

Reply: Thanks. Following your valuable suggestion, here we provide the quantitative results of the cross-domain generation experiments with all datasets combinations. To be more specific, the model is first trained on one domain (*e.g.,* VGGFace) and then tested on another domain (*e.g.,* Animal Faces), while other settings remained consistent with the main experiments. The following table (Tab. 1) presents the quantitative results. It is evident that the synthesis performance deteriorates when the training and testing data are from different domains, particularly when the domain gap is substantial (*e.g.,* transferring from Flowers to Animal Faces and VGGFace). Nonetheless, our proposed techniques effectively

MM '23, October 29–November 02, 2023, Ottawa, Canada

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

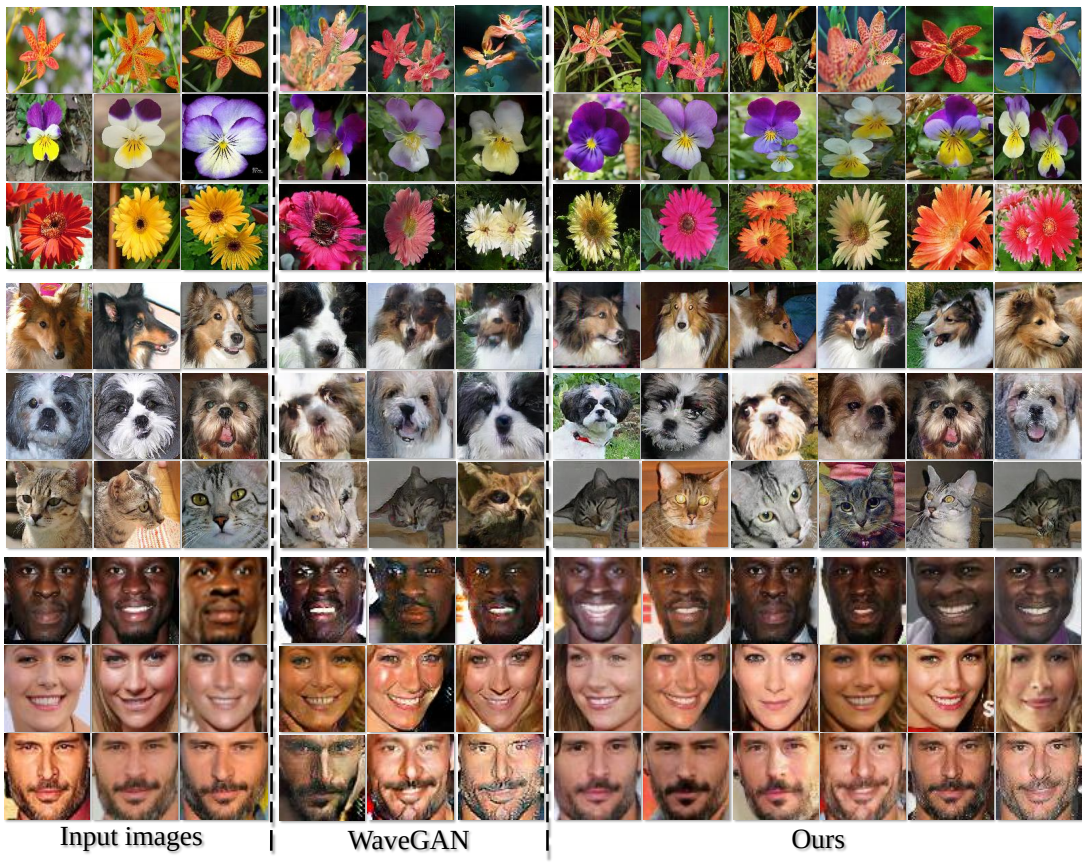

Figure 1: Qualitative comparison results of our method with WaveGAN. Images produced by our model performs better in term of the global structure (*e.g.,* the outline and shape of petals and the coherence of Animal Faces) and semantic variance (*e.g.,* different hair colors of Animal Faces and various expression of Human Face.)

improve the transfer performance under different baselines, further emphasizing the compatibility and flexibility of our method. We will include these quantitative results in the revised version of our manuscript to provide a more comprehensive investigation of the transferability of our proposed method across different domains.

Q2: Typos in the text.

