



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

Doctoral Thesis

Example-based Methods  
to Explain the Internal Generative Mechanism  
of Deep Generative Neural Networks

Haedong Jeong

Department of Mechanical Engineering  
(System Design and Control Engineering)

Ulsan National Institute of Science and Technology

2022



Example-based Methods  
to Explain the Internal Generative Mechanism  
of Deep Generative Neural Networks

Haedong Jeong

Department of Mechanical Engineering  
(System Design and Control Engineering)

Ulsan National Institute of Science and Technology

Example-based Methods  
to Explain the Internal Generative Mechanism  
of Deep Generative Neural Networks

A thesis/dissertation submitted to  
Ulsan National Institute of Science and Technology  
in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy

Haedong Jeong

06.08.2022 of submission

Approved by



---

Advisor

Kwang In Kim

# Example-based Methods to Explain the Internal Generative Mechanism of Deep Generative Neural Networks

Haedong Jeong

This certifies that the thesis/dissertation of Haedong Jeong is approved.

06.08.2022 of submission

Signature



---

Advisor: Kwang In Kim

Signature



---

Jaesik Choi

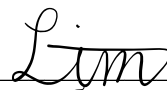
Signature



---

Seungchul Lee

Signature



---

Sungbin Lim

Signature



---

Jaejun Yoo

## Abstract

This thesis studies the internal generative mechanism of Deep Generative Neural Networks based on the example-based method. Since the adversarial training scheme which the generator and the discriminator compete for each objective has emerged, deep generative neural networks with this scheme (mainly called Generative Adversarial Networks, GANs) have shown the remarkable generation performance in various fields (e.g., image or time-series generations.). However, despite of the advances with various structures of the network and training strategies, the state-of-art generative models still generate the low visual fidelity of outputs called *artifact*. In particular, because the previous researches mainly focus to increase the overall performance of the generator, the internal generative mechanism for artifacts is not fully analyzed yet. In this thesis, the example-based methods are provided to understand the internal generative mechanism of deep generative neural networks. Especially, we mainly focus on artifacts of GANs.

The aforementioned problems are important for the reliability of the generator. To exploit the generator for real-world applications, the quality of the generations should be guaranteed. In particular, for the mission-critical system related to human, the low visual fidelity of generations may cause the fatal problems. At the same time, the described problems are challenging: (1) considering the internal generative mechanism of deep generative neural networks indicates analyzing the internal units (or neurons) of internal layers of the generator. Because the modern networks not only use the high dimensional latent space but also numerous internal neurons to generate output, it is non-trivial to interpret internal components directly, and it will be in-efficient. (2) highly non-linear nature of Deep Neural Network disturbs the intuitive interpretation of the functionality of the generator. (3) the supervision (e.g., pre-trained segmentation network) can be needed to understand the mechanism of the internal components. However, because the additional networks aligned with the generation modes of the generator are not always available, the dependency for extra resources will reduce practicality for analysis.

The goal of this thesis is to devise tools for the given problems. The first contribution of this thesis is to devise the efficient sampling strategy which uses the generative boundaries to understand shared semantic information in the internal layers of the generator. Second, this thesis presents the classifier-based internal unit identification to detect internal defective units automatically which cause the low visual fidelity of generations. Furthermore, the sequential correction method is proposed based on the detected units to remove the areas of the low visual fidelity as maintaining the plausible areas. Last but not least, to alleviate the weakness for the supervised analysis, this thesis investigate the internal properties of neurons to understand artifact generating internal units of Generative Adversarial Networks in an unsupervised manner. This thesis also provides the partial geometrical analysis for observed properties based on the relationship between generator and discriminator.



## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Thesis Scope . . . . .	1
1.2	Challenges . . . . .	2
1.3	The Contributions of Thesis . . . . .	3
1.3.1	An Efficient Explorative Sampling Considering the Generative Boundaries	3
1.3.2	Classifier-based Artifact Generating Internal Units Identification and Correction . . . . .	4
1.3.3	An Unsupervised Way to Understand Artifact Generating Internal Units in GANs . . . . .	4
1.4	The Outline of Thesis . . . . .	5
1.5	Publication Notes . . . . .	5
<b>2</b>	<b>An Efficient Explorative Sampling Considering the Generative Boundaries</b>	<b>6</b>
2.1	Introduction . . . . .	6
2.2	Generative Boundary Aware Sampling in Deep Generative Neural Networks . . . . .	8
2.2.1	Deep Generative Neural Networks . . . . .	8
2.2.2	Generative Boundary and Generative Region . . . . .	8
2.2.3	Smallest Supporting Generative Boundary Set . . . . .	10
2.2.4	Explorative Generative Boundary Aware Sampling . . . . .	13
2.3	Experimental Results . . . . .	14

2.3.1	Qualitative Comparison . . . . .	15
2.3.2	Quantitative Results . . . . .	17
2.4	Related Work . . . . .	19
2.5	Final Remark . . . . .	21
<b>3</b>	<b>Automatic Correction of</b>	
	<b>Artifact Generating Internal Units</b>	<b>22</b>
3.1	Introduction . . . . .	22
3.2	Ablation of Artifact Generating Internal Units in GANs . . . . .	24
3.2.1	FID-Based Artifact Unit Identification . . . . .	25
3.2.2	Classifier-Based Artifact Generating Internal Unit Identification . . . . .	26
3.2.3	Generation Concepts of Units in GANs . . . . .	28
3.3	Sequential Correction of Artifacts . . . . .	31
3.3.1	Exploration of Hyperparameters for Sequential Correction . . . . .	32
3.3.2	Local Region Correction . . . . .	32
3.4	Experimental Evaluations . . . . .	33
3.4.1	Qualitative Comparison . . . . .	34
3.4.2	Quantitative Comparison . . . . .	35
3.4.3	Human Evaluation . . . . .	36
3.4.4	Generalization . . . . .	36
3.5	Related Work . . . . .	38
3.6	Final Remark . . . . .	39

<b>4</b>	<b>An Unsupervised Way to Understand Artifact Generating Internal Units in GANs</b>	<b>41</b>
4.1	Introduction . . . . .	41
4.2	Locally Activated Neurons in GANs . . . . .	43
4.2.1	Quantification of Local Activation . . . . .	43
4.2.2	Dynamics for Local Activation during Training . . . . .	46
4.3	Experimental Analysis . . . . .	47
4.3.1	Qualitative Results . . . . .	48
4.3.2	Quantitative Results . . . . .	49
4.4	Discussion . . . . .	52
4.4.1	Geometrical Interpretation of Local Activation . . . . .	52
4.4.2	Comparisons between Perceptual Path Length and CLA . . . . .	55
4.5	Related Work . . . . .	55
4.6	Final Remark . . . . .	57
<b>5</b>	<b>Conclusion and Future Work</b>	<b>58</b>
5.1	Summary of Contribution . . . . .	58
5.2	Future Work . . . . .	58
5.3	Conclusion . . . . .	59
<b>6</b>	<b>Appendix</b>	<b>60</b>
6.1	Details of Human Evaluations . . . . .	60
6.1.1	Evaluation Results . . . . .	60
6.1.2	Evaluation Criteria . . . . .	61



6.2	Artifact Correction Results . . . . .	63
6.2.1	Correction Results in PGGAN trained with CelebA-HQ: Artifact . . . . .	64
6.2.2	Correction Results in PGGAN with CelebA-HQ: Normal . . . . .	65
6.2.3	Correction Results in PGGAN trained with LSUN-Church: Artifact . . . . .	66
6.2.4	Correction Results in PGGAN trained with LSUN-Church: Normal . . . . .	67
6.2.5	Correction Results in PGGAN trained with LSUN-Bedroom: Artifact . . . . .	68
6.2.6	Correction Results in PGGAN trained with LSUN-Bedroom: Normal . . . . .	69
6.3	Sequential Correction in StyleGAN2 and U-net GAN . . . . .	70
6.4	Representative Generation and Highly Activated Generations for Internal Units . . . . .	71
6.5	Exploration of Hyperparameter for CLA . . . . .	75
6.6	Computational Complexity for Computation of CLA . . . . .	76
6.7	Artifact Detection in StyleGAN2 trained with LSUN-Car . . . . .	77
6.8	Artifact Detection in StyleGAN2 trained with LSUN-Cat . . . . .	78
6.9	Artifact Detection in StyleGAN2 trained with LSUN-Horse . . . . .	79
6.10	Artifact Detection in StyleGAN2 trained with FFHQ . . . . .	80
6.11	Artifact Detection in PGGAN trained with LSUN-Bedroom . . . . .	81
6.12	Artifact Detection in PGGAN trained with LSUN-Church . . . . .	82
6.13	Artifact Detection in PGGAN trained with CelebA-HQ . . . . .	83
6.14	Artifact Correction with CLA . . . . .	84
	References . . . . .	84
	Acknowledgements . . . . .	92

## List of Figures

2.1	Illustrative examples for comparison between $\epsilon_{L_2}$ -based sampling (red) and the proposed explorative generative boundary aware sampling (E-GBAS) (blue). We can identify that the $\epsilon_{L_2}$ -based sampling may include the out of samples from the generative region where the given latent code resides. . . . .	7
2.2	An illustrative example for generative region with half-space for the given latent code $z$ (white dot) in the simple generator with 2D latent space. (Left) the half-space, generative boundary (pink), and generative region (green) for each internal neuron. (Right) intersection of each half-space and defined generative region for the internal layer ( $l = 1$ ). . . . .	9
2.3	An illustrative example for the Bernoulli parameter $\theta$ optimization for the given latent code $z$ in the DCGAN trained with MNIST over 2D dimensional latent space [1, 2]. The top-right red box indicates the synthesized digit image and the bottom-right blue box indicates the magnified area nearby the given latent code. (a) shows entire generative boundaries in the first internal layer ( $l = 1$ ). (b) shows SSGBS after Bernoulli optimization with constraint $p = 0.5$ . We can identify that the generative region is expanded as maintaining the quality of generation after optimization. . . . .	10
2.4	An illustrative example for the exploration range of $\epsilon$ -based sampling with various magnitude of $\epsilon$ . The green shaded area denotes the generative region for the given latent code (green dot). (a) The original generative boundaries and region. (b) SSGBS and SSGR after Bernoulli mask optimization. (c) The exploration range (black circle) for small magnitude of $\epsilon$ . We can identify that there may exist a blind area (blue) for exploration in SSGR. (d) The exploration range (black circle) for large magnitude of $\epsilon$ . We can identify that the sampling method can obtain the out-of-region latent codes (red) from SSGR. . . . .	13

2.5	An illustrative example for exploration of GB-RRT and synthesized digit images from obtained latent codes. (a) Visualization of explorative trajectories of GB-RRT for a given latent code (red dot) in the first hidden layer ( $l = 1$ ) of DCGAN trained with MNIST. (b) Generated outputs from uniformly selected latent codes in the trajectories (blue dot in (a)). The red box indicates the synthesized digit image from the given latent code. . . . .	14
2.6	An illustrative example of calculating the magnitude of $\epsilon$ . (a) The accepted samples (black dots), rejected samples (red dots) and average of the accepted samples (blue dot) by E-GBAS in SSGR. (b) Visualization of the selection of $\epsilon_{L_2}$ to make the area close to SSGR. The $\epsilon_{L_2}$ -balls of each distance are depicted as red (min), orange (max) and blue (average of min/max distance) colored circle. . . . .	16
2.7	Geometrical comparison of (a) E-GBAS, (b) $\epsilon_{L_2}$ and (c) $\epsilon_{L_\infty}$ -based sampling methods. Although the two $\epsilon$ -balls cover some area of the generative region, they cannot cover all of the generative region and have a possibility to include infeasible area. Whereas, the E-GBAS includes the only feasible area of the generative region for sampling. . . . .	16
2.8	Synthesized images from the given latent code (left), $\epsilon$ -based sampling (middle) and E-GBAS (right) in the three deep generative neural networks (DCGAN trained with MNIST, PGGAN trained with LSUN-Church, and PGGAN trained with CelebA-HQ.). We confirm that the synthesized images from latent codes obtained by E-GBAS have more consistent generative information compared to the $\epsilon_{L_2}$ -based sampling. . . . .	18
2.9	Examples of variations of synthesized images for each target internal layer. The first row depicts the standard deviations of synthesized images for each target internal layer. The second row depicts randomly selected synthesized images in each SSGR. . . . .	19

3.1	An illustration of the proposed classifier-based artifact generating internal unit detection and sequential correction method. (Top) Identification of the artifact generating internal units for each internal layer. (Bottom) the generation flow for single-layer correction and sequential correction method. In identification procedure, we measure the intersection-over-unions (IoUs) between defective areas (Grad-CAM mask) and an activation pattern of internal unit $u$ for artifact generations. The averages of IoUs over artifact generations are used as the defective scores in each internal layer. The sequential correction method adjusts the generation flow of artifact generating internal units and improves the generation performance of GANs without retraining of the network. . . . .	23
3.2	Illustrative examples for artifact generations of PGGAN trained with LSUN-Church (top), CelebA-HQ (middle), and LSUN-Bedroom (bottom). . . . .	24
3.3	The histogram of D-values for each class of generation in PGGAN trained with CelebA-HQ. The black line denotes the corresponding D-value for each generation. The third and the last generations have similar defective background area, however, the corresponding D-values are largely different. . . . .	25
3.4	The representative generation (discuss in 3.2.3) and highly activated generations for unit 134 in layer 6 of PGGAN trained with LSUN-Church. Although the unit 134 seems to be related to night view, the high FID-score is measured. . . . .	25
3.5	Illustrative examples for the explanation masks from the Grad-CAM method with artifact class score. The masks highlight on the defective areas which the normal generations are hard to contain. The red color indicates high value compared to white color. . . . .	26
3.6	Illustrative examples for the single-layer correction results. We ablate top-20 internal units for the baseline correction method (i.e., FID-based unit detection, FID) [3] and our method (defective score based unit detection, DS). (a) Correction results for normal generations. We can identify that the ablations from both methods do not affect visual fidelity of generations. (b) While the baseline fails to correct generation, our method can remove the shadow effect with maintaining the original outline. (c) Both correction methods fail to repair the hole on the wall (left) or the texture error (right). . . . .	27

3.7	Illustrative examples of the representative generation and highly activated generations for the given internal unit in layer 6 of PGGAN trained with LSUN-Church. (a) A unit seems to be related to concrete generation concepts (shadow artifacts). (b) A unit that does not have concrete generation concepts. The representative generation in this case seems to be blurred and it is non-trivial to define one clear generation concept for this unit. . . . .	28
3.8	Illustrative example for the clusters of the representative generations in layer 6 of PGGAN trained with LSUN-Church. We can identify that the multiple internal units are related to one artifact concept. For instance, the texture artifact concept can be derived independently or together by unit 5 and unit 39. . . . .	29
3.9	An illustrative example for correction results for texture artifact. In the Manual unit detection case, we select and ablate 15 units related to the texture artifact. The manual case only corrects the texture artifact and improve the details of church. 29	29
3.10	An illustrative example for a trade-off between defective and plausible area. The % denotes the ratio of zero-ablation for internal units. Although the size of the stain (top-right) is reduced when increasing the number of ablation units, the details of trees and church is degraded at the same time. . . . .	30
3.11	The computed FID-scores on the numbers of ablated units on layer 6 of PGGAN trained with various dataset. We can identify that when the number of zero-ablation for internal units increase, the FID-score exponentially increase. . . . .	30
3.12	Quantitative and qualitative results over various hyperparameter settings (stopping layer $l$ and the portion of ablated internal units $n$ ). We can identify that when the stopping layer $l$ is determined as the deeper layer, the visual fidelity of generation is degraded, while for shallow layer, the pink defective areas are still remained. . . . .	32
3.13	The effect of sample-based correction method for the sequential correction method (SeqC) in PGGAN trained with CelebA-HQ. We can observe that the areas which was not highlighted on by the artifact mask cannot be changed. . . . .	33

3.14 Illustrative examples for the various correction methods in the PGGAN trained with the various dataset. Original denotes the initial generation for the given latent code and FID denotes FID-score based unit detection with zero-ablation in the single-layer. DS and SeqC commonly utilize the defective score based unit detection. DS performs zero-ablation in single-layer and SeqC indicates the sequential correction method. We confirm that the sequential correction removes the defection areas effectively compared to other correction methods. More examples for correction are available in Chapter 6.2. . . . . . 34

3.15 Illustrative examples for corrected generations by the sequential correction method on StyleGAN2 trained with FFHQ. We choose 100 artifact generations, and calculate the FID-score and Realism Score (RS) for the corrected generations with various stopping layers  $l$ . We can identify that FID-score and Realism Score are improved when the stopping layer  $l$  increases up to the middle layer ( $l = 6$ ). In first row in (b), we can identify that the sequential correction method with middle stopping layer can remove the stains on the face effectively. In second row in (b), we also observe the stain in left-side is removed without drastic change of plausible areas. . . . . . 37

3.16 Illustrative examples for the corrected generations with glasses or sunglasses. Although the defective areas are expected in background, the sequential correction method removes the glasses or sunglasses, while the background information is slightly changed. . . . . . 39

4.1 An illustrative example of the locally activated neuron in PGGAN trained with CelebA-HQ. We manually choose two internal units in layer 6 ( $\in \mathbb{R}^{512 \times 16 \times 16}$ ) which are expected to relate to low visual fidelity (red background in  $G(z)$ ) and mouth (plausible area). (Left) The black box in each internal unit denotes the spatial information of the selected internal neuron. (Right) The visualization of activation patterns in the latent space. The black solid line in each side indicates the activation values in the corresponding latent axis. The latent axes are randomly selected for the visualization. The red dots indicate the change points (Definition 8) and green/blue lines are the approximated curvature of local activation for each latent axis. The neuron in unit 179 (related to the low visual fidelity) has more locally activated pattern compared to the activation pattern of the neuron in unit 229 (related to mouth). . . . . . 42

4.2 An illustrative example for activation patterns of the internal neurons in the manually selected internal units (unit 19 and unit 158) for the given latent code  $z$ . The internal neuron 3 and 4 are related to distorted jaw, while neuron 1 and 2 are related to eye information. The middle column depicts the visualization of heatmaps for the activation values nearby  $z$  where each row corresponds to each latent axis  $a$  in the latent space ( $a \in [1, 2, \dots, D_z]$ ). For intuitive visualization, we sort the rows based on the magnitude of activations in the descending order from top to bottom. The right column depicts the activation values for a specific axis (the dotted rows in the middle column) as the 2D visualization. X-axis denotes the perturbations and Y-axis denotes the corresponding activation value. The internal neurons related to the defective areas (3 and 4) are more locally activated compared to normal neurons. . . . . 44