Reply: Thanks. We apologize for the typographical errors in our paper and appreciate your feedback in bringing them to our attention. We have fixed these typos and have thoroughly reviewed the content of our manuscript to ensure its accuracy and clarity. Moving forward, we will continue to proofread our paper to enhance its presentation. For instance, we have corrected the following errors: " In line 214, "techniques is" has been corrected to "techniques are."; In line 479, "G and G" has been corrected to "G and D."; The missing "max()" in Eqs.(4) and Eqs.(5) have been added; In line 857, "Fig.2" has been corrected to "Fig.6." "

Q3: This work is slightly out of the scope of the ACM MM conference.

Reply: Thanks. We submit our manuscript strictly following the author guidelines and instructions provided on the homepage of the ACM MM conference at https://www.acmmm2023.org/cfp/. Our work focuses on generating novel images in few-shot scenarios and aligns with the theme of "understanding multimedia content." This theme seeks novel processing of media-related information in any form that can lead to new ways of interpreting multimedia content. Examples include processing of image, video, audio, music, language, speech, or other sensory modalities, for interpretation, knowledge discovery, and understanding. As you have pointed out, previous works such as [4, 5] have been presented at previous ACM MM conferences. Therefore, we believe that our work is well within the scope of the ACM MM conference. Our work aligns with the conference's goal of exploring innovative approaches to multimedia processing and interpretation. Our paper makes a valuable contribution to the field of few-shot image generation and would be of interest to the conference attendees.

Hope that the above discussions could address your concerns, please let us know if you have any further questions. Thanks for your effort and constructive suggestions again.

To Reviewer bGec

Q1: "StructD" is misspelled on line 454, 456, 544 and 827.

Reply: Thanks. We express our sincere apologies for the typographical errors and greatly appreciate your efforts in bringing them to our attention. We have since rectified the errors you highlighted and conducted multiple rounds of proofreading to ensure the manuscript is free of any other typographical or grammatical mistakes. Moving

Table 1: FID scores of cross-domain experiments. “Source" and “Target" represent the datasets that the model is trained and tested, respectively.

Method	Source	Flowers		Animal Faces		VGGFace	
	Target	Animal Faces	VGGFace	Flowers	VGGFace	Flowers	Animal Faces
LoFGAN [3]	3-shot	158.82	34.44	101.92	26.42	95.04	124.64
+ Ours	3-shot	**150.09**	**30.12**	**99.67**	**23.59**	**93.46**	**119.99**
WaveGAN [3]	3-shot	56.32	16.27	89.87	12.19	68.43	59.05
+ Ours	3-shot	**48.21**	**14.35**	**78.46**	**9.61**	**65.71**	**55.62**

forward, we will continue to proofread our paper to enhance its presentation and clarity.

Q2: Sec 3.3 proposes two methods, *i.e.,* StructD and FreD, but with the name of Structural Discrimination.

Reply: Thanks. Following your valuable suggestion, we have revised the name of Sec 3.3 from “Structural Discrimination" to “Structural and Frequency Discriminator", which illustrates that a structural discriminator and a frequency discriminator are involved in this section. Thank you for pointing this out to make the presentation more clear.

Q3: Explanation on similar pictures in Fig.4 from LoFGAN and the proposed method.

Reply: Thanks. Fig. 4 exhibits some similarities between the results of LoFGAN and our proposed method due to the use of identical input images during testing. As a result, the generated output images may exhibit some common features, such as the arrangement of flower petals, the fur color and texture of animal faces, and the facial expressions of human faces. Nevertheless, it could be seen from Fig. 4 that the images generated by our proposed approach are characterized by a greater degree of photorealism and visual plausibility. For instance, the images generated by our approach exhibit finer-grained details in the flower petals and more realistic-looking eyes in animal faces, in contrast to the presence of unsatisfactory artifacts in the results produced by LoFGAN. Additionally, we will provide the explanation of the similarities observed from the images in Fig. 4 in our revised paper to improve the clarity.

Hope that the above discussions could address your concerns, please let us know if you have any further questions. Thanks for your effort and constructive suggestions again.

To Reviewer NCzv

Q1: The context logic is sometimes weak, making the paper somewhat hard to follow.

Reply: Thanks. Following your valuable suggestion, we have thoroughly reviewed our manuscript and revised it to enhance its overall presentation. Specifically, we in this paper propose textural modulation and structural discrimination to enable more fine-grained semantic injection and provide explicit structural guidance, respectively. Our paper is arranged by relevant descriptions around these proposed techniques. Thanks for your constructive advice and for reminding us that the descriptions and logic of our paper could be further improved. We will continue to engage in ongoing proofreading to further enhance the manuscript’s presentation. Here we list some revisions as follows:

Line 161-164: “To mitigate the aforementioned limitations, we in this paper propose a novel few-shot generation model, dubbed SDTM-GAN, including two key ingredients: structural discrimination (StructD) and textural modulation (TexMod). Fig. 2 presents the concept diagram of StructD and TexMod." → “ In this paper, we present a novel few-shot generation model, named SDTM-GAN, that addresses the aforementioned limitations through the incorporation of two key components: structural discrimination (StructD) and textural modulation (TexMod). "

Line 199-201: “A lightweight discriminator, i.e., StructD, which distinguishes the laplacian representations of real and generated images, is then proposed to explicitly provide structural guidelines to the generator, facilitating the fidelity of global appearance." → “ Then, a lightweight discriminator, i.e., StructD, is proposed to differentiate between the Laplacian representations of real and generated images. In this way, StructD provides explicit structural guidance to the generator, thereby enhancing the fidelity of the global appearance. "

Q2: There are some typos and grammatical errors need to be fixed.

Reply: Thanks. We apologize for the typographical errors in our paper and appreciate your feedback in bringing them to our attention. We have fixed these typos and have thoroughly reviewed the content of our manuscript to ensure its accuracy and clarity. Moving forward, we will continue to proofread our paper to enhance its presentation. For instance, we have corrected the following errors: “ In line 214, "techniques is" has been corrected to "techniques are"; In line 224, “computation cost" has been corrected to “computation costs" In line 479, "G and G" has been corrected to "G and D"; In line 857, "Fig.2" has been corrected to "Fig.6". "

Q3: The introduction to the details of the proposed method is somewhat confused. Provide more details and more information in Fig. 2 as it appears to be a duplicate of Fig. 3.

Reply: Thanks. We include Fig. 2 and Fig. 3 in our manuscript for the following reasons. In the introduction, Fig. 2 presents a conceptual diagram of our proposed textural modulation and structural discriminator, providing a visual representation of the primary components and high-level ideas of our model. The inclusion of Fig. 2 in the introduction is intended to provide a clear and concise overview of the primary components and high-level ideas of our model, facilitating reader comprehension of our paper. Then, in the methodology section (*i.e.,* Section 3), Fig. 3 presents a detailed pipeline of our model, providing a more granular view of the forward process and network computation.

Q4: The results in Tab. 2 are not persuasive as the results are not the best. Apply the proposed method to stronger baselines for comparison.

Algorithm 1 The detailed network architecture of our encoder.

```
(encoder): Encoder(
  (conv1): Conv2dBlock(
    (pad): ReflectionPad2d((2, 2, 2, 2))
    (norm): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(3, 32, kernel_size=(5, 5), stride=(1, 1))
  )
  (pool1): WavePool(
    (LL): Conv2d(32, 64, kernel_size=(2, 2), stride=(2, 2),
        groups=32, bias=False)
    (LH): Conv2d(32, 64, kernel_size=(2, 2), stride=(2, 2),
        groups=32, bias=False)
    (HL): Conv2d(32, 64, kernel_size=(2, 2), stride=(2, 2),
        groups=32, bias=False)
    (HH): Conv2d(32, 64, kernel_size=(2, 2), stride=(2, 2),
        groups=32, bias=False)
  )
  (conv2): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(32, 64, kernel_size=(3, 3), stride=(2, 2))
  )
  (pool2): WavePool(
    (LL): Conv2d(64, 128, kernel_size=(2, 2), stride=(2, 2),
        groups=64, bias=False)
    (LH): Conv2d(64, 128, kernel_size=(2, 2), stride=(2, 2),
        groups=64, bias=False)
    (HL): Conv2d(64, 128, kernel_size=(2, 2), stride=(2, 2),
        groups=64, bias=False)
    (HH): Conv2d(64, 128, kernel_size=(2, 2), stride=(2, 2),
        groups=64, bias=False)
  )
  (conv3): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2)
        )
  )
  (pool3): WavePool2(
    (LL): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
    (LH): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
    (HL): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
    (HH): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
  )
  (conv4): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(128, 128, kernel_size=(3, 3), stride=(2,
        2))
  )
  (pool4): WavePool2(
    (LL): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
    (LH): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
    (HL): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
    (HH): Conv2d(128, 128, kernel_size=(2, 2), stride=(2, 2),
         groups=128, bias=False)
  )
  (conv5): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(128, 128, kernel_size=(3, 3), stride=(2,
        2))
  )
)
```