4.3 An illustrative examples for left/right change points and approximated curvature of activation values for the given activation values of perturbations. The white start denotes the activation value for the given latent code, and red dots denote the left/right change points. The green line denotes the quadratic approximation of the given activation values for perturbations. . . . . 45

4.4 An illustrative examples for CLA based artifact generating internal unit identification. Bilinear upsampling for internal units are performed to overlay with the original generation. The red color denotes the high activation values. . . . . 46

4.5 An illustrative example for the change of magnitude of CLA and emerging process of defective regions. (First row) The section of activation pattern in the latent space for the selected internal neuron (white box in each internal unit) with randomly selected axes. The red color denotes positively high value. (Second row) The generation  $G(z)$  for the fixed latent code. (Third row) Computed the magnitude of CLA for the selected internal neurons in each internal unit. The blue color denotes negatively high value. We can observe that increase of magnitude of CLA, change of activation pattern, and semantic information of generation are well-aligned during the training process. . . . . 47

4.6 Results of artifact detection game in GANs trained with various dataset. We visualize bottom-24 (low CLA score) and top-24 (high CLA score) generations for qualitative comparison. We verify that the generations with the high CLA score have lower visual fidelity compared to the the generations with the low CLA score. Chapter 6.7 - 6.13 provide more examples for results of artifact detection game. . . . . 50

4.7	An illustrative examples for the artifact correction results on GANs trained with various dataset for high CLA group. More correction examples are available in Chapter 6.14. . . . .	51
4.8	Precision and recall for each group in GANs trained with various dataset. Each color bar denotes each group, and each alphabet indicates each dataset. The $\psi$ means the hyperparameter for truncation trick. . . . .	52
4.9	The geometrical illustrative examples of available weight parameter update cases ( $\bar{h}_{l-1} \rightarrow \bar{h}_{l,i}$ ). The black arrow and the black dot represent $\bar{h}_{l-1}$ , the green arrow is $\delta_i \bar{h}_{l-1}$ (update term), the blue arrow is $\bar{w}_{g_l,i}$ (original parameter), and the red arrow is $\bar{w}_{g_l,i}^+$ (updated parameter). The bottom-left plot denotes the activation values on the black dashed line before and after the update. The bottom-right plot shows the conceptual representation of the activation pattern change after the update ( $z_0 \rightarrow \bar{h}_{l,i}$ ). . . . .	54
6.1	Detailed results for human evaluation of the re-labeling on the corrected generations in PGGAN trained with CelebA-HQ, LSUN-Church, and LSUN-Bedroom datasets . . . . .	61
6.2	Illustrative examples for artifact generations according to the descriptions in the criteria for each dataset. The one generation can be included in multiple criteria. For instance, left-bottom generation satisfies the hair and background type of artifact at the same time. . . . .	61
6.3	An illustrative examples for correction result for each correction method in PGGAN trained with various dataset. Although the entire correction method can preserve the plausible areas, the proposed sequential correction method can repair the defective areas effectively. See the improved quality of church in 4th row. . . . .	63
6.4	Correction results for the artifact generations in PGGAN trained with CelebA-HQ. . . . .	64
6.5	Correction results for the normal generations in PGGAN trained with CelebA-HQ. . . . .	65
6.6	Correction results for the artifact generations in PGGAN trained with LSUN-Church. . . . .	66
6.7	Correction results for the normal generations in PGGAN trained with LSUN-Church. . . . .	67



6.8	Correction results for the artifact generations in PGGAN trained with LSUN-Bedroom. . . . .	68
6.9	Correction results for the normal generations in PGGAN trained with LSUN-bedroom. . . . .	69
6.10	Correction results for the generations in StyleGAN2 trained with FFHQ. . . . .	70
6.11	Correction results for the generations in U-net GAN trained with FFHQ. . . . .	70
6.12	An illustrative example for how to obtain highly activated generations and synthesize the representative image. . . . .	71
6.13	Representative image and highly activated generations for each internal unit in layer 6 of PGGAN trained with CelebA-HQ. (Left) FID-score based unit generations. (Right) defective score based unit generations. . . . .	72
6.14	Representative image and highly activated generations for each internal unit in layer 6 of PGGAN trained with LSUN-Church. (Left) FID-score based unit detection. (Right) defective score based unit detection. . . . .	73
6.15	Representative image and highly activated generations for each internal unit in layer 6 of PGGAN trained with LSUN-Bedroom. (Left) FID-score based unit detection. (Right) defective score based unit detection. . . . .	74
6.16	Realism Score and Perceptual Path Length for the various hyperparameter settings. (First column) division $n = 20$ and target internal layer $l = 4$ are fixed. (Second column) search bound $R = 30$ and division $n = 20$ are fixed. (Third column) search bound $R = 30$ and target internal layer $l = 4$ are fixed. . . . .	75
6.17	Generations depending on the magnitude of CLA in StyleGAN2 trained with LSUN-Car. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	77
6.18	Generations depending on the magnitude of CLA in StyleGAN2 trained with LSUN-Cat. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	78

6.19 Generations depending on the magnitude of CLA in StyleGAN2 trained with LSUN-Horse. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	79
6.20 Generations depending on the magnitude of CLA in StyleGAN2 trained with FFHQ. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	80
6.21 Generations depending on the magnitude of CLA in PGGAN trained with LSUN-Bedroom. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	81
6.22 Generations depending on the magnitude of CLA in PGGAN trained with LSUN-Church. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	82
6.23 Generations depending on the magnitude of CLA in PGGAN trained with CelebA-HQ. We identify low CLA group has higher visual fidelity compared to high CLA group. . . . .	83
6.24 An illustrative examples for the results of artifact correction for high CLA group on GANs trained with various dataset. We note that the correction is performed with only internal information of the trained generator. . . . .	84

# Chapter 1

## Introduction

The main purpose of a deep generative neural network is to generate realistic data such as image. Recently proposed adversarial training framework (i.e., Generative Adversarial Networks, GANs) with deep neural networks [4], have shown superior generation performance not only the quality of individual generations but also the diversity of generated outputs. In particular, for the image generation, the modern GANs synthesize high resolution images which are often difficult to distinguish from real data [5–7]. These outstanding capabilities pave the way for GANs to be utilized in various real-world applications [8–10]. Although the GANs show the remarkable successes, the networks still suffer from synthesizing generated outputs which include defective areas called as *artifacts*, which can make them unsuitable for being employed in the human related mission-critical problems. As a result, investigating the root cause of artifact generations and possible improvements to enhance the quality of generations have turned out to be essential. Because in the GAN domain, the main interests of existing work has performed to enhance the visual fidelity of generations by designing the training strategies or designing more complex structure of models, the internal generative mechanism for artifact generations in the generative models is not well-studied yet.

### 1.1 Thesis Scope

This thesis aims to explain the internal generative mechanism of Deep Generative Neural Networks. The main studying target which this thesis focuses on is artifact of GANs. To understand the internal generative mechanism of the generator, one may consider two aspects: (1) generating neighbor examples for the given latent code, and (2) using the supervision to interpret the generative role of internal units in the generator. Under the observations for the generating of neighbors, the  $\epsilon$ -ball based sampling strategy can become one solution to interpret the generation mechanism. However, the question of how the  $\epsilon$  is determined for the reliable explanation is still remained. In terms of the supervision, the pre-trained image segmentation network can be used to dissect each generative role of the internal unit [3]. Although the additional network can establish the intuitive relation between the activation patterns of the internal units and the

segmentation maps of the generations, it is non-trivial to identify the generative modes from the non-coverage modes of the segmentation network. For example, the outside scene segmentation network is not proper to apply for an animal head generator.

This thesis considers the following problems. First, generative boundaries of internal layer in the generator are considered as the constraints for sampling in the latent space. The goal is to guarantee the shared semantic information in the perspective of the network, and acquire the reliable examples to explain the internal generative mechanism. The second aspect is to consider the additional network (i.e., classifier) for the specific purpose such as artifact detection. From this procedure, the artifact generating internal units can be detected and corrected automatically without post-training of the generator. Finally, this thesis provides an unsupervised method to interpret artifact generating internal units based on the internal activation information of the generator in the GANs.

## 1.2 Challenges

In the given problem settings, there are certain challenges. For analysis of internal generative mechanism for deep generative neural networks, the numerous internal neurons have to be considered. Although the internal neurons of hidden layer in deep neural networks are well-studied for each role from given task (e.g., detection for the dome of building) in previous work [3, 11, 12], direct accessing to entire internal neurons may be computationally prohibited.

The second challenge is that the modern generator mainly have the high-dimensional latent space. Because it is well-known that the rich semantic information is embedded to the latent space of the generative models [13, 14], exploration in the latent space to analyze internal generative mechanism of GANs seems to be natural. However, the efficient exploration strategy is needed, as the brute-force search in the high-dimensional latent space is infeasible.

The third challenge is that the current frameworks mainly focus the well-established semantic concepts such as tree or color of hair. In order to handle these concepts, they utilize extra network or annotation procedure as the supervision. However, because artifacts usually include abstract and vague features, it is non-trivial to apply these frameworks directly.

Lastly, the supervision and generative models have to be aligned for reliable explanations. There is a clear dependency between reliability of explanation and the coverage of the supervision, as the explanations are generated based on the supervision. For example, if the face segmentation model is used to investigate the generative role of internal neurons for church generator, the reliability for identified generative roles will be low. Although, for general datasets (e.g., LSUN-Church outdoor [15]), the aligned supervisions are already prepared, because the current deep generative neural networks have covered various dataset, it may be infeasible to possess the proper supervision for each dataset. It is necessary to alleviate the dependency of the supervision for explanation of the internal generative mechanism in deep generative neural networks.

## 1.3 The Contributions of Thesis

This thesis propose three example-based methods to explain the internal generative mechanism of deep generative neural networks: (1) Explorative generative boundaries aware sampling (E-GBAS), (2) classifier-based artifact generating internal units identification and correction, and (3) investigation for internal properties of neurons of GANs to understand artifact generating internal units in an unsupervised manner. Sampling strategy with generative boundaries (Chapter 2) is a novel framework to explore the latent space as maintaining the shared semantic information in a perspective of the network, while general methods (e.g.,  $\epsilon$ -ball based sampling) cannot guarantee the shared semantic information.

The second contribution (Chapter 3) is a classifier-based artifact generating internal unit identification and correction. Although this chapter follows the previous supervised framework, we focus that the investigation for internal generative mechanism related to the artifacts. The proposed method is empirically verified to show a better detection and correction performance compared to the previous work. The empirical evidences for the generalization of the proposed approach regardless of the structure of the generator also provided.

The third contribution (Chapter 4) is investigation of internal properties of neurons to understand the artifact generating internal units of GANs in an unsupervised manner. This method presents novel metric, *local activation*, constructed from the internal activation information of the generator. From the proposed metric, the visual fidelity of individual generations can be quantified without any extra resources. The discussion with restricted conditions for the cause of local activation is also provided in a perspective of the competition between the generator and the discriminator.

### 1.3.1 An Efficient Explorative Sampling Considering the Generative Boundaries

This thesis devises an explorative sampling method to obtain reliable examples for interpreting the internal generative mechanism of deep generative neural networks. To guarantee the reliability of acquired samples, the attributes should be maintained between samples in a perspective of the network. Chapter 2 presents exploration strategy with constraints based on generative boundaries to retain the neural representation during the sampling process. The proposed method is comprised of two steps: (1) to handle numerous generative boundaries in the generator, our method investigates the set of critical generative boundaries for the given latent code with Bernoulli dropout approach [16]; (2) then our method efficiently acquire latent codes which share similar attributions as the given latent code in a perspective of the trained generator by expanding the tree-like exploring structure [17] until it reaches the generative boundaries of the generative region.

The advantages of our method are twofold: (1) While the rejection sampling based on the Monte Carlo method easily fails when the generative region is unknown, ours can guarantee

sample acceptance for the generative region in high dimensional space; (2) it can obtain the latent codes which share the similar attributes in a perspective of the trained generator where the commonly used  $\epsilon$ -based sampling [18] is not precise to acquire latent codes in the complex non-spherical generative region [19]. Experimental results empirically demonstrate that the proposed sampling strategy gathers more reliable latent codes compared to various  $\epsilon$ -based sampling methods on various GANs to understand the internal generative mechanism.

### 1.3.2 Classifier-based Artifact Generating Internal Units Identification and Correction

This thesis devises the classifier-based artifact generating internal units identification and correction method. Chapter 3 focuses internal generative mechanism related to artifact and provides empirical evidences which well-trained generators usually have artifact generating internal units. Chapter 3 also presents that the side-effect of the single-layer based correction, and proposes the sequential correction method which considers the consequent layers to improve the visual fidelity of individual generations with suppression of the side-effect of the single-layer based correction.

In our approach, at first, we perform labeling procedure for randomly sampled generations based on pre-defined criteria into two classes, *Normal* and *Artifact*. Then we train a classifier on some randomly sampled real images and entire the annotated generations to discriminate inputs into their corresponding classes. After training, we perform an explanation method [20] on our trained classifier to generate an estimated mask for defective areas. By computing the alignment between an activation pattern of individual internal unit and the defective areas' segmentation mask, we can determine internal units related to defective area. To improve the visual fidelity of individual generations (i.e., correct the defective areas), we ablate detected internal units which have high overlaid scores.

The contributions of this chapter are threefold: (1) We perform an empirical analysis for artifact generations as compiling a large dataset of selected generations; (2) We identify artifact generating internal units of the trained generator as computing the intersection-over-union of the activation pattern of each internal unit and pseudo defective area masks extracted from the trained classifier on the curated dataset. (3) We devise an artifact correction method by ablating artifact generating internal units in global sense, which improve the visual fidelity of artifact generations while preserving normal area in the generations. We additionally propose the sequential correction method to improve the correction performance by sequentially adjusting the activations of artifact generating internal units throughout the consequent layers.

### 1.3.3 An Unsupervised Way to Understand Artifact Generating Internal Units in GANs

This thesis proposes the novel concept, *local activation*, related to the properties of internal neurons to evaluate and repair the artifact generations in an unsupervised manner. We empirically

demonstrate the relationship between the proposed concept and the low visual fidelity in artifact detection and correction settings. Chapter 4 also includes the geometrical discussion to partially reveal the mathematical reason of the observed relationship in a perspective of the adversarial relationship between the generator and the discriminator.

The contributions of this chapter are twofold: (1) The detection and correction of artifact generations are performed without any additional networks or supervisions such as annotation. The evaluation of visual fidelity is achieved solely on the trained generator as utilizing the internal properties of neurons for evaluation; (2) the devised method can be applied to various structures of GANs for assessing the visual fidelity of individual generations, as the proposed approach is based on internal neurons, which are the common components of deep neural networks. In experiments, we empirically verify that the proposed approach can detect and correct artifact generations effectively on various structure of GANs. We also confirm that the experimental results from the proposed method are well-aligned to the exist metrics which require extra network or training dataset.

## 1.4 The Outline of Thesis

Chapter 2 presents an sampling strategy which considers the generative boundaries induced by the internal layers in the latent space. Chapter 3 devises the classifier-based artifact generating internal units identification and correction. Chapter 4 provides the analysis for the relationship between visual fidelity of individual generations and the internal properties of neurons in the trained generator. The proposed unsupervised way can be helpful to understand the artifact generating internal units, which can alleviate the weakness of the supervised method for the explanation of internal generative mechanism in GANs. In this chapter, we also provide a geometrical analysis to partially reveal why the local activation is related to the low visual fidelity in a perspective of the adversarial training. The thesis ends with Chapter 5 containing a summary and open directions for future work.

## 1.5 Publication Notes

This thesis is structured with modifications and revisions from the following publications<sup>1</sup>:

- G. Jeon.\*, H. Jeong.\*, and J. Choi., AAI-20 [21]: Chapter 2.
- A. Tousi.\*, H. Jeong.\*, J. Han., H. Choi., and J. Choi., CVPR-21 [22]: Chapter 3.
- H. Jeong., J. Han., and J. Choi., AAI-22 [23]: Chapter 4.

---

<sup>1</sup>\*equal contribution

## Chapter 2

# An Efficient Explorative Sampling Considering the Generative Boundaries

Deep generative neural networks have shown diverse and high-quality data generation such as face image. In particular, since the adversarial training framework has emerged, many deep neural networks with this scheme (i.e., Generative Adversarial Networks, GANs) has achieved remarkable generation performance. Despite recent advances in these generative models, interpreting internal generative mechanisms of deep generative neural networks remains a challenge. In this chapter, we devise an efficient sampling method considering the generative boundaries with exploration to interpret the internal generative mechanism of the trained generator. The proposed method efficiently acquires latent codes with similar attributes for a given latent code in a perspective of the generator. We introduce generative boundaries and region which can represent shared attributes in the internal layers of the generator, and investigate the inside of the network based on this shared semantic information. To handle numerous generative boundaries in the trained generator, we find the essential set of generative boundaries to maintain the quality of generation using optimization. From obtaining latent codes in the generative region comprised of important generative boundaries, we can indirectly explain the internal generative mechanism on the internal layers of the trained generator. We also demonstrate that our algorithm can find more consistent, and reliable latent codes to explain the internal generative mechanism of the generator compared to the various  $\epsilon$ -based sampling methods.