Algorithm 2 The detailed network architecture of our decoder.

```
(decoder): Decoder(
  (Upsample): Upsample(scale_factor=2.0, mode=nearest)
  (Conv1): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(128, 128, kernel_size=(3, 3), stride=(1,
        1))
  )
  (recon_block1): WaveUnpool(
    (LL): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
    (LH): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
    (HL): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
    (HH): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
  )
  (Conv2): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(128, 128, kernel_size=(3, 3), stride=(1,
        1))
  )
  (recon_block2): WaveUnpool(
    (LL): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
    (LH): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
    (HL): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
    (HH): ConvTranspose2d(128, 128, kernel_size=(2, 2),
        stride=(2, 2), groups=128, bias=False)
  )
  (Conv3): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1)
        )
  )
  (recon_block3): WaveUnpool(
    (LL): ConvTranspose2d(64, 64, kernel_size=(2, 2), stride
        =(2, 2), groups=64, bias=False)
    (LH): ConvTranspose2d(64, 64, kernel_size=(2, 2), stride
        =(2, 2), groups=64, bias=False)
    (HL): ConvTranspose2d(64, 64, kernel_size=(2, 2), stride
        =(2, 2), groups=64, bias=False)
    (HH): ConvTranspose2d(64, 64, kernel_size=(2, 2), stride
        =(2, 2), groups=64, bias=False)
  )
  (Conv4): Conv2dBlock(
    (pad): ReflectionPad2d((1, 1, 1, 1))
    (norm): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=
        True, track_running_stats=True)
    (activation): LeakyReLU(negative_slope=0.2)
    (conv): Conv2d(64, 32, kernel_size=(3, 3), stride=(1, 1))
  )
  (recon_block4): WaveUnpool(
    (LL): ConvTranspose2d(32, 32, kernel_size=(2, 2), stride
        =(2, 2), groups=32, bias=False)
    (LH): ConvTranspose2d(32, 32, kernel_size=(2, 2), stride
        =(2, 2), groups=32, bias=False)
    (HL): ConvTranspose2d(32, 32, kernel_size=(2, 2), stride
        =(2, 2), groups=32, bias=False)
    (HH): ConvTranspose2d(32, 32, kernel_size=(2, 2), stride
        =(2, 2), groups=32, bias=False)
  )
  (Conv5): Conv2dBlock(
    (pad): ReflectionPad2d((2, 2, 2, 2))
    (activation): Tanh()
    (conv): Conv2d(32, 3, kernel_size=(5, 5), stride=(1, 1))
  )
 )
)
```

Algorithm 3 The detailed network architecture of our TexMod.

```
(modulated_spade_layer): ModulatedSPADE(
  (pad): ZeroPad2d(padding=(1, 1, 1, 1), value=0.0)
  (norm): InstanceNorm2d(128, eps=1e-05, momentum=0.1,
       affine=False, track_running_stats=False)
  (activation): LeakyReLU(negative_slope=0.2, inplace=True)
  (conv2d): Conv2d(128, 128, kernel_size=(3, 3), stride=(1,
       1))
  (sigmoid): Sigmoid()
  (mlp_shared): Sequential(
    (0): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1),
         padding=(1, 1))
    (1): ReLU()
  )
  (mlp_gamma): Conv2d(64, 128, kernel_size=(3, 3), stride=(1,
       1), padding=(1, 1))
  (mlp_beta): Conv2d(64, 128, kernel_size=(3, 3), stride=(1,
       1), padding=(1, 1))
  (mlp_shared_2): Sequential(
    (0): Conv2d(128, 64, kernel_size=(3, 3), stride=(1, 1),
         padding=(1, 1))
    (1): ReLU()
  )
  (mlp_gamma_ctx_gamma): Conv2d(64, 128, kernel_size=(3, 3),
       stride=(1, 1), padding=(1, 1))
  (mlp_beta_ctx_gamma): Conv2d(64, 128, kernel_size=(3, 3),
       stride=(1, 1), padding=(1, 1))
  (mlp_gamma_ctx_beta): Conv2d(64, 128, kernel_size=(3, 3),
       stride=(1, 1), padding=(1, 1))
  (mlp_beta_ctx_beta): Conv2d(64, 128, kernel_size=(3, 3),
       stride=(1, 1), padding=(1, 1))
)
```