### 2.1 Introduction

In general, a deep generative neural network maps a randomly sampled vector in the latent space to a sample (e.g., face image) which we want to generate in the data space. In other words, generative instances are embedded as the latent codes of the latent space in a perspective of the trained generator. In this sense, we can expect that the latent space of the trained generator is divided by the generative boundaries induced by the structure of the generator. The latent codes



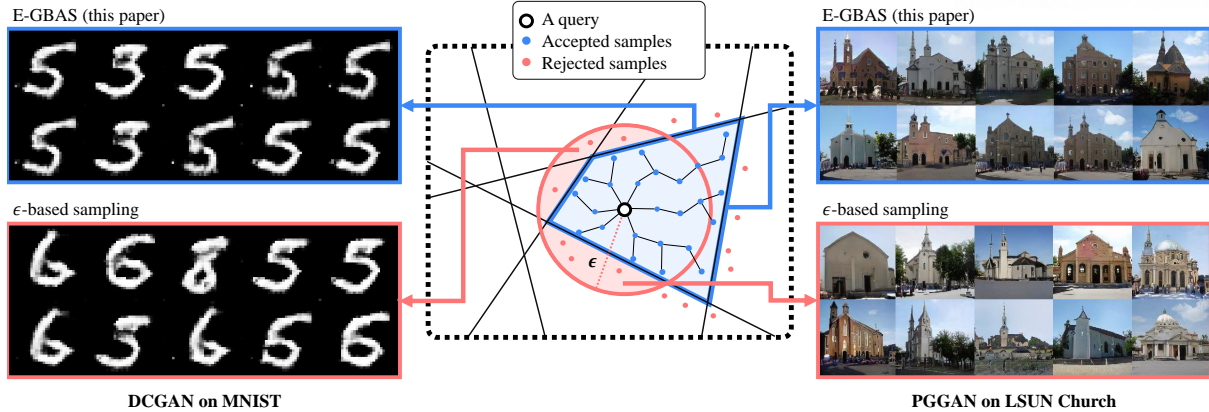


Figure 2.1: Illustrative examples for comparison between  $\epsilon_{L_2}$ -based sampling (red) and the proposed explorative generative boundary aware sampling (E-GBAS) (blue). We can identify that the  $\epsilon_{L_2}$ -based sampling may include the out of samples from the generative region where the given latent code resides.

in the latent space can include the generation concepts, and this information can be represented by which side of generative boundaries latent codes are resided. We utilize these properties to interpret the internal generative mechanism of the generator.

If we have a latent code and choose the target internal layer in the trained generator, we can define the corresponding generative region which is comprised of generative boundaries. Latent codes in the generative region will have the similar activation vector as the given latent code and deliver similar generation information to the next internal layer. The details of the delivered information can be interpreted indirectly by comparing the generated outputs from these latent codes. For example, in the trained generator which can synthesize human faces, if we observe the generative region where latent codes share a specific hair color but vary in other properties (e.g., eye, mouth, and etc.), we can identify that this generative region controls the generative information for the hair color.

However, it is difficult to gather latent codes in the generative region of the latent space induced by the trained generator, because (1) numerous generative boundaries are correlated in the latent space, and (2) a small perturbations in the latent space may cause a highly non-linear change in the internal activation vector and the final generation. As we recap the previous example, we may observe the various generative regions with distinct attributes such as different hair colors or their combinations. Additionally, a small perturbations of the latent code in the latent space may change the entire attributes of the generation output [24]. To overcome these challenges, we need a method to investigate the proper generative region and explore the latent space efficiently.

This chapter organizes as follows. The concepts of generative boundary and region are defined to establish the semantic sharing between latent codes in a perspective of the network. The described concepts are motivated the decision boundary and region in the classifier. Then

we devise explorative generative boundary aware sampling (E-GBAS) comprised of two steps to obtain the reliable examples: (1) extraction for a set of essential generative boundaries to handle numerous generative boundaries in the trained generator, and (2) tree-like exploration to consider the complicated generative region in the latent space. We empirically demonstrate that our method obtains more consistent and reliable latent codes compared to various  $\epsilon$ -based sampling methods on deep convolutional GANs (DCGAN) [1] and progressive growing of GANs (PGGAN) [25] trained with various dataset.

## 2.2 Generative Boundary Aware Sampling in Deep Generative Neural Networks

In this section, we present the explorative generative boundary aware sampling (E-GBAS) method which is our main contribution of this chapter. The proposed method can acquire latent codes which share the similar generative attributes in a perspective of the trained generator. At first, we define the notions used in our explorative sampling algorithm. Then, we introduce E-GBAS which consists of (1) an estimation of the essential generative boundaries where a set of selected generative boundaries can maintain the generation quality, and (2) an efficient tree-like exploration to acquire reliable latent codes in the generative region which mainly have the complicated non-convex shape.

### 2.2.1 Deep Generative Neural Networks

Despite various structure of deep generative neural networks, we can represent the network in a unified form. For the given deep generative neural network with  $L$  layers, the function of generator  $G$  is represented by  $G(z) = g_L(g_{L-1}(\dots(g_1(z)))) = g_{L:1}(z)$ , where  $z$  is a latent code in the latent space  $\mathcal{Z} \subset \mathbb{R}^{D_z}$ .  $g_{l:1}^i(\cdot)$  indicates the value of  $i$ -th element and  $g_{l:1}(z) \in \mathbb{R}^{D_l}$ . The operation  $g_l(\cdot)$  includes linear transformations and a non-linear activation function such as LeakyReLU.

### 2.2.2 Generative Boundary and Generative Region

In the trained deep generative neural networks, the latent space is divided by hyper-surfaces, and the configuration of these boundaries is determined after the training process. The trained generator synthesizes the final output based on the configuration of boundaries. We name these boundaries as the generative boundaries.

**Definition 1 (Generative Boundary)** *Let a latent code  $z$ , a generator  $g$ , and the index of internal layer  $l$ . The  $i$ -th generative boundary at the  $l$ -th internal layer is defined as,*

$$B_l^i = \{z | g_{l:1}^i(z) = 0, z \in \mathcal{Z}\}.$$

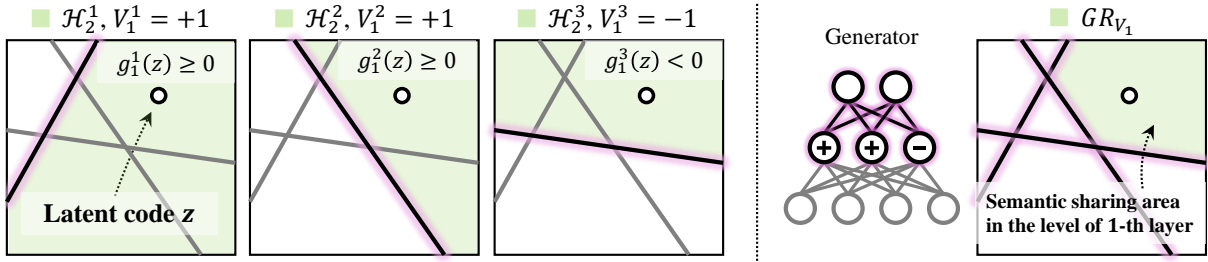


Figure 2.2: An illustrative example for generative region with half-space for the given latent code  $z$  (white dot) in the simple generator with 2D latent space. (Left) the half-space, generative boundary (pink), and generative region (green) for each internal neuron. (Right) intersection of each half-space and defined generative region for the internal layer ( $l = 1$ ).

In general, there are a large number of generative boundaries in the  $l$ -th internal layer of the trained generator, and the arrangement of generative boundaries comprises the generative region in the latent space. As we are mainly focus on the generative region consisted of generative boundaries, we refer the definition of half-space that can be a essential component of the generative region.

**Definition 2 (Half-space)** Let a half-space indicator  $\mathbf{V}_l \in \{-1, 0, +1\}^{D_l}$  for the  $l$ -th internal layer. Each element  $V_l^i$  indicates either or both of two sides of the half-space divided by the  $i$ -th generative boundary. We define the half-space as,

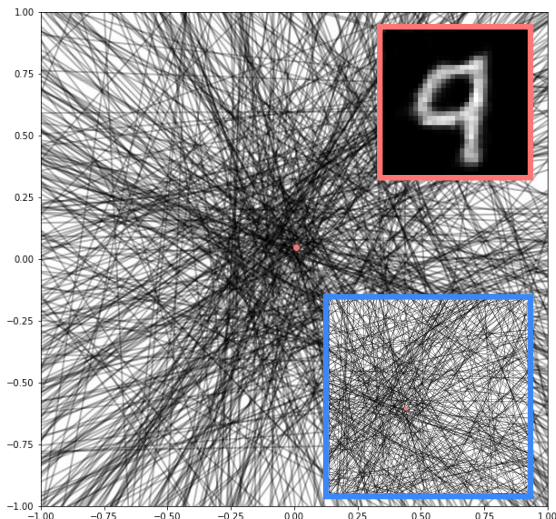
$$\mathcal{H}_l^i = \begin{cases} \mathcal{Z} & \text{if } V_l^i = 0 \\ \{z | V_l^i g_{l,1}^i(z) \geq 0\} & \text{if } V_l^i \in \{-1, +1\} \end{cases}$$

where  $\mathcal{Z}$  is latent space,  $D_l$  is a dimension of the  $l$ -th internal layer, and  $z \in \mathcal{Z}$ . The generative region can be expressed as the intersection of each half-space in the  $l$ -th internal layer. For the case where  $V_l^i = 0$ , as the half-space is defined as the entire latent space  $\mathcal{Z}$ ,  $i$ -th generative boundary does not affect to consist of the generative region.

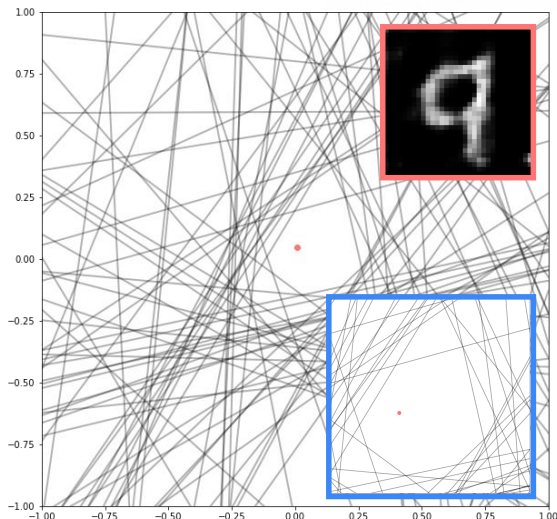
**Definition 3 (Generative Region)** Given a half-space indicator  $\mathbf{V}_l$  in the  $l$ -th internal layer, let the set of corresponding half-spaces  $\mathbf{H} = \{\mathcal{H}_l^1, \dots, \mathcal{H}_l^{D_l}\}$ . Then the generative region  $GR_{V_l}$  is defined as,

$$GR_{V_l} = \cap_{\mathcal{H} \in \mathbf{H}} \mathcal{H}.$$

When we consider the first internal layer ( $l = 1$ ), the generative boundaries can be represented as the linear hyperplanes. The generative region will be constructed by generative boundaries (i.e., linear hyperplanes), and appears as a convex polytope. Figure 2.2 shows an illustrative example for the generative region in the first internal layer. However, if the internal layers are stacked with non-linear activation functions ( $l > 1$ ), the generative boundaries are bent, and then the generative region shows a complicated non-convex shape [26, 27].



(a) Original generative boundaries and synthesized digit image from the given latent code.



(b) Generative boundaries in the SSGBS and synthesized digit image from the given latent code.

Figure 2.3: An illustrative example for the Bernoulli parameter  $\theta$  optimization for the given latent code  $z$  in the DCGAN trained with MNIST over 2D dimensional latent space  $[1, 2]$ . The top-right red box indicates the synthesized digit image and the bottom-right blue box indicates the magnified area nearby the given latent code. (a) shows entire generative boundaries in the first internal layer ( $l = 1$ ). (b) shows SSGBS after Bernoulli optimization with constraint  $p = 0.5$ . We can identify that the generative region is expanded as maintaining the quality of generation after optimization.

### 2.2.3 Smallest Supporting Generative Boundary Set

In classification task, the decision boundaries of the classifier have crucial role, because input samples in the same decision region are categorized as the same class label. In this sense, we will consider the generative boundaries and generative regions of the deep generative neural networks.

Specifically, we want to gather latent codes which reside in the same generative region defined by the given latent code. The latent codes in the this generative region will have similar attributes in a perspective of the trained generator. To define this characteristics formally, at first, we define the condition, Neural Representation Sharing (NRS), which can evaluate that the latent codes share the neural representation or not.

**Definition 4 (Neural Representation Sharing)** *Let a generator  $g$  and the index of internal layer  $l$ . Given a pair of latent codes  $z_i, z_j \in \mathcal{Z}$  satisfies the neural representation sharing condition in  $l$ -th internal layer if*

$$\text{sign}(g_{l:1}^k(z_i)) = \text{sign}(g_{l:1}^k(z_j)), \quad \forall k \in \{1, 2, \dots, D_l\}.$$

Although the above condition is well-defined for entire generative boundaries in the latent space, it is practically challenging to obtain latent codes which satisfy the condition. Because as shown in Figure 2.3 (a), numerous generative boundaries exist in the latent space, we can expect that the variation of generations from latent codes in the condition is too small to interpret the internal generative mechanism. Also, various information which is represented by a large number of generative boundaries, makes it hard to investigate which boundary is in charge of each piece of information. To alleviate these challenges, we relax the neural representation sharing condition by considering a set of the important generative boundaries.

**Definition 5 (Relaxed Neural Representation Sharing)** *Given a subset  $S$  and a pair of latent vectors  $z_i, z_j \in \mathcal{Z}$  satisfies the relaxed neural representation sharing condition if*

$$\text{sign}(g_{l:1}^k(z_i)) = \text{sign}(g_{l:1}^k(z_j)), \quad \forall k \in S \subset \{1, 2, \dots, D_l\}.$$

Then, we should select essential generative boundaries for the relaxed neural representation sharing in the  $l$ -th internal layer. We believe that not entire internal neurons deliver significant information for the final generations of the network, because some neurons can have low contribution for the final information [28]. Additionally, in previous work [3], we can identify that a subset of featuremap units mainly contributes to the final generations in GANs. From these observations, we define the *smallest supporting generative boundary set (SSGBS)*, which minimizes the relevance of minor (non-critical) generative boundaries to synthesize the final generation.

**Definition 6 (Smallest Supporting Generative Boundary Set)** *Given the generator  $G$  and a query  $z_0 \in \mathcal{Z}$ , for  $l$ -th layer and any real value  $\delta > 0$ , if there exists an indicator  $\mathbf{V}_l^*$  such that*

$$\|G(z) - G(z_0)\| \leq \delta, \quad z \in \{z | f_{l-1:1}(z) \in GR_{\mathbf{V}_l^*}\}$$

*and there is no  $\mathbf{V}_l'$  where  $\|\mathbf{V}_l'\|_1 < \|\mathbf{V}_l^*\|_1$  such that*

$$\|G(z') - G(z_0)\| \leq \delta, \quad z' \in \{z' | f_{l-1:1}(z') \in GR_{\mathbf{V}_l'}\}$$

*then we denote a set  $\mathcal{B}_{\mathbf{V}_l^*} = \{B_l^i | V_l^{*i} \neq 0, i \in \{1, 2, \dots, D_l\}\}$  as the smallest supporting generative boundary set (SSGBS).*

*In the same context, we denote the generative region  $GR_{\mathbf{V}_l^*}$  that corresponds to the SSGBS as the smallest supporting generative region (SSGR).*

It is non-trivial to explore entire combinations of generative boundaries to decide the optimal smallest supporting generative boundary set, as the combinations own to the exponential combinatoric search space.<sup>1</sup> To alleviate this expensive computation, we use the Bernoulli dropout approach [16] to determine the smallest supporting generative boundary set. We set the dropout function as  $\phi(h, \theta) = h \odot m$ ,  $m \sim \text{Ber}(\theta)$ , where  $\odot$  is an element-wise multiplication. We perform

<sup>1</sup>For instance, a common fully-connected layer with  $N$  output neurons has up to  $2^N$  generative boundary sets.



---

**Algorithm 1** Bernoulli Mask Optimization (BerOpt)
 

---

Input:  $z_0$ : a query,  $G(\cdot) = g_{L:1}(\cdot)$ : a DGNN model,

$l$ : a target layer

Output:  $\theta$ : the optimized Bernoulli parameter for SSGBS

- 1: Initialize  $\theta \in [0, 1]^{D_l}$
  - 2:  $h_0 = g_{l:1}(z_0)$
  - 3: **while** not converge **do**
  - 4:   Sample  $m \sim \text{Ber}(\theta)$
  - 5:    $h_m = h_0 \odot m$
  - 6:    $x_0 = g_{L:l+1}(h_0)$ ,  $x_m = g_{L:l+1}(h_m)$
  - 7:   Compute loss  $L(z_0, l, \theta)$
  - 8:   Update  $\theta$  with  $\nabla_{\theta} L$
  - 9: **end while**
  - 10: **return**  $\theta$
- 

optimization to determine  $\theta$  as minimizing the loss function  $L$ , which quantifies the degradation of generated output with the sparsity of Bernoulli mask  $m$ .

$$\begin{aligned} \theta^* &= \arg \min_{\theta} L(z_0, l, \theta) \\ &= \arg \min_{\theta} \|g_{L:l+1}(\phi(g_{l:1}(z_0), \theta)) - G(z_0)\| + \lambda \|\theta\|_1 \end{aligned} \quad (2.1)$$

We optimize the parameter  $\theta$  using a gradient descent to minimize the loss function in Equation (2.1). Then we acquire the smallest supporting generative boundary set  $\mathcal{B}_{\mathcal{V}_l^*}$  from the optimized Bernoulli parameter  $\theta$  with a proper threshold in the  $l$ -th internal layer. For each iteration in the optimization process, we perform the element-wise multiplication between  $g_{l:1}(z_0)$  and sampled mask  $m \sim \text{Ber}(\theta)$  to obtain masked feature vector, and feed it to obtain the modified generation. After that, we calculate the perceptual distance between an original generation  $G(z_0)$  and the modified generation.

After acquiring the optimal Bernoulli parameter  $\theta^*$ , at first, we define an optimal half-space indicator  $\mathbf{V}_l^*$  with the proper probability threshold (e.g.,  $p = 0.5$ ). We set the value of elements in  $\mathbf{V}_l^*$  as zero for suppressing the generative boundaries which have minor contributions to the final generation. That is,

$$V_l^{*i} = \mathbb{I}(\theta^* > p) \cdot \text{sign}(f_{l:1}^i(z_0))$$

where  $\mathbb{I}$  is indicator function. Representing SSGBS  $\mathcal{B}_{\mathcal{V}_l^*}$  and SSGR  $GR_{\mathcal{V}_l^*}$  is straightforward from the Definition 6 with  $\mathbf{V}_l^*$ . Figure 2.3 denotes the generative boundaries and the synthesized digit image of the smallest supporting generative boundary set without and with the optimized Bernoulli parameter  $\theta^*$  with  $p = 0.5$ . From Figure 2.3, we can observe that after selection of

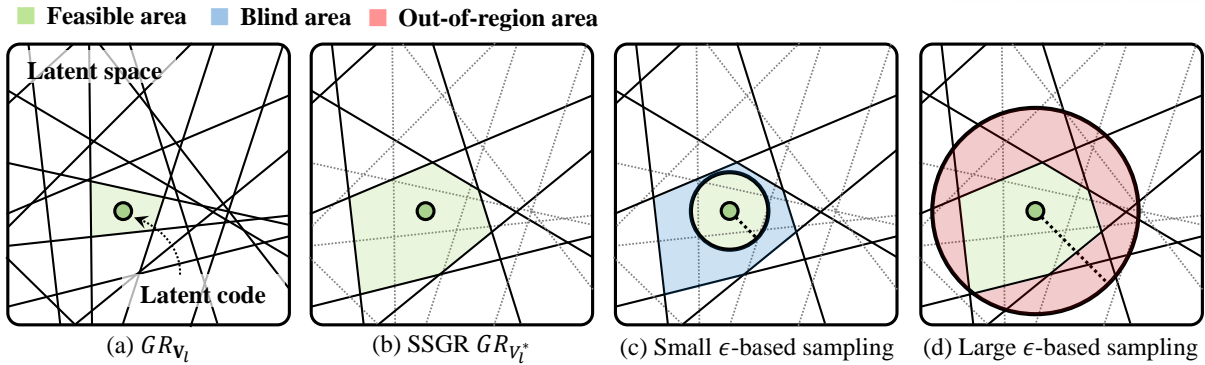


Figure 2.4: An illustrative example for the exploration range of  $\epsilon$ -based sampling with various magnitude of  $\epsilon$ . The green shaded area denotes the generative region for the given latent code (green dot). (a) The original generative boundaries and region. (b) SSGBS and SSGR after Bernoulli mask optimization. (c) The exploration range (black circle) for small magnitude of  $\epsilon$ . We can identify that there may exist a blind area (blue) for exploration in SSGR. (d) The exploration range (black circle) for large magnitude of  $\epsilon$ . We can identify that the sampling method can obtain the out-of-region latent codes (red) from SSGR.

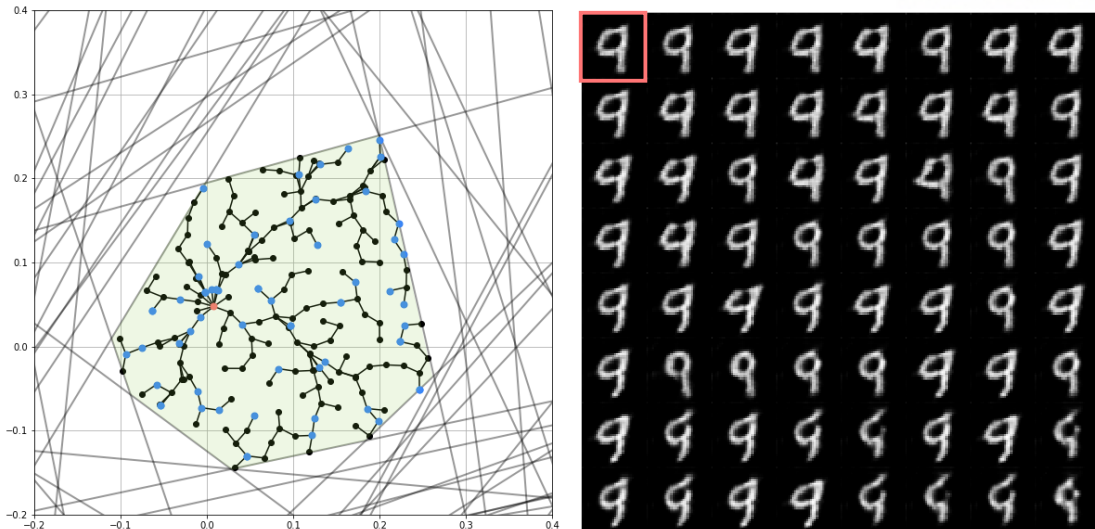
essential generative boundaries, the generative region is expanded with preserving the visual fidelity of the generation. As a result, we can identify that the effect of the suppression of minor generative boundaries on the generated output is not important.

## 2.2.4 Explorative Generative Boundary Aware Sampling

After determining smallest supporting generative region  $GR_{V_l}^*$ , we will obtain latent codes in this generative region and investigate the generated outputs of them. Because the  $GR_{V_l}^*$  represents a complicated and a non-convex shape, simple distance-based sampling methods (e.g.,  $\epsilon$ -based sampling) cannot guarantee exploration inside the  $GR_{V_l}^*$ . Figure 2.4 shows an illustrative example for the exploration range in two cases of the magnitude of  $\epsilon$  (small vs large). From Figure 2.4, we can confirm that the  $\epsilon$ -based sampling method is not proper to explore the obtained smallest supporting generative region  $GR_{V_l}^*$ .

To guarantee the relaxed neural representation sharing, we apply the generative boundaries constrained exploration algorithm motivated by the rapidly-exploring random tree (RRT) algorithm [17], which is invented for the robot path planning in complex configuration spaces. We name this modified exploration algorithm as generative boundary constrained RRT (GB-RRT). Figure 2.5(a) shows the explorative trajectories of GB-RRT in the smallest supporting generative region.

We refer to these entire procedure for the trained generator which is comprised of (1) finding the smallest supporting generative boundary set in an arbitrary internal layer, and (2) tree-like explorative sampling in the smallest supporting generative region with the collision condition induced by generative boundaries as Explorative Generative Boundary Aware Sampling (E-



(a) Trajectories of GB-RRT in SSGR. The green shaded area indicates SSGR. (b) Synthesized digit images from uniformly selected latent codes in the trajectories

Figure 2.5: An illustrative example for exploration of GB-RRT and synthesized digit images from obtained latent codes. (a) Visualization of explorative trajectories of GB-RRT for a given latent code (red dot) in the first hidden layer ( $l = 1$ ) of DCGAN trained with MNIST. (b) Generated outputs from uniformly selected latent codes in the trajectories (blue dot in (a)). The red box indicates the synthesized digit image from the given latent code.

GBAS).

## 2.3 Experimental Results

In this section, we provide experimental evaluations of the proposed method and empirical comparisons with various  $\epsilon$ -based sampling methods. We select three different deep generative neural networks; (1) DCGAN [1] with the wasserstein distance [29] trained with MNIST, (2) PGGAN [25] trained with LSUN-Church [30] and (3) CelebA-HQ [31].

In general, the  $\epsilon$ -based sampling obtains latent codes based on  $L_p$  distance metric. We select  $L_2$  and  $L_\infty$  distance as baselines, and sample latent codes in each  $\epsilon$ -ball centered at the given latent code. Although the value of  $\epsilon$  is manually selected in practice, for fair comparisons, we utilize the set of accepted samples and rejected samples,  $Z_{accept}$  and  $Z_{reject}$ , acquired by the E-GBAS to determined the magnitude of  $\epsilon$ . We define the average of accepted samples  $z_{avg}$  which can represent the middle point of the smallest supporting generative region, and we determine  $\epsilon_{L_2}$  with min/max distance between  $z_{avg}$  and  $Z_{reject}$  as,

$$\epsilon_{L_2} = \frac{1}{2} \left( \max_{z \in Z_{reject}} \|z_{avg} - z\| + \min_{z \in Z_{reject}} \|z_{avg} - z\| \right).$$

Figure 2.6 depicts an illustrative example of computing  $\epsilon$  in the DCGAN trained with MNIST. After  $\epsilon_{L_2}$  is determined,  $\epsilon_{L_\infty}$  is calculated to have the same volume as the  $\epsilon_{L_2}$ -ball. Figure 2.7



---

**Algorithm 2** Generative boundary constrained rapidly-exploring random tree (GB-RRT)
 

---

 Input:  $z_0$ : a query,  $GR_{\mathbf{V}_l^*}$ : SSGR

 Parameters:  $I$ : a sampling interval,

 $N$ : a maximum number of iterations,  $\delta$ : a step size

 Output:  $Q$ : samples in the SSGR

- 1: Initialize queue  $Q_0 = \{z_0\}$
  - 2: **for**  $i = 1 \dots N$  **do**
  - 3:   Sample  $z_i \sim U(z_0 - I, z_0 + I)$
  - 4:    $q_i = \text{nearest}(Q_{i-1}, z_i)$
  - 5:    $z'_i = (z_i - q_i) / \|z_i - q_i\| * \delta + q_i$
  - 6:   **if**  $z' \in GR_{\mathbf{V}_l^*}$  **and**  $\|z' - \text{nearest}(Q_{i-1}, z')\| > \delta$
  - 7:     **then**  $Q_i = Q_{i-1} \cup \{z'\}$
  - 8: **end for**
  - 9: **return**  $Q_N$
- 

---

**Algorithm 3** Explorative generative boundary aware sampling (E-GBAS)
 

---

 Input:  $z_0$ : a query,  $G(\cdot) = g_{L:1}(\cdot)$ : Generator,

 $l$ : a target layer,  $p$ : threshold for SSGBS selection

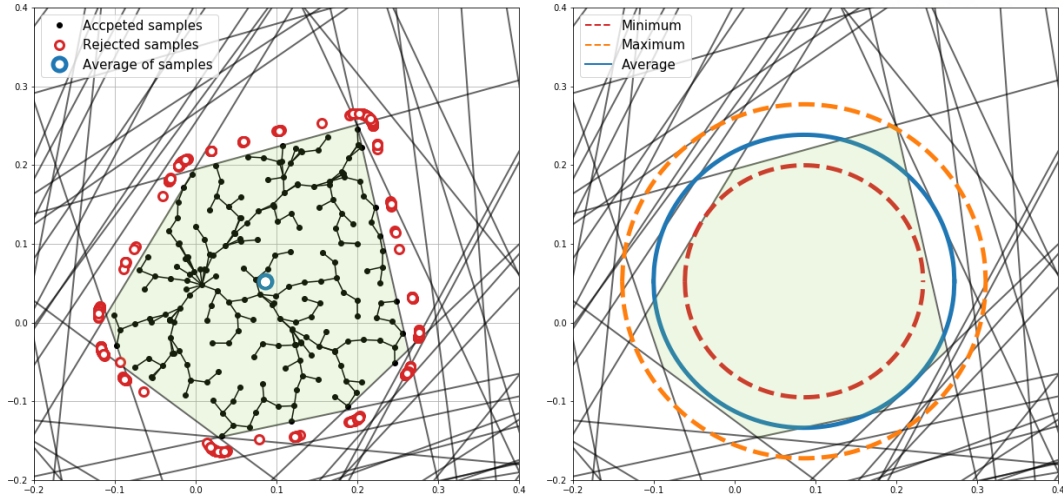
 Output:  $Z$ : a set of samples in the same SSGR of  $z_0$ 

- 1: Optimize  $\theta^* = \mathbf{BerOpt}(z_0, G, l)$
  - 2: Compute  $\mathbf{V}_l^* = [V_l^{*1}, \dots, V_l^{*D_l}]^T$   
       where  $V_l^{*i} = \mathbb{I}(\theta^* > p) \cdot \text{sign}(g_{l:1}(z_0))$
  - 3: Derive  $GR_{\mathbf{V}_l^*}$
  - 4: Sample a set  $Z = \mathbf{GB-RRT}(z_0, GR_{\mathbf{V}_l^*})$
  - 5: **return**  $Z$
- 

depicts the geometrical comparisons of each sampling method in the first internal layer ( $l = 1$ ) of DCGAN trained with MNIST.

### 2.3.1 Qualitative Comparison

At first, we verify how the synthesized images vary if the latent codes are inside or outside of the computed generative region. Figure 2.1 depicts an illustrative comparison between E-GBAS (blue region) and the  $\epsilon_{L_2}$ -based sampling (red region). As shown in Figure 2.1, we will mainly compare the synthesized images from the latent codes obtained from each sampling method, respectively. From a given latent code and a target internal layer, E-GBAS explores the smallest supporting generative region and gather latent codes which satisfy the relaxed neural representation sharing. Figure 2.8 shows the results of the synthesized images from latent codes obtained by E-GBAS and the  $\epsilon_{L_2}$ -based sampling. We empirically observe that the images synthesized by E-GBAS share more consistent attributes (e.g., composition of view and



(a) Accepted/rejected samples from E-GBAS. (b) Computation of  $\epsilon_{L_2}$  to close for SSGR.

Figure 2.6: An illustrative example of calculating the magnitude of  $\epsilon$ . (a) The accepted samples (black dots), rejected samples (red dots) and average of the accepted samples (blue dot) by E-GBAS in SSGR. (b) Visualization of the selection of  $\epsilon_{L_2}$  to make the area close to SSGR. The  $\epsilon_{L_2}$ -balls of each distance are depicted as red (min), orange (max) and blue (average of min/max distance) colored circle.

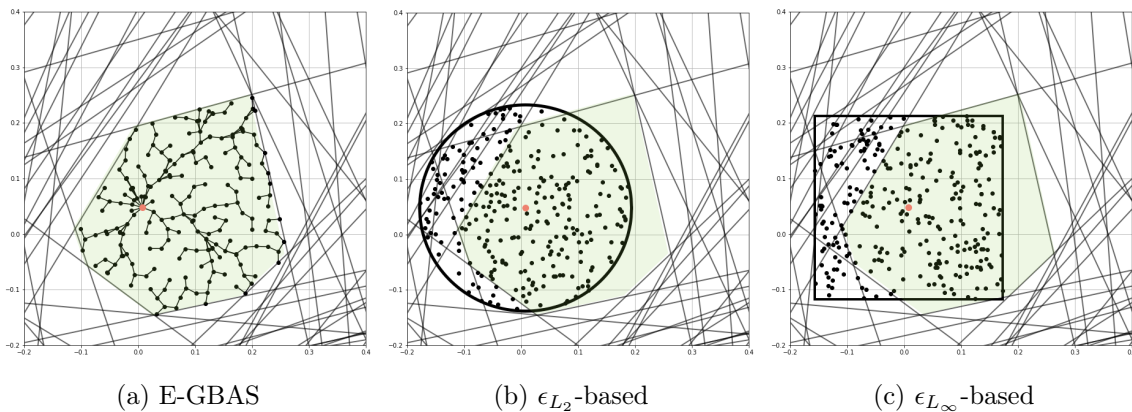


Figure 2.7: Geometrical comparison of (a) E-GBAS, (b)  $\epsilon_{L_2}$  and (c)  $\epsilon_{L_\infty}$ -based sampling methods. Although the two  $\epsilon$ -balls cover some area of the generative region, they cannot cover all of the generative region and have a possibility to include infeasible area. Whereas, the E-GBAS includes the only feasible area of the generative region for sampling.

hair color) which can be expected property of the relaxed neural representation sharing. For instance, in the first row of PGGAN trained with CelebA-HQ, we can identify the synthesized images share the hair color and angle of face with different properties such as hair style. In PGGAN trained with LSUN-Church, from the second row of images, we can identify that a set of images shares the composition of buildings (right-aligned) and the weather (cloudy).

We also try to analyze the internal generative mechanism of the generator for the depth of

internal layers by changing the target internal layer  $l$ . Figure 2.9 depicts the synthesized images and the standard deviations of these images obtained by E-GBAS in each target internal layer  $l$ . In Figure 2.9, we can identify that the variation of synthesized images is more localized when the target internal layer is determined to be deeper. For instance, in PGGAN trained with CelebA-HQ, the high standard deviations spread out in entire area of generation at the shallow layer, while the high variation values are focused on the edge of outlines in the generation. In PGGAN trained with LSUN-Church, we also observe similar pattern according to the depth of layers. We conjecture that the generative boundaries in the lower internal layer attempt to control an abstract and generic information (e.g., entire shape of face or the angle of scene), while those in the deeper internal layer tend to handle a concrete and localized information (e.g., mustache or the edge of wall).

### 2.3.2 Quantitative Results

**The Cosine Similarity of Activations in Discriminator** A deep generative neural network with the adversarial training scheme has a discriminator to quantify how realistic the output generated by the generator. During the training process, the discriminator learns features to evaluate the visual fidelity of synthesized images. In this perspective, we expect that generated outputs from the latent codes which satisfy the relaxed neural representation sharing have similar feature vector in the internal layers of the discriminator. We use cosine similarity between feature vector of synthesized images and the generation from the given latent code. The relative evaluations of the relaxed neural representation sharing for each sampling method are computed as the average of cosine similarities for synthesized images. When we denote a discriminator  $D(X) = d_{L:1}(X)$ , the given latent code  $z_0$  and the obtained set of latent codes  $Z$  from each sampling method, the average of cosine similarity of feature vector in the  $l$ -th internal layer is defined as the Equation (2.2). The operation  $d$  consists of linear transformations and a non-linear activation function such as LeakyReLU.

$$S_{d_{l:1}} = E_{z \in Z} \left[ \frac{d_{l:1}(G(z))^T d_{l:1}(G(z_0))}{\|d_{l:1}(G(z))\| \|d_{l:1}(G(z_0))\|} \right], \quad l \in \{1, 2, \dots, L\} \quad (2.2)$$

Table 2.1 denotes the results of computed the average of cosine similarity for each sampling method in each internal layer of the discriminator. We identify that the E-GBAS has the highest average of cosine similarity compared to baselines in entire deep generative neural networks.