Reply: Thanks. On the one hand, it is noteworthy that our proposed method achieves state-of-the-art FID scores on all tested datasets under both three-shot and one-shot settings, as evidenced by the best FID scores of 39.51, 26.65, and 3.96 achieved on Flowers, Animal Faces, and VGGFace, respectively. In addition, Tab. 2 demonstrates the compatibility and flexibility of our method by integrating our techniques into different baselines to achieve further performance improvements. On the other hand, as pointed out by the reviewer, our method does not achieve the best LPIPS scores. However, the quantitative results in Tab. 2 demonstrate that our proposed techniques can also improve the LPIPS scores under different baselines and settings. Furthermore, following your valuable suggestions, we are also curious about the results of applying our method to stronger baselines such as SAGE [2] and DiscoFUNIT [4]. Due to limited time constraint, we will add these results in the revised version of our manuscript.

Q5: The ablation study on loss weights (Tab. 6) are not enough. Results under more values and pairs should be presented. It is best to show the visual results as a function of the two coefficients: λ_{str} and λ_{fre}.

Reply: Thanks. In Table 6, we conducted a grid search to iteratively identify the appropriate coefficients for λ_{str} and λ_{fre} within the range of [0.1, 1, 10, 100]. The results showed that setting $\lambda_{str} = \lambda_{fre}$ = 1 was appropriate, as excessively small or large values could either ignore or overemphasize the impact of our proposed techniques. Accordingly, we used $\lambda_{str} = \lambda_{fre} = 1$ in our main experiments. We appreciate the reviewer's suggestion to investigate the upper bound of our proposed model under different values and pairs of λ_{str} and λ_{fre}. To this end, we plan to conduct further investigations within the range of [0.1, 10], as suggested by the reviewer. The results will be presented in our revised paper due to the limited time constraint during the rebuttal. Furthermore, following your valuable suggestion, the quantitative results will be presented in visual format in our revised version, providing readers with a more intuitive and comprehensive understanding of the importance of λ_{str} and λ_{fre}.

Algorithm 4 The detailed network architecture of our discriminator.

```
Discriminator(
  (cnn_f): Sequential(
    (0): Conv2dBlock(
      (pad): ReflectionPad2d((2, 2, 2, 2))
      (conv): Conv2d(3, 64, kernel_size=(5, 5), stride=(1, 1))
    )
    (1): ActFirstResBlock(
      (conv_0): Conv2dBlock(
        (pad): ReflectionPad2d((1, 1, 1, 1))
        (activation): LeakyReLU(negative_slope=0.2)
        (conv): Conv2d(64, 128, kernel_size=(3, 3), stride=(1,
             1))
      )
      (conv_1): Conv2dBlock(
        (pad): ReflectionPad2d((1, 1, 1, 1))
        (activation): LeakyReLU(negative_slope=0.2)
        (conv): Conv2d(128, 128, kernel_size=(3, 3), stride=(1,
             1))
      )
      (conv_s): Conv2dBlock(
        (pad): ZeroPad2d(padding=(0, 0, 0, 0), value=0.0)
        (conv): Conv2d(64, 128, kernel_size=(1, 1), stride=(1,
             1), bias=False)
      )
    )
    (2): ReflectionPad2d((1, 1, 1, 1))
    (3): AvgPool2d(kernel_size=3, stride=2, padding=0)
    {(1) - (3)} X 4
    )
  )
  (cnn_lap): Sequential(
    (0): Conv2dBlock(
      (pad): ReflectionPad2d((2, 2, 2, 2))
      (conv): Conv2d(1, 1024, kernel_size=(5, 5), stride=(1, 1)
           )
    )
  )
  (cnn_adv_lap): Sequential(
    (0): AdaptiveAvgPool2d(output_size=1)
    (1): Conv2dBlock(
      (pad): ZeroPad2d(padding=(0, 0, 0, 0), value=0.0)
      (conv): Conv2d(1024, 85, kernel_size=(1, 1), stride=(1,
           1))
    )
  )
  (cnn_adv): Sequential(
    (0): AdaptiveAvgPool2d(output_size=1)
    (1): Conv2dBlock(
      (pad): ZeroPad2d(padding=(0, 0, 0, 0), value=0.0)
      (conv): Conv2d(1024, 85, kernel_size=(1, 1), stride=(1,
           1))
    )
  )
  (cnn_c): Sequential(
    (0): AdaptiveAvgPool2d(output_size=1)
    (1): Conv2dBlock(
      (pad): ZeroPad2d(padding=(0, 0, 0, 0), value=0.0)
      (conv): Conv2d(1024, 85, kernel_size=(1, 1), stride=(1,
           1))
    )
  )
  (cnn_fre): Sequential(
    (0): AdaptiveAvgPool2d(output_size=1)
    (1): Conv2dBlock(
      (pad): ZeroPad2d(padding=(0, 0, 0, 0), value=0.0)
      (conv): Conv2d(1024, 85, kernel_size=(1, 1), stride=(1,
           1))
    )
  )
)
```