**Variations of Synthesized Images** To quantify the consistency of attributes for the synthesized images, we calculate the standard deviations of generated images obtained by E-GBAS and various  $\epsilon$ -based sampling methods. The standard deviations are calculated as Equation (2.3) where  $P$  is a set of spatial position of each pixel. The computed standard deviations are shown in Table 2.2.

$$\sigma = \frac{1}{|P|} \sum_{p \in P} \left( \sqrt{E_{z \in GR_{\mathbf{V}}} \left[ (G(z) - E_{z \in GR_{\mathbf{V}}} [G(z)])^2 \right]} \right)_p \quad (2.3)$$

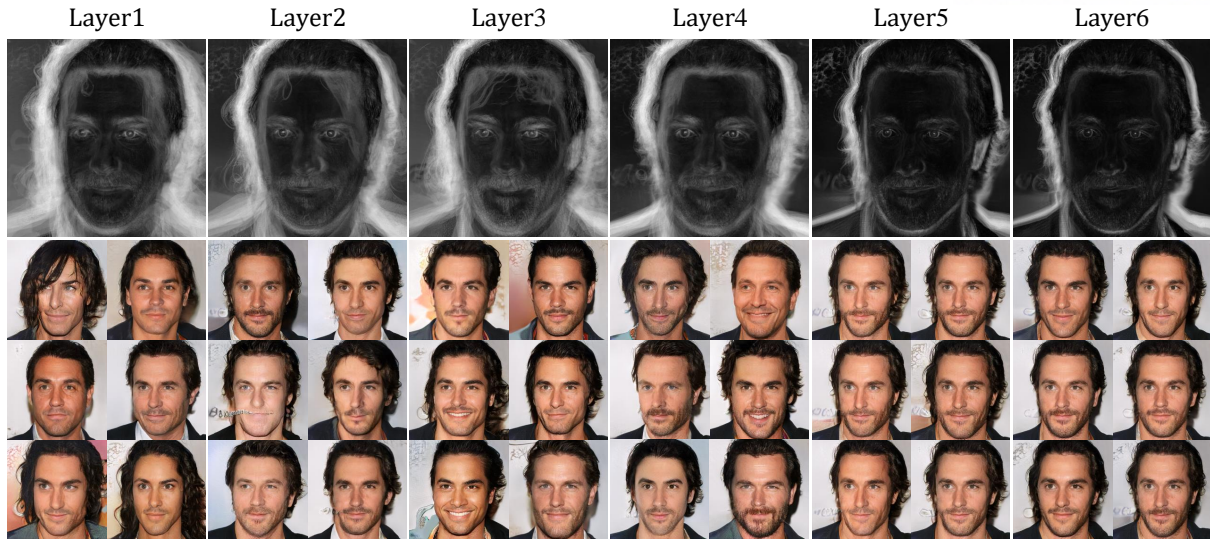




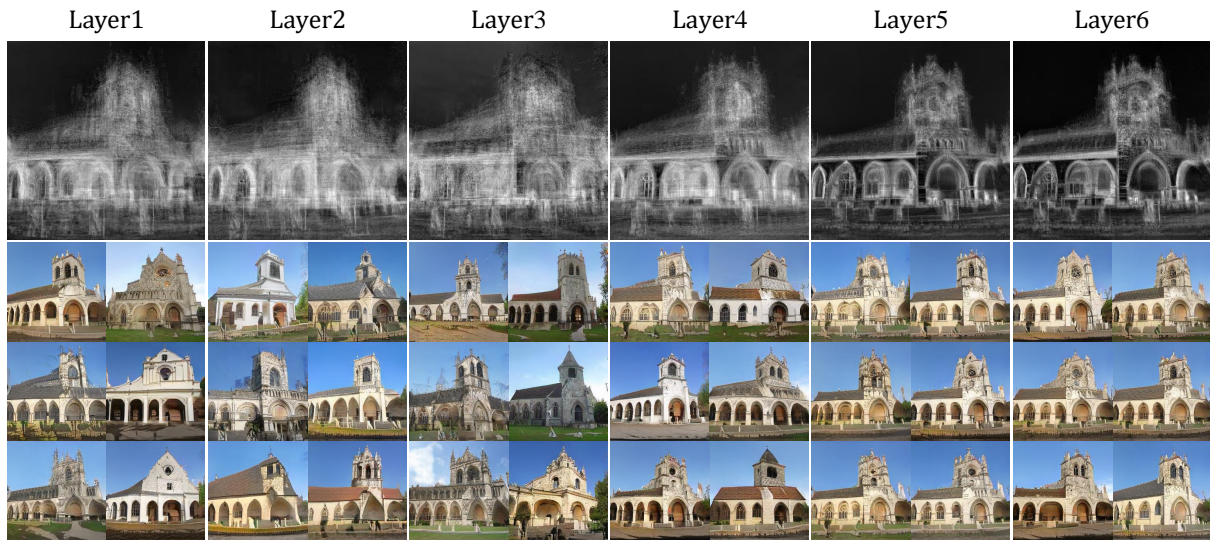
Figure 2.8: Synthesized images from the given latent code (left),  $\epsilon$ -based sampling (middle) and E-GBAS (right) in the three deep generative neural networks (DCGAN trained with MNIST, PGGAN trained with LSUN-Church, and PGGAN trained with CelebA-HQ.). We confirm that the synthesized images from latent codes obtained by E-GBAS have more consistent generative information compared to the  $\epsilon_{L_2}$ -based sampling.

We randomly select 10 latent codes and compute the average of standard deviations of generated images. From Table 2.2, we can identify that the proposed E-GBAS has lower values (i.e.,





(a) PGGAN trained with CelebA-HQ



(b) PGGAN trained with LSUN-Church

Figure 2.9: Examples of variations of synthesized images for each target internal layer. The first row depicts the standard deviations of synthesized images for each target internal layer. The second row depicts randomly selected synthesized images in each SSGR.

consistent with the given latent code) compared to various  $\epsilon$ -based sampling methods in entire target internal layers and three deep generative neural networks.

## 2.4 Related Work

**Generative Adversarial Networks** The adversarial training scheme between a generator and a discriminator has improved the quality and diversity of samples generated by deep generative neural networks [4]. Many generative adversarial networks (GANs) have been proposed to synthesize room images [1] and realistic face images [25, 32]. Despite of these advances, the

Table 2.1: Comparisons of the average of cosine similarity of feature vector for each sampling method in each internal layer of the discriminator. The number indicates the index of internal layer in a discriminator.

	Layer #	1	2	3	4
DCGAN-MNIST	$\epsilon_{L_2}$ -based sampling	0.722	0.819	0.864	0.991
	$\epsilon_{L_\infty}$ -based sampling	0.727	0.823	0.867	0.991
	E-GBAS	<b>0.747</b>	<b>0.838</b>	<b>0.878</b>	<b>0.992</b>
PGGAN-LSUN Church	$\epsilon_{L_2}$ -based sampling	0.578	0.602	0.957	0.920
	$\epsilon_{L_\infty}$ -based sampling	0.551	0.613	0.960	0.946
	E-GBAS	<b>0.578</b>	<b>0.637</b>	<b>0.967</b>	<b>1.000</b>
PGGAN-CelebA HQ	$\epsilon_{L_2}$ -based sampling	0.678	0.718	0.785	0.963
	$\epsilon_{L_\infty}$ -based sampling	0.684	0.720	0.789	0.965
	E-GBAS	<b>0.702</b>	<b>0.733</b>	<b>0.804</b>	<b>0.970</b>

Table 2.2: Standard deviations of synthesized images for each sampling method. The number indicates the index of target internal layer which generative boundary constraints are applied, where higher number is close to the final generation. The E-GBAS shows the lowest standard deviations compared to various  $\epsilon$ -based sampling methods.

	DCGAN-MNIST			PGGAN-Church			PGGAN-CelebA HQ		
#	2	3	4	2	4	6	2	4	6
$\epsilon_{L_2}$	0.0711	0.0718	0.0343	0.4951	0.4971	0.4735	0.5150	0.4994	0.4892
$\epsilon_{L_\infty}$	0.0722	0.0720	0.0344	0.4641	0.4322	0.3365	0.4859	0.4799	0.3384
Ours	<b>0.0694</b>	<b>0.0675</b>	<b>0.0323</b>	<b>0.3116</b>	<b>0.2558</b>	<b>0.1748</b>	<b>0.2980</b>	<b>0.2789</b>	<b>0.1446</b>

internal generative mechanisms of the GANs are not clearly understood yet. Recent work [1, 33] revealed that the relationship between the input latent space and the output space in a trained generator as manipulating of latent codes. The perturbations of latent codes can change attributes of the generated images. Internal units' generative roles in the trained generator are also investigated with the intervention method with pre-trained image segmentation network [3]. **Explaining the decision of deep neural networks** Sensitivity analysis aim to investigate which area of input mainly contributes to the decision of network. In sensitivity analysis, one can analyze the relative attribution between input features. The sensitivity can be computed by class activation probabilities [34], relevance scores [35] or gradients [20]. LIME [36] and SincNet [37] provide a surrogate model to interpret the decision of the trained deep neural networks. DeconvNet [38] provides visualization technique to interpret the internal trained features in the network with transposed convolution filter.

Example-based explanation is another approach to understand the decision of the network.

In example-based explanation, the examples which the trained network shows similar decision compared to the given input, are usually synthesized with some conditions. Activation maximization is one of example-based methods to synthesize the preferred inputs of neurons or according activation patterns in internal layers [18]. GANs [39] also have been used to generate examples, and the generated images are utilized to explain the trained features of the deep neural network. The reliability of examples for explanation of the network also has been discussed as considering the connectivity between the justified samples [19].

**Geometrical analysis of the internal mechanism in deep neural networks** Geometrical analysis tries to explain the internal mechanism of network based on the geometrical properties such as boundaries dividing the input space or manifolds induced by the boundaries. The depth of a network with non-linear activation function, such as ReLU, was shown to contribute to the formation of boundary shape [26]. This property makes complicated and non-convex decision regions surrounded by decision boundaries induced by internal layers, and it is related to representation power of deep neural networks. Although such decision regions are complicated, each region in deep neural network based classifiers is empirically shown to be topologically connected [40]. It has also been shown that the manifolds trained by deep neural networks and distributions over them are highly related to the representation capability of a network [41]. Verification of the decision region for DNNs with Lagrangian relaxed optimization is also performed [42]. Although this chapter is mainly related to preliminary work in terms of the internal regions for network, the main difference is that the previous work approximates the regions as convex polytope, but our method can handle the complicate non-convex regions directly.

## 2.5 Final Remark

In this chapter, we introduce the explorative sampling method as considering the generative boundaries to understand the internal generative mechanism of the deep generative neural networks. In particular, we provide novel concept, generative boundaries and generative region to define semantic sharing condition in the internal layer of the trained generator. To gather latent codes which satisfy the relaxed neural representation sharing condition in the complex and non-convex generative region, we devise GB-RRT which the generative boundaries are used to represent the collision constraint. We empirically verify that the obtained latent codes from E-GBAS share more consistent generative attributes compared to the various  $\epsilon$ -based sampling methods. From E-GBAS, we also qualitatively confirm that the relaxed neural representation sharing condition for the depth of the internal layers implies distinct generative attributes. Furthermore, because the concept of the proposed explorative sampling method uses the basic component of neural network (i.e., internal neurons), it can also be utilized to probe the decision region in the deep neural network based classifier.

## Chapter 3

# Automatic Correction of Artifact Generating Internal Units

The adversarial training scheme with deep generative neural networks (mainly called Generative Adversarial Networks, GANs) has achieved remarkable performance in the image generation field. Although GANs can generate images with high visual fidelity, there exists many synthesized images with defective visual areas which are called as artifacts. While most of the previous work tries to repair artifact generations as manipulating the given latent code, few investigate the role of internal units in a generator to improve the visual fidelity of artifact generations. In this chapter, we introduce a classifier-based method that automatically detects the internal units related to cause various types of artifact generations. We also devise the sequential correction method which adjusts the generation flow by modifying the activations of the detected artifact generating internal units to improve the visual fidelity of generations with preserving the original outline. We empirically verify that the proposed correction method outperforms the baseline correction method (i.e., FID-score based unit detection and correction), and show satisfactory results with human evaluation.

### 3.1 Introduction

In the GAN domain, the main focus of previous work is improving the generation performance by devising new structure of the generator or changing the training strategy. Although the various approaches enhance the generation performance of GANs, the networks still suffer from synthesizing outputs that contain unrealistic area called as *artifacts*, which makes them inadequate for being utilized in human centered mission-critical applications. As a result, investigating the reasons for artifact generations and devise possible solutions to improve the overall quality has proved to be important.

To handle this issue, recent work shows that the manipulating of latent code with the specific direction can be effective to remove artifact areas [13]. Based on understanding the represen-



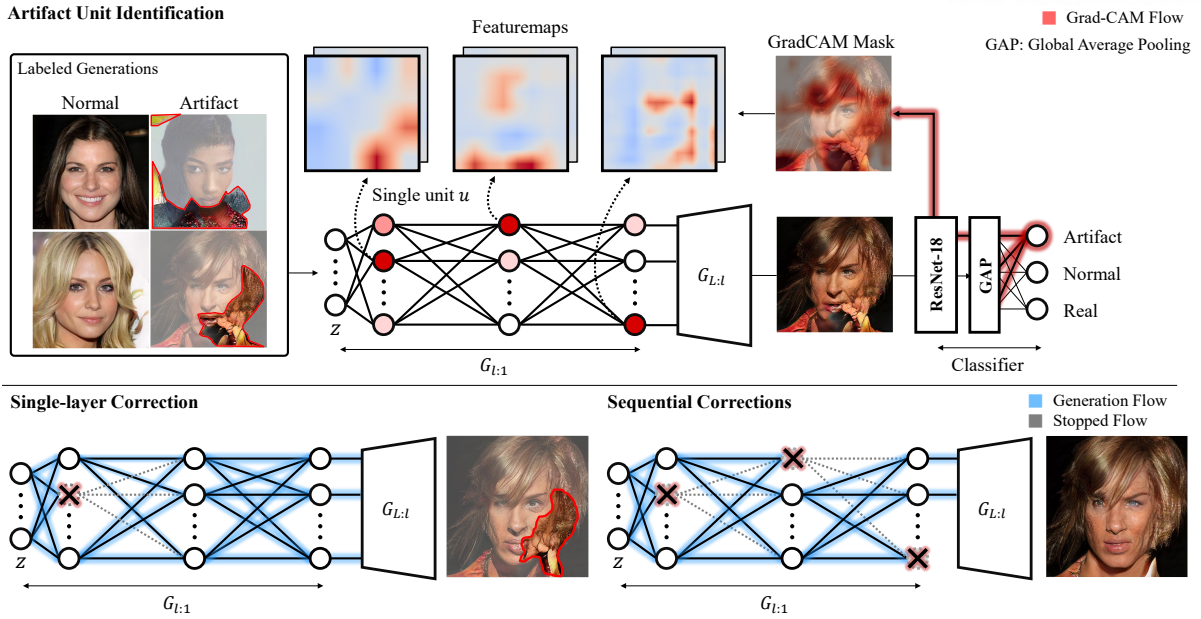


Figure 3.1: An illustration of the proposed classifier-based artifact generating internal unit detection and sequential correction method. (Top) Identification of the artifact generating internal units for each internal layer. (Bottom) the generation flow for single-layer correction and sequential correction method. In identification procedure, we measure the intersection-over-unions (IoUs) between defective areas (Grad-CAM mask) and an activation pattern of internal unit  $u$  for artifact generations. The averages of IoUs over artifact generations are used as the defective scores in each internal layer. The sequential correction method adjusts the generation flow of artifact generating internal units and improves the generation performance of GANs without retraining of the network.

tation of internal units from human annotation, ablation technique for the internal artifact generating unit is also proposed [3]. Unlike the previous methods, we identify artifact generating internal units by training a simple classifier from annotated generations. After training, we perform the Grad-CAM method on the trained classifier to obtain explanation mask for defective areas in the individual generation. Furthermore, we devise the global multi-layers based artifact generating internal unit ablation scheme which improves the visual fidelity of artifact generations with maintaining the plausible areas in the generations.

This chapter organizes as follows. The artifact generations in modern GANs and exist artifact generating internal unit detection are discussed to provide a detailed background to motivate the proposed method. Following up, we provide the empirical studies for the side-effect of the single-layer based correction. Then, we devise the sequential correction method which considers the consequent layers to suppress the side-effect and cover the various artifact concepts. Finally, we provide the experimental results in the artifact correction setting to demonstrate the effectiveness of the proposed method compared to exist work.

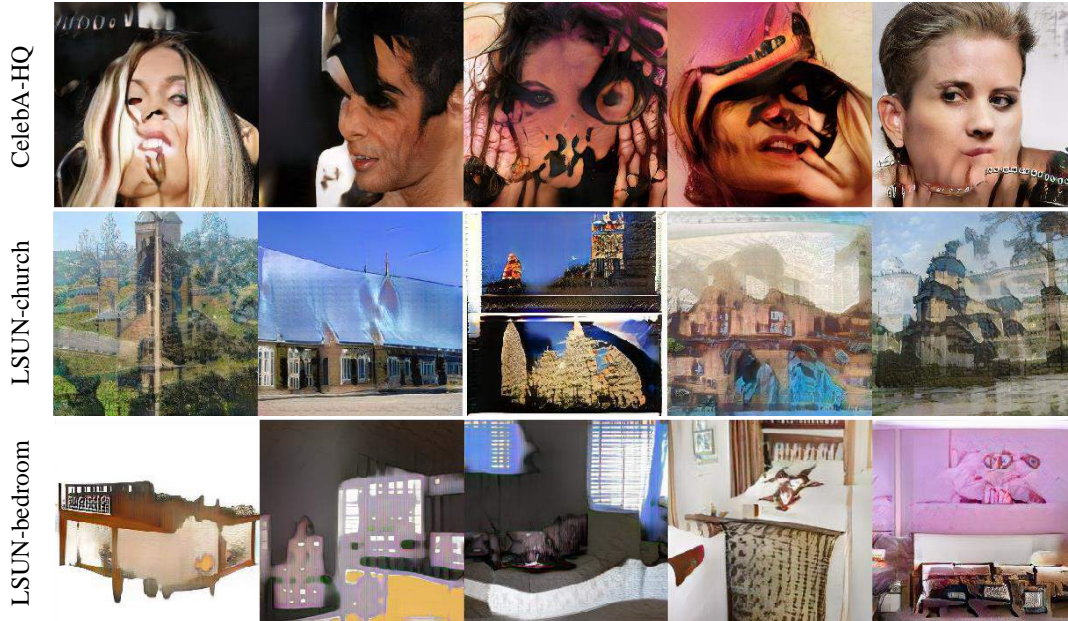


Figure 3.2: Illustrative examples for artifact generations of PGGAN trained with LSUN-Church (top), CelebA-HQ (middle), and LSUN-Bedroom (bottom).

Table 3.1: The ratio of artifact generations in PGGAN trained with CelebA-HQ, LSUN-Church, and LSUN-Bedroom.