Q6: The meanings of the dotted lines in Fig. 5 are not introduced, which makes it confusing.

Reply: Thanks. The dotted lines in Figure 5 represent the average slope, which illustrates the overall trend of the FID values as the sample size increases. Specifically, the FID score tends to decrease as both the number of training and testing images increase. We apologize for any confusion caused by the lack of clarity in our original manuscript and appreciate the reviewer for bringing this to our attention. We will clarify the meaning of the dotted lines in Figure 5 in the revised version of our manuscript.

Q7: Provide more quantitative results and more comparisons with other models on the cross-domain settings.

Reply: Thanks. Following your valuable suggestion, here we provide the quantitative results of the cross-domain generation experiments with all datasets combinations. To be more specific, the model is first trained on one domain (*e.g.,* VGGFace) and then tested on another domain (*e.g.,* Animal Faces), while other settings remained consistent with the main experiments. The following table presents the quantitative results. It is evident that the synthesis performance deteriorates when the training and testing data are from different domains, particularly when the domain gap is substantial (*e.g.,* transferring from Flowers to Animal Faces and VGGFace). Nonetheless, our proposed techniques effectively improve the transfer performance under different baselines, further emphasizing the compatibility and flexibility of our method. We will include these quantitative results in the revised version of our manuscript to provide a more comprehensive investigation of the transferability of our proposed method across different domains.

Hope that the above discussions could address your concerns, please let us know if you have any further questions. Thanks for your effort and constructive suggestions again.

REFERENCES

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.* 248–255.

[2] Guanqi Ding, Xinzhe Han, Shuhui Wang, Xin Jin, Dandan Tu, and Qingming Huang. 2023. Stable Attribute Group Editing for Reliable Few-shot Image Generation. *arXiv preprint arXiv:2302.00179* (2023).

[3] Zheng Gu, Wenbin Li, Jing Huo, Lei Wang, and Yang Gao. 2021. Lofgan: Fusing local representations for few-shot image generation. In *ICCV*. 8463–8471.

[4] Yan Hong, Li Niu, Jianfu Zhang, and Liqing Zhang. 2022. Few-shot Image Generation Using Discrete Content Representation. In *ACM Int. Conf. Multimedia*. 2796–2804.

[5] Yan Hong, Li Niu, Jianfu Zhang, Weijie Zhao, Chen Fu, and Liqing Zhang. 2020. F2GAN: Fusing-and-Filling GAN for Few-shot Image Generation. In *ACM Int. Conf. Multimedia*.

[6] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2020. Training Generative Adversarial Networks with Limited Data. In *Adv. Neural Inform. Process. Syst.* 12104–12114.

[7] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. 2020. Differentiable augmentation for data-efficient gan training. *Adv. Neural Inform. Process. Syst.* 33 (2020).