	CelebA-HQ	LSUN-Church	LSUN-Bedroom
Artifact ratio	38.71 %	16.79 %	40.91 %

### 3.2 Ablation of Artifact Generating Internal Units in GANs

The term, *artifact*, has been denoted in the previous work [3, 43] to describe the generated images which have undesired (or unnatural) visual patterns. Figure 3.2 depicts illustrative examples of artifact generations in PGGAN trained with various dataset. For instance, we can identify that the distorted face part in the first row, and the church images have transparent parts. When we label the generations into *artifact* or *normal* generations, we observe non-negligible amount of artifact generations as listed in Table 3.1.

To evaluate the generation is artifact or not, one can suggest utilizing the output of a discriminator which we call *D-value* for the rest of this paper. Because the role of the discriminator is to distinguish the generation is real or fake. It is common to think the normal generations which look plausible will have D-values following the distribution of real training images, while artifact generations will be thought to have a distant distribution. However, it was not the case from our observations. Figure 3.3 depicts the histogram of D-values for the normal and artifact generations. In the Figure 3.3, we can identify that it is hard to classify the class of

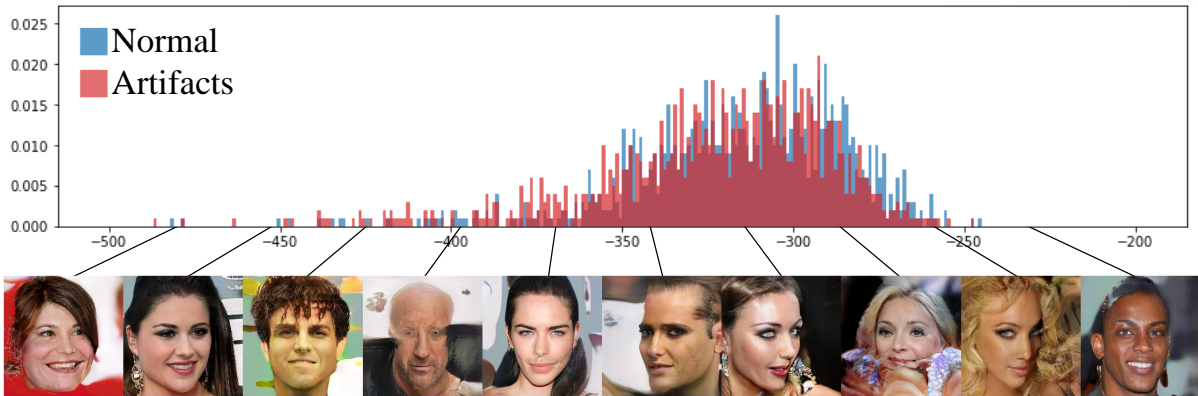


Figure 3.3: The histogram of D-values for each class of generation in PGGAN trained with CelebA-HQ. The black line denotes the corresponding D-value for each generation. The third and the last generations have similar defective background area, however, the corresponding D-values are largely different.



Figure 3.4: The representative generation (discuss in 3.2.3) and highly activated generations for unit 134 in layer 6 of PGGAN trained with LSUN-Church. Although the unit 134 seems to be related to night view, the high FID-score is measured.

generation (normal or artifact) based on the D-value, as two distribution are overlaid. We can also confirm that the degree of defectiveness of individual generation is not proportional to the D-value. From these observations, we believe that we need to dig deeper into the generator to identify and repair the artifact generations.

### 3.2.1 FID-Based Artifact Unit Identification

In previous work [3], artifact generations are also investigated as one type of object. To obtain the artifact generating internal units in an unsupervised manner, the authors use Fréchet Inception Distance (FID) which is a commonly adopted metric for measuring generation performance for the generative models [44]. For activation pattern of each featuremap unit, they compute the FID-score on 10K images which have the highest activation of the given featuremap unit among 200K generated images. Featuremap units which have the high FID-score are considered as the artifact generating internal units.

The authors empirically verify that the generation performance of the trained generator can be improved when the top-20 artifact generating internal units are zero-ablated in terms of FID-





Figure 3.5: Illustrative examples for the explanation masks from the Grad-CAM method with artifact class score. The masks highlight on the defective areas which the normal generations are hard to contain. The red color indicates high value compared to white color.

score. Although this FID-score based automatic correction has shown overall enhancement, our observation indicates there is still a chance to improve the identification of artifact generating internal units. Figure 3.4 depicts the internal unit with the 5th highest FID-score among 512 units in PGGAN trained with LSUN-Church. In Figure 3.4, we can identify that although the unit 134 has a high FID-score, the related generation concepts seem to be related to night view not artifact. This observation illustrates that the reason why we need a more delicate approach for the identification of artifact generating internal units.

### 3.2.2 Classifier-Based Artifact Generating Internal Unit Identification

To investigate the internal units which can cause high-level semantic artifacts, we perform annotation process for 2k synthesized images to normal or artifact generations. After annotation, we train a network to classify our dataset into three categories, namely: artifact, normal, and randomly selected real samples. We adopt an image classifier (e.g. ResNet-18) as our feature extraction module and add single fully-connected layer on top of it for classification of this downstream task. During the training, we freeze the parameters of the feature extraction module, and only optimize the parameters of the classifier. After training, we apply Grad-CAM method [20] to obtain a attribution mask for the defective areas. Such a mask highlights the areas that highly contribute for the model’s decision. For example, for artifact class, we can obtain artifact mask which the defective areas are highlighted. This operation will become more efficient than hand labeling for the defective areas, as the latter need a lot of human resources.

From the artifact mask by Grad-CAM method, we have a pseudo segmentation ground truth mask for defective areas. Comparing between this mask and activations of internal units can be utilized to detect the artifact generating internal units. For this purpose, we follow the principles in the prior work [45, 46]. For every internal unit  $u$  in the  $l$ -th layer of the generator, we compute  $A_u(z_x) \in \mathbb{R}^{H_l \times W_l}$  which is the activations for a given latent code  $z_x$

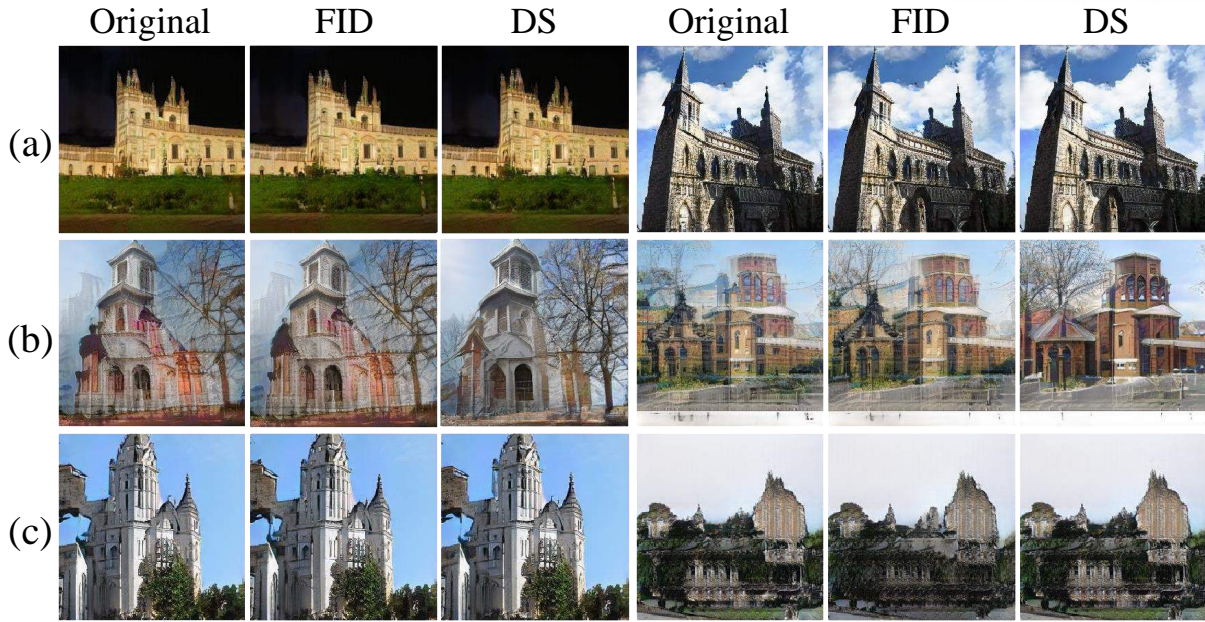


Figure 3.6: Illustrative examples for the single-layer correction results. We ablate top-20 internal units for the baseline correction method (i.e., FID-based unit detection, FID) [3] and our method (defective score based unit detection, DS). (a) Correction results for normal generations. We can identify that the ablations from both methods do not affect visual fidelity of generations. (b) While the baseline fails to correct generation, our method can remove the shadow effect with maintaining the original outline. (c) Both correction methods fail to repair the hole on the wall (left) or the texture error (right).

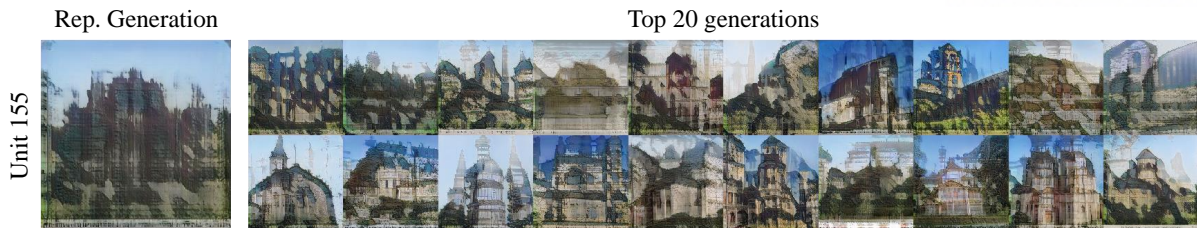
and the corresponding generation  $x$ . We threshold the activations with the quantile  $T_u$  such that  $P(A_u(z_x) > T_u) = \tau^1$ . This is computed as considering the activation value distribution of each unit for entire images. After applying the threshold, we perform bilinear upsampling to resize the size of the activations of each internal unit  $A_u(z_x)$ , and compute IoU with the pseudo artifact segmentation mask  $L_a(x)$ . To make the identification process global, we define the defective score which averages each IoU over a set of artifact generations  $X_a$ .

**Definition 7 (Defective Score)** *Let the set of artifact generations  $X_a$  and artifact segmentation mask  $L_a(x)$  and activation  $A_u(z_x)$  of unit  $u$  in the  $l$ -th internal layer for  $x \in X_a$  and  $x = G(z_x)$ . The defective score of unit  $u$  is defined as,*

$$DS_{l,u,a} = \frac{1}{|X_a|} \sum_{x \in X_a} \frac{|A_u(z_x) \cap L_a(x)|}{|A_u(z_x) \cup L_a(x)|}.$$

The defective scores for entire internal units in the  $l$ -th internal layer are defined as  $DS_{l,a} = \{DS_{l,1,a}, DS_{l,2,a}, \dots, DS_{l,D_l,a}\}$  where  $D_l$  is the number of internal units in the  $l$ -th internal layer. Finally, we can sort the defective scores, and select internal units with higher scores as candidates for artifact generating internal units.

<sup>1</sup>We empirically set  $\tau = 0.005$  in the experiments throughout the remained chapter.



(a) Unit for concrete generation concepts.



(b) Unit for mixed generation concepts.

Figure 3.7: Illustrative examples of the representative generation and highly activated generations for the given internal unit in layer 6 of PGGAN trained with LSUN-Church. (a) A unit seems to be related to concrete generation concepts (shadow artifacts). (b) A unit that does not have concrete generation concepts. The representative generation in this case seems to be blurred and it is non-trivial to define one clear generation concept for this unit.

Figure 3.6 depicts the results of ablation with the top-20 internal units in layer 6 on PGGAN trained with LSUN-Church comparing with the FID-based unit detection as the baseline. In the first row, we can confirm that both correction methods barely harm the plausible areas in the generations. It implies that the detected internal units are related to the defective areas and less correlated with the plausible or natural areas. For the shadow artifact in the second row, defective score based unit detection shows more reasonable correction performance compared to FID-based unit detection. However, both correction methods fail to repair in some artifact types as in the last row: (1) the hole on the church and (2) the texture error of the building.

### 3.2.3 Generation Concepts of Units in GANs

From the investigation described in previous Section, we can identify that a simple zero-ablation in single-layer is hard to guarantee to repair all types of artifacts. To analyze and interpret this phenomenon, we perform the analysis for the generation concept which each internal unit controls. To reveal the generation concept, at first, we randomly synthesize 20k generations and select the top-20 generations which maximize the magnitude of activations for each unit  $u$ . Then, from the collected generations, we observe that there are two categories for the type of handled generation concept of internal units (concrete vs mixed generation concept.). Figure 3.7 shows the illustrative examples of the internal units with concrete and mixed generation concepts. More examples for each category are available in Chapter 6.4.



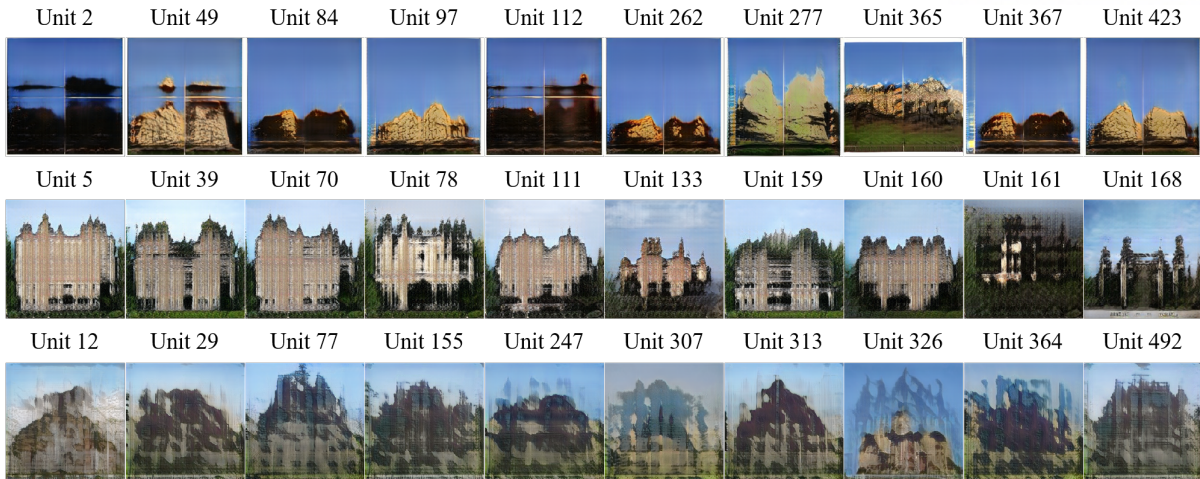


Figure 3.8: Illustrative example for the clusters of the representative generations in layer 6 of PGGAN trained with LSUN-Church. We can identify that the multiple internal units are related to one artifact concept. For instance, the texture artifact concept can be derived independently or together by unit 5 and unit 39.

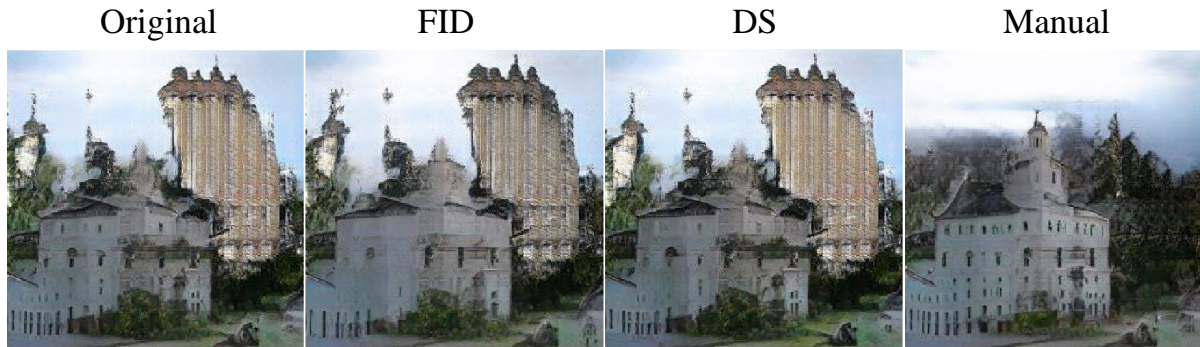


Figure 3.9: An illustrative example for correction results for texture artifact. In the Manual unit detection case, we select and ablate 15 units related to the texture artifact. The manual case only corrects the texture artifact and improve the details of church.

From the Figure 3.7 and Figures in Chapter 6.4, we can identify that some featuremap units seem to have concrete generation concepts, as the selected images share concrete semantic information which includes the type of artifacts. For instance, (a) in Figure 3.7, we can observe that the highly activated generations have shadow-style artifacts. However, for some featuremap units, it is hard to define one clear generation concept. To better investigation for such generation concepts, we additionally compute the mean featuremap amongst highly activated images to synthesize the representative generation for each internal unit.

From manual investigation, we observe the fact that multiple internal units can be related to one artifact concept. Figure 3.8 shows the representative generation which are clustered by the type of artifacts with the corresponding unit indices. In Figure 3.8, we can clearly confirm that each artifact concept is related to multiple internal units in a single-layer. This

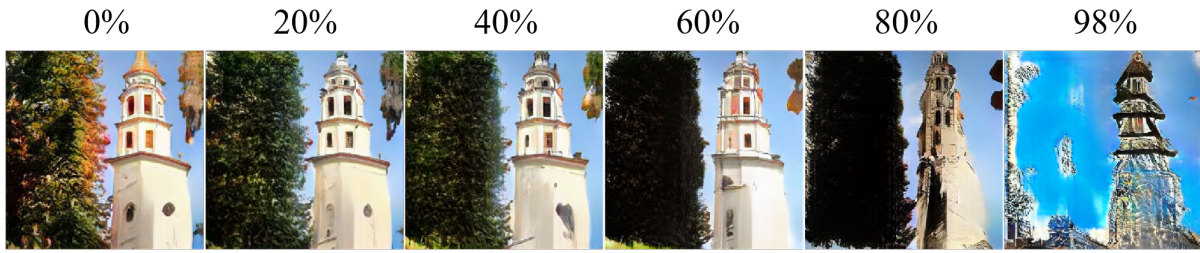


Figure 3.10: An illustrative example for a trade-off between defective and plausible area. The % denotes the ratio of zero-ablation for internal units. Although the size of the stain (top-right) is reduced when increasing the number of ablation units, the details of trees and church is degraded at the same time.

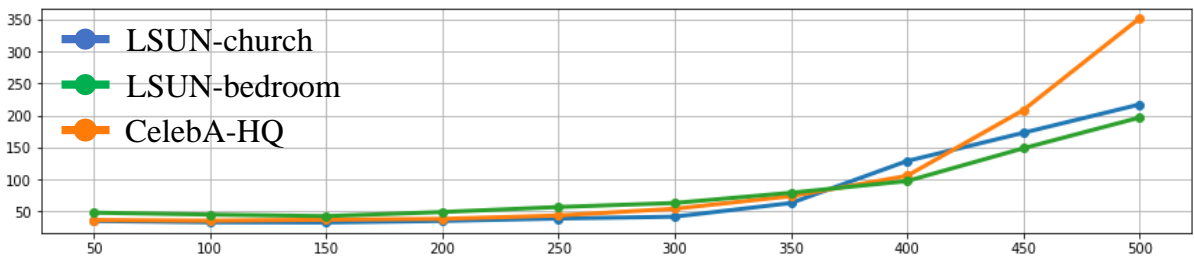


Figure 3.11: The computed FID-scores on the numbers of ablated units on layer 6 of PGGAN trained with various dataset. We can identify that when the number of zero-ablation for internal units increase, the FID-score exponentially increase.

observation requires that we need zero-ablation for a set of units, which includes most internal units sharing the type of artifact concepts, to guarantee the correction performance. Figure 3.9 shows the correction results comparing with manual internal unit detection. Because the top-20 internal units did not include entire texture artifact units in FID-score and defective score based detection, they cannot remove the texture artifact in the generation. However, when we manually perform zero-ablation for 15 internal units related to the texture artifact, the correction can be performed effectively. The manual unit selection can remove the tile artifact and improve the details of church, while others fail to repair the generation.

Although the intuitive approach to cover various types of artifacts is simply increasing the number of zero-ablated internal units, we observe a trade-off between removing defective areas and maintaining the plausible areas in the generation. Figure 3.10 depicts a trade-off between defective area and plausible information according to the number of zero-ablations. When we increase the number of units (% denotes the number of zero-ablation internal units in Figure 3.10), it gradually removes the defective spot on the top-right corner in the generation, but degradation for plausible parts also appears. For instance, we can identify that the quality of tree and the building is degraded among 60% – 98%. In addition, we plot the FID-scores according to the number of zero-ablation of internal units at layer 6 for the PGGAN across various datasets in Figure 3.11. We can confirm that the FID-score increases exponentially



---

**Algorithm 4** Sequential Correction
 

---

Input:  $z_0$ : a query,  $G(\cdot) = g_{L:1}(\cdot)$ : a generator,

$l$ : a stopping layer,  $DS_{l:1,a}$ : normalized defective scores for each layer,  $\lambda$ : a scaling factor,  $n$ : the number of ablated units

Output:  $X$ : the corrected generation

```

1:  $h_0 = z_0$ 
2: for  $k \leftarrow 0$  to  $l$  do
3:    $h_{k+1} = g_{k+1:k}(h_k)$ 
4:   for  $j \leftarrow \text{Top } 1$  to  $\text{Top } n$  do
5:      $h_{k+1,j} = \lambda(1 - DS_{k+1,j,a})h_{k+1,j}$ 
6:   end for
7: end for
8:  $X = g_{L:l+1}(h_{l+1})$ 
9: return  $X$ 

```

---

when we increase the number of internal units for zero-ablation.

### 3.3 Sequential Correction of Artifacts

In this section, we devise the sequential correction method to enhance the visual fidelity of individual generation by suppressing the side-effect of the single-layer based correction method. Let the generator  $G$  with  $L$  layers is decomposed into  $G(z) = g_L(g_{L-1}(\cdots(g_1(z)))) = g_{L:1}(z)$ , where  $z$  is a vector in the latent space  $\mathcal{Z} \subset \mathbb{R}^{D_z}$ .  $h_{l,u} = g_{l:1,u}(\cdot)$  denotes the values of the  $u$ -th internal unit in the  $l$ -th layer with  $g_{l:1}(z) \in \mathbb{R}^{D_l \times H_l \times W_l}$ . In general, the operation  $g_l(\cdot)$  includes linear transformations and a non-linear activation function such as LeakReLU. For the given latent code  $z$ , we sequentially adjust the activation of internal units for the consequent layers. The detailed procedure is described in Algorithm 4.

From the previous work [21], we can identify that shallow layers control the abstract generation concepts and deeper layers control the localized information in GANs. From these observations, we ablate the activations in each internal units for the shallow layers from the first layer to the stopping layer  $l < L$ . To prevent the degradation of semantic attributes of the given generation as discussed in Section 3.2.3, we reduce the magnitude of the original activations of featuremap units instead of the simple zero-ablation. Line 5 of Algorithm 4 states this soft ablation as,

$$h_{k+1,j} = \lambda(1 - DS_{k+1,j,a})h_{k+1,j}$$

where  $\lambda \in [0, 1]$  is the scaling factor and the pre-computed defective score  $DS_{k+1,j,a} \in [0, 1]$  is normalized.  $\lambda(1 - DS_{k+1,j,a})$  handles the relative generation flow of the selected featuremap units. Note that if the scaling factor  $\lambda = 0$ , the proposed method performs simple zero-ablations in consequent layers.

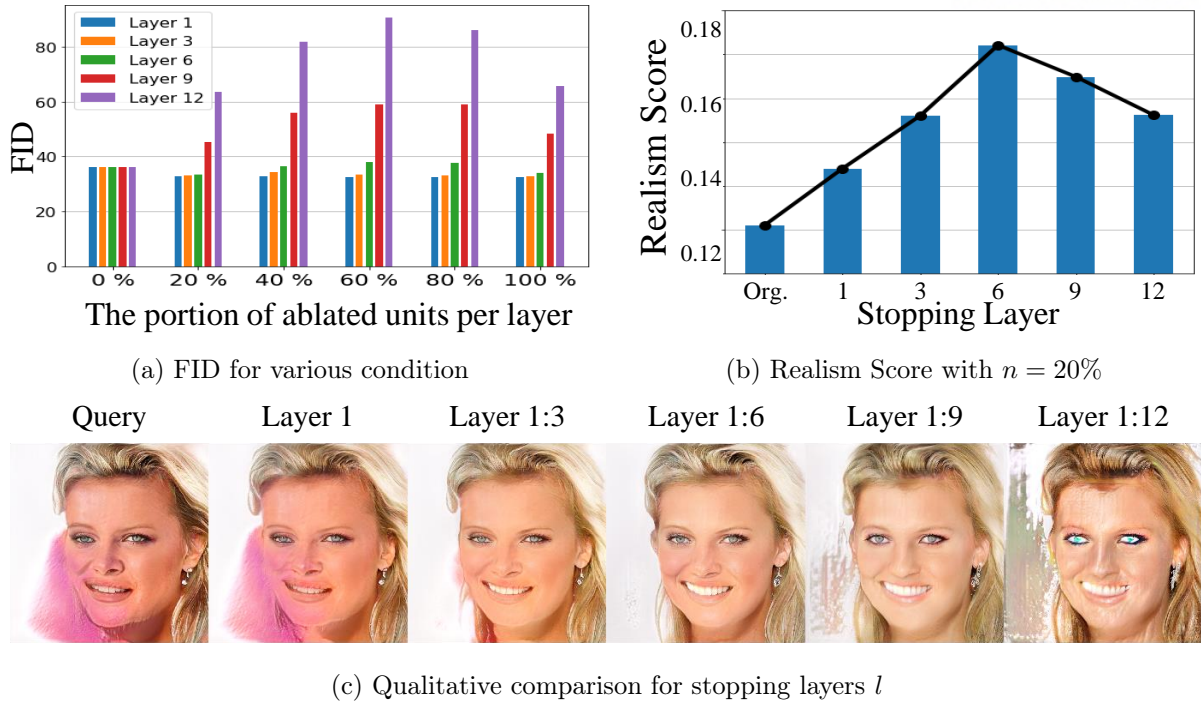


Figure 3.12: Quantitative and qualitative results over various hyperparameter settings (stopping layer  $l$  and the portion of ablated internal units  $n$ ). We can identify that when the stopping layer  $l$  is determined as the deeper layer, the visual fidelity of generation is degraded, while for shallow layer, the pink defective areas are still remained.

### 3.3.1 Exploration of Hyperparameters for Sequential Correction

To select optimal hyperparameters for the sequential correction method, we perform grid search in various combinations. At first, we measure the FID-score for corrected generations from the 1k artifact generations with various stopping layers  $l$  and the portion of ablation for internal units  $n$  in PGGAN trained with CelebA-HQ. Figure 3.12a shows the results of exploration for the hyperparameters. From Figure 3.12a, we can identify that the portion of ablation  $n = 20\%$  of internal units in each internal layer shows the lowest FID-score.

The FID-score sharply increases in layers 9 and 12 whereas the differences are minimal for the lower layers. We also measure another metric called *Realism Score* ( $RS$ ) [47] which can quantify the visual fidelity of individual generations, by varying stopping layers  $l$  with the fixed  $n = 20\%$ . Figure 3.12b shows the average of Realism Score over the 1k artifact generations. We can obtain the similar result as the FID-score case in terms of Realism Score. The stopping layer 6 with  $n = 20\%$  shows the best performance for the sequential correction method.

### 3.3.2 Local Region Correction

Although the devised artifact generating internal unit detection identifies the defective internal units in a global manner, we can also perform the sequential correction method for local areas additionally, by utilizing the artifact mask of defective areas for individual samples. We change

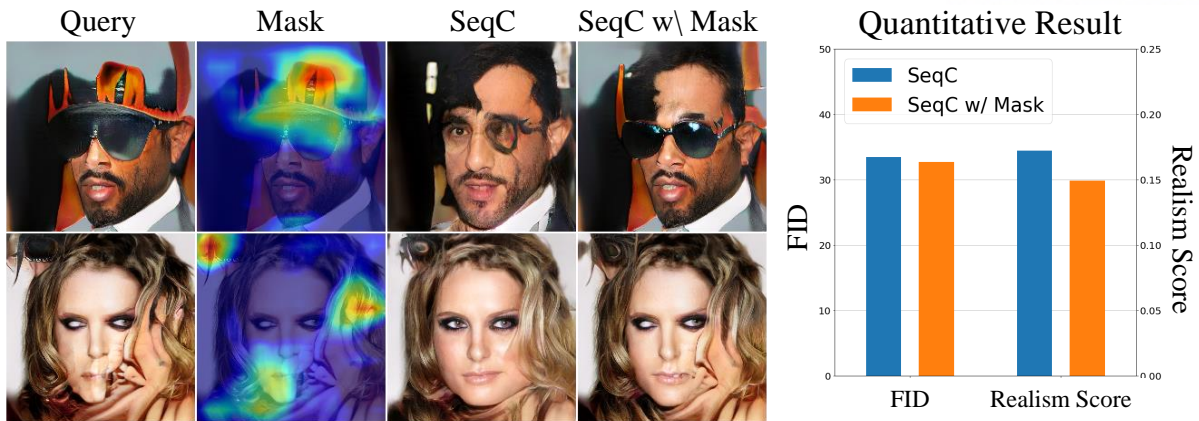


Figure 3.13: The effect of sample-based correction method for the sequential correction method (SeqC) in PGGAN trained with CelebA-HQ. We can observe that the areas which was not highlighted on by the artifact mask cannot be changed.

the reducing ablation weight  $\lambda(1 - DS_{k+1,j,a})$  to  $\lambda(1 - L_a(x))$  where  $L_a(x)$  is the downsampled artifact mask obtained by Grad-CAM method. In this sense, we can perform the sample-based sequential correction method, and expect that the correction result will be local. Figure 3.13 shows the illustrative examples for the sample-based sequential correction method compared to the global sequential correction method. From Figure 3.13, we confirm that the sample-based sequential correction method can minimize the change of unmasked information for individual generations. For instance, in the first row, we can identify that the sequential correction method repairs the glasses, the distorted hat, and the green colored background, while the sampled-based one repairs only the distorted hat aligned to the artifact mask. In second row, we also observe similar correction result as the first example. The sequential correction repair the overall quality of the face including the eyes and mouse, while the sampled-based sequential correction method change only the aligned areas.

### 3.4 Experimental Evaluations

This section provides the experimental results of the proposed method and empirical comparisons of various artifact correction methods. We perform correction on three PGGANs trained with LSUN-Church, LSUN-Bedroom, and CelebA-HQ datasets, respectively. For artifact correction experiments, we manually annotate generations from each generator and acquire 1k artifact generations and corresponding latent codes for each network. For obtained latent codes, we compute the activations of featuremap units through the model and ablate the detected featuremap units. Throughout the experiments in this chapter, we utilize the stopping layer  $l = 6$ , the number of ablation units  $n = 20\%$ , and the scaling factor  $\lambda = 0.9$ . Entire artifact correction experiments are conducted on the same 1k latent codes of the original artifact generations. The qualitative and quantitative results are provided in the following subsections.



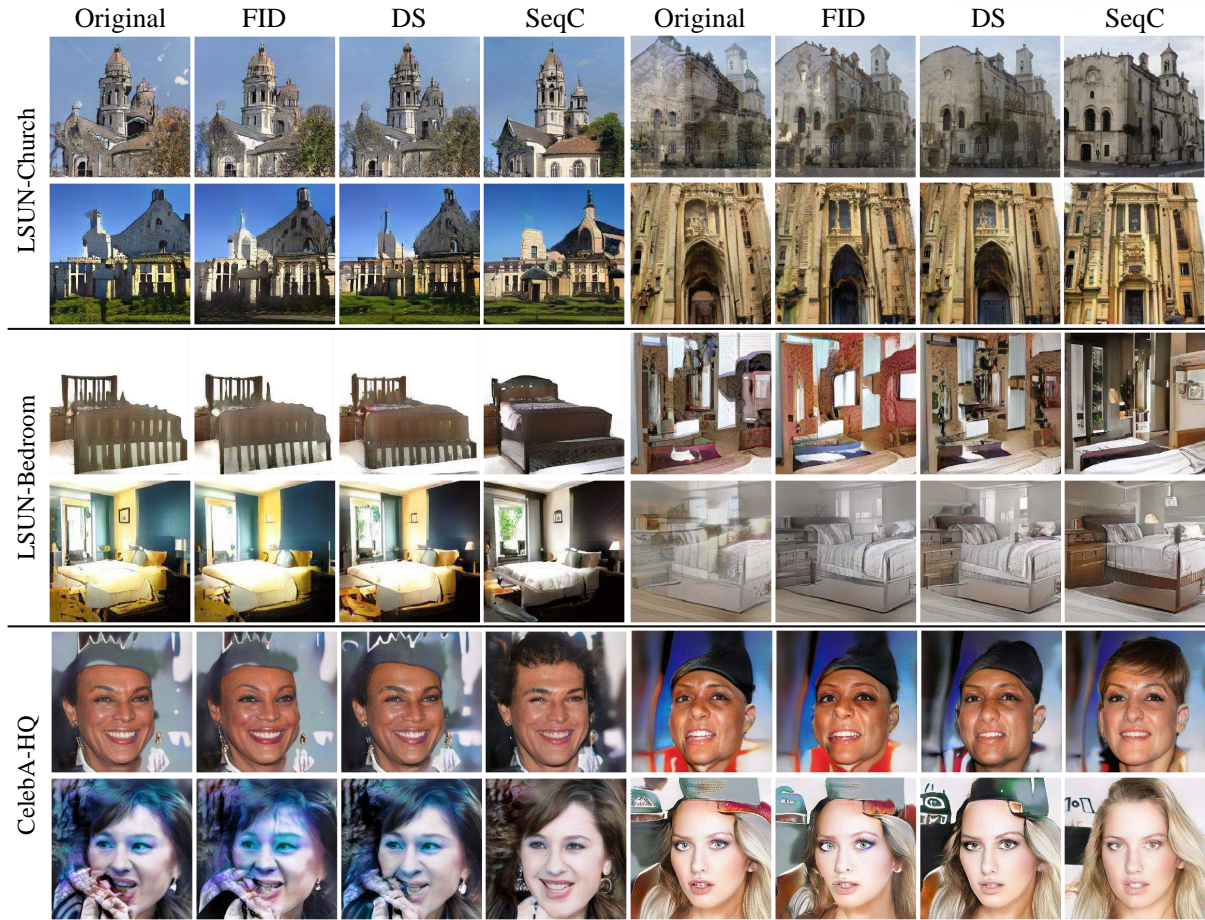


Figure 3.14: Illustrative examples for the various correction methods in the PGGAN trained with the various dataset. Original denotes the initial generation for the given latent code and FID denotes FID-score based unit detection with zero-ablation in the single-layer. DS and SeqC commonly utilize the defective score based unit detection. DS performs zero-ablation in single-layer and SeqC indicates the sequential correction method. We confirm that the sequential correction removes the defection areas effectively compared to other correction methods. More examples for correction are available in Chapter 6.2.

### 3.4.1 Qualitative Comparison

At first, we compare how the artifact generations can be improved in GANs trained with various dataset. Figure 3.14 shows the artifact correction results for each correction method. From Figure 3.14, we can confirm that the sequential correction method can remove the defective area effectively. In particular, adjustment of activations in the shallow layer are helpful to correct for the defective areas where the generation information is not clear. It means that the proposed sequential correction method can minimize the change of plausible areas and mainly focuses on the defective areas. For instance, in the fifth row with left hand side (row-5-LHS) for CelebA-HQ generation, we can identify that the constant green colored areas are clearly repaired with preserving information related to the normal attributes (basic information for face such as eyes

Table 3.2: FID-scores of corrected artifact generations for PGGAN trained with LSUN-Church, CelebA-HQ, and LSUN-Bedroom datasets. We identify that the sequential correction method shows the highest correction performance in three GANs.

Correction	LSUN-Church	CelebA-HQ	LSUN-Bedroom
Random	53.43	42.10	67.46
FID	40.66	44.37	48.48
DS	32.82	35.40	44.93
SeqC	<b>23.96</b>	<b>34.71</b>	<b>40.71</b>
Artifacts	46.95	36.16	61.17
Normals	22.37	29.80	29.15

and mouth). In the third row with left hand side (row-3-LHS) for LSUN-Bedroom generation, we can observe that the sequential correction method only remove the striped pattern and recover the details of bed, while others fail to correct. In the first row with right hand side (row-1-RHS) for LSUN-Church generation, we can observe that the proposed method can relax the shadow-style artifact and improve the details of church. As a result, from observations, we confirm that the proposed sequential correction method can repair the defective areas in the individual generations effectively.

### 3.4.2 Quantitative Comparison

We utilize the FID-score to quantify the improvements of visual fidelity of artifact generations for each correction method. In FID-score calculation, 1k of real training images are used. For better comparison, we provide the FID-scores for 1k original *artifact* generations and 1k *normal* generations in Table . As the baselines, we select random ablation and FID-based correction method [3], and summarize each score under *random* and *FID* row in Table . Then we perform zero-ablation for the global artifact generating internal units obtained by classifier-based approach (defective score based unit detection), and summarize it under *DS* row. The results for the sequential correction method are summarized under *SeqC* row. The results on each dataset are summarized under the LSUN-Church, CelebA-HQ, and LSUN-Bedroom columns, respectively. From Table 3.4.2, we can identify that the proposed sequential correction method (SeqC) shows the significantly improvement for the correction performance in entire GANs compared to exist work. We note that although the single-layer correction method based on the defective score (DS) has slightly high FID-score compared to sequential correction method, in entire models, it shows better correction performance than the FID-based correction method. It means that the proposed classifier-based unit identification is more delicate approach than FID-based detection to identify artifact generating internal units.

Table 3.3: Human evaluation results for the corrected generations in PGGANs trained with various dataset. The number in the parentheses denotes the standard deviations over raters.

Dataset	Corrected (%)	Improved (%)
CelebA-HQ	53.00 (4.20)	96.00 (2.00)
LSUN-Church	54.50 (0.90)	86.10 (6.30)
LSUN-Bedroom	46.80 (8.60)	95.50 (1.10)

### 3.4.3 Human Evaluation

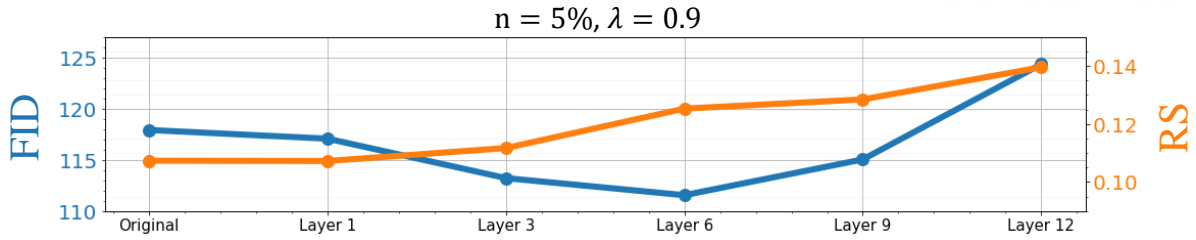
To support the improvement of visual fidelity for the corrected generations, we also perform human evaluation experiment. At first, we prepare 500 artifact generations for each dataset, and corresponding corrected generations obtained by the sequential correction method. To quantify the evaluation for the improvement of the visual fidelity consistently, we design the criteria related to the type of artifacts for each dataset to guide the decision of raters. A detailed description for each criterion of artifact type can be found in Chapter 6.1.2. The human evaluation process with pre-defined criteria is twofold: (1) Raters perform re-labeling for the corrected generations, and (2) determine the improvement for corrected generations (improved/not improved regardless of class) compared to original generations based on the criteria.

The human evaluation results for each dataset are summarized in Table 3.3. For PGGAN trained with CelebA-HQ, we can identify 53% out of 500 corrected generations computed by the sequential correction method for artifact-labeled generations are re-labeled as *normal*. While 47% are still annotate as the artifact, 97% of the total artifact generations have significant improvements for the defective areas or in the visual fidelity of the generation. For PGGAN trained with LSUN-Church, we can identify the class of 54.5% of generations are changed as *normal*. For PGGAN trained with LSUN-Bedroom, we can identify the class of 46.8% of generations are changed as *normal*. The improvement results for these two datasets also show the high enough ratio to validate the correction performance. As a result, from the human evaluations, we can confirm that the proposed method can correct artifact generations in the trained generator without post-training.

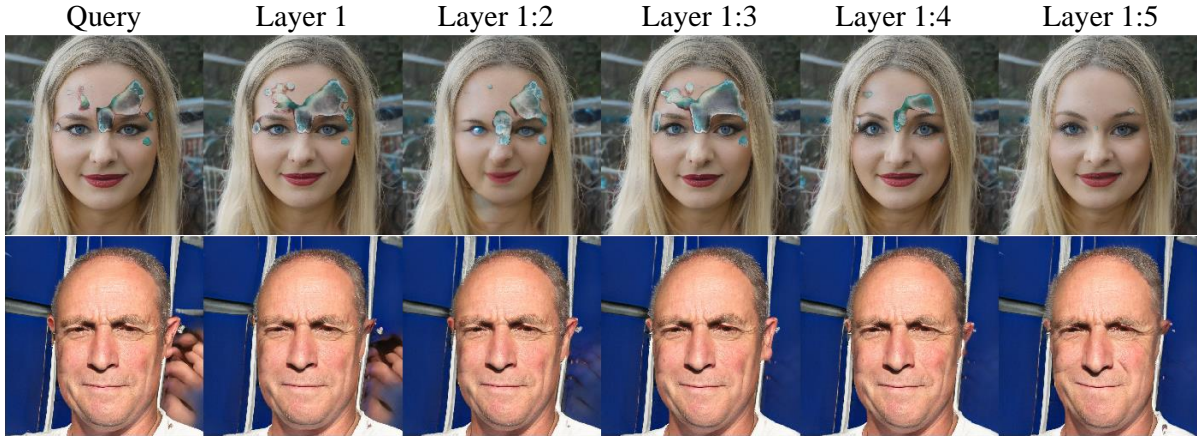
### 3.4.4 Generalization

Although the devised correction method is investigated on the generator with a conventional structure (i.e., PGGAN), our approach can be generalized for the recent state-of-the-art generators such as StyleGAN2 [48] or U-net GAN [49] which is a variation of BigGAN [5] with minor modification. Because of the distinct structure of each generator (e.g., Convolution kernels are changed for each generation in the StyleGAN2), it is hard to align the proposed framework to





(a) The quantitative results for various stopping layers  $l$ .



(b) The qualitative results for various stopping layers  $l$ .

Figure 3.15: Illustrative examples for corrected generations by the sequential correction method on StyleGAN2 trained with FFHQ. We choose 100 artifact generations, and calculate the FID-score and Realism Score (RS) for the corrected generations with various stopping layers  $l$ . We can identify that FID-score and Realism Score are improved when the stopping layer  $l$  increases up to the middle layer ( $l = 6$ ). In first row in (b), we can identify that the sequential correction method with middle stopping layer can remove the stains on the face effectively. In second row in (b), we also observe the stain in left-side is removed without drastic change of plausible areas.

Table 3.4: FID scores of 100 artifact generations and the corrected generations for StyleGAN2 and U-net GAN trained with FFHQ.

Model	Artifact	Corrected
StyleGAN2	117.91	<b>113.08</b>
U-net GAN	145.15	<b>143.85</b>

detect the artifact generating internal units in the global sense. However, we can obtain the relative defective score for each internal unit by individually comparing it with the Grad-CAM mask for implementing the sample-based sequential correction method.

Figure 3.15 shows the sample-based sequential correction method in the StyleGAN2 with various stopping layer  $l$ . Figure 3.15 (a) includes the FID-score and Realism Score and (b) depicts illustrative examples qualitatively. From Figure 3.15, we can confirm that the proposed method

has opportunity for generalization regardless of the structure of generator. More correction examples for StyleGAN2 and U-net GAN are available in Chapter 6.3.

### 3.5 Related Work

**Generative Adversarial Networks** Since the introduction of adversarial training scheme [50], the realism and diversity of synthesized images of the deep generative neural networks have improved steadily [6, 7, 51, 52]. In general, generator in GANs maps a sampled latent vector to a synthesized image in the generation process. While the synthesized images from the vanilla GAN model can be easily identified, modern GANs generate indistinguishable images from real image dataset. Despite recent advances, few researches has been performed in order to interpret the internal generative mechanism of GANs.

**Characterizing Deep Neural Networks' Internal Units** Various methods have been proposed to investigate and interpret the internal representation of deep neural networks [53, 54]. Explanatory heatmaps can be utilized to interpret the decision of network [20, 55]. The heatmaps emphasize which input features contribute to the predicted class of the trained networks. Recently, [45] proposes the *Network Dissection* framework to investigate the role of internal units of convolution neural networks. Authors obtain a dataset with semantic concepts, and analyze each internal unit in terms of the alignment of its activation map and the concept annotation. From the analysis, the fact that internal units of deep neural networks control object information such as tree or dome, is empirically verified.

**Artifacts in Deep Generative Neural Networks** Areas which include defective visual pattern can be observed in generated images by deep generative neural networks [3, 13, 43, 56]. [43] finds the phenomenon of checkerboard related defective visual pattern is induced by deconvolution (i.e., transposed convolution) operation and propose a simple resize-convolution based upsampling technique to resolve this issue. [57] exploits the distinctive inherent artifact generations, which appear in the warping process of generating deepfake, to discriminate the real video and fake. [13] train a linear classifier (e.g., support vector machine, SVM) with hand-labeled generations to divide the latent space of GANs into normal and artifact regions. The authors manipulate the latent codes toward normal region with the direction obtained from the trained hyperplane to gradually correct the artifact generations. [3] investigates artifact generating internal units with human supervision or FID-score for each internal unit. They visualize the highly activated generations for each internal unit, and determine that the internal unit is related to artifact or not by investigating the obtained generations. After that, they perform zero-ablation for artifact generating internal units in order to correct the artifact generations.



### 3.6 Final Remark

In this chapter, we devise the sequential correction method to enhance the visual fidelity of individual generations without additional training for the generator. In particular, we devise defective score that measures the alignment between activations of each internal unit and artifact mask obtained from the trained classifier. The sequential correction method detect artifact generating internal units based on the defective score in the unit identification process, and adjusts the generation information of these detected internal units for the consequent layers to improve the visual fidelity of individual generations. We empirically demonstrate that the proposed framework achieves satisfactory correction performance and suggest the opportunity of the generalization for the various structures of the generator.



Figure 3.16: Illustrative examples for the corrected generations with glasses or sunglasses. Although the defective areas are expected in background, the sequential correction method removes the glasses or sunglasses, while the background information is slightly changed.

While our sequential correction method has enhanced the visual fidelity of the artifact generations in terms of both FID-score and human evaluations, there exist some failure cases which the sequential correction method improves the visual fidelity of generations by simplifying the attributes. In Figure 3.16, we can observe that glasses/sunglasses are removed instead of completing defective areas. We conjecture that this phenomenon is induced by the undesirable features extracted from the trained classifier, which can be further analyzed in future work.

In addition, while the normal information are hardly changing dramatically in the sequential correction method, we can observe some failure cases which do not preserve the original attributes of generations. For instance, from the LSUN-Church correction on second row with right hand side (row-2-RHS) in Figure 3.14, we can identify that although the unclear pattern (blurred black entrance) is changed as clear entrance, the original structure of the church is also changed (The angle of church and details of pattern). In fourth row with left hand side (row-4-LHS) for LSUN-Bedroom case, we observe that although the details of bed are improved, the color balance of entire image is also modified at the same time. In sixth row with right hand side (row-6-RHS) for CelebA-HQ generation case, we also observe that the stains in top of head are removed and the blond hair semantic appears, while the angle of the face is changed at the same time. We

conjecture that these side-effects are came from a limitation of sequential correction method, as the identification of artifact generating internal units is performed depending on average over generations. To alleviate this problems, the fusion of global and individual sample-based correction can be considered for future work.

## Chapter 4

# An Unsupervised Way to Understand Artifact Generating Internal Units in GANs

Although Generative Adversarial Networks (GANs) have improved the generation performance significantly in the image generation field, generations with low visual fidelity called *artifact* still have been synthesized. As commonly used metrics to measure the performance of GANs mainly focus on the overall generation performance of the network, evaluation on the visual fidelity of individual generations or artifact detection is challenging. Although recent researches try to detect internal units which induce defective visual pattern and evaluate individual generations, these methods usually need additional resources such as a large number of training data to estimate the real data manifold or external networks to build supervision. In this chapter, we introduce the novel concept, *local activation*, and propose a metric based on the local activation to evaluate artifact generations in unsupervised manner. We empirically demonstrate that the proposed method can detect and correct artifact generations in GANs trained with various datasets. Finally, we discuss a geometrical analysis to partially understand the relationship between the locally activated neurons and low visual fidelity of individual generations in a perspective of the adversarial training framework.

### 4.1 Introduction

Despite remarkable improvements for GANs, generators sometimes synthesize defective outcomes which include unnatural visual pattern, and these generations are usually regarded as the undesirable outputs. In previous work [3, 14], these perceptually defective generations call as *artifacts*. Various metrics have been introduced to assess the generation performance of a trained generator [58–61]. However, it is difficult to assess the visual fidelity of each individual generation, as existing metrics mainly handles the distributional difference between the real dataset

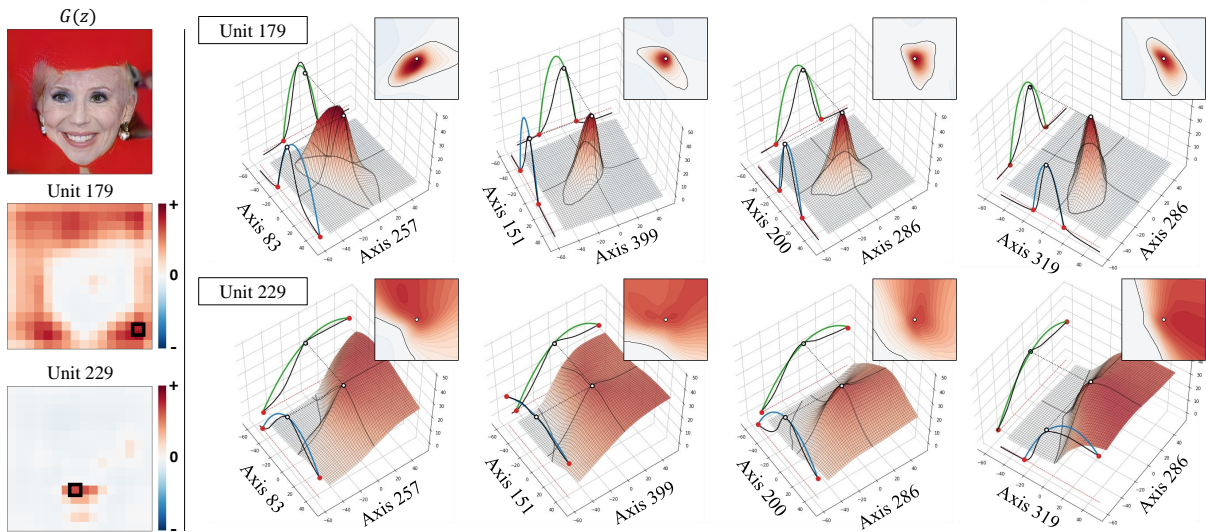


Figure 4.1: An illustrative example of the locally activated neuron in PGGAN trained with CelebA-HQ. We manually choose two internal units in layer 6 ( $\in \mathbb{R}^{512 \times 16 \times 16}$ ) which are expected to relate to low visual fidelity (red background in  $G(z)$ ) and mouth (plausible area). (Left) The black box in each internal unit denotes the spatial information of the selected internal neuron. (Right) The visualization of activation patterns in the latent space. The black solid line in each side indicates the activation values in the corresponding latent axis. The latent axes are randomly selected for the visualization. The red dots indicate the change points (Definition 8) and green/blue lines are the approximated curvature of local activation for each latent axis. The neuron in unit 179 (related to the low visual fidelity) has more locally activated pattern compared to the activation pattern of the neuron in unit 229 (related to mouth).

and the generations in the feature manifold obtained by external classifier such as VGG-16. To alleviate the distributional comparison, an alternative which utilizes the nearest neighbor based on similarity in the feature manifold [61] was devised to estimate the visual fidelity of individual generations. Although the effectiveness of this method is empirically verified in evaluation of the quality of individual generations, it requires a large amount of real data and a pre-trained external network for feature embedding to guarantee the reliability of the scoring process.

A few researches have been introduced to interpret the internal generative mechanism of GANs to detect or correct the individual generations in a perspective of the low visual fidelity. In GAN Dissection [3], the authors investigate the artifact generating internal units which mainly induce defective areas depending on a set of generations on which the specific internal unit is highly activated. The authors additionally enhance the visual fidelity of individual generations by zero-ablating the detected artifact generating internal units. A similar approach based on an external classifier is described in Chapter 3. In Chapter 3, we extract the artifact mask related to low visual fidelity for individual generations from the Grad-CAM method, and detect the artifact generating internal units based on these masks. On the other hand, manipulation of the

latent code in the latent space based on the binary linear classifier (e.g., support vector machine) has been introduced to correct the artifact generations [13]. Although these approaches can be used to assess the visual fidelity of individual generations, they still need additional resources such as a human annotation process and an external network.

This chapter organizes as follows. The empirical observations for activation pattern in the latent space to describe the properties of the internal neurons related to artifact are provided. From the observations, we devise the novel concept *local activation* to detect and correct the individual artifact generations in an unsupervised manner. Then, we experimentally verify that our method can detect and correct artifact effectively on PGGAN [6], and StyleGAN2 [62] trained with various datasets. Finally, we provide the geometrical analysis to partially reveal the relationship between the local activation and low visual fidelity of individual generations in a perspective of adversarial training framework.

## 4.2 Locally Activated Neurons in GANs

In this section, we introduce our main contribution, the novel concept, *local activation*, and analyze the relationship between low visual fidelity of individual generations and local activation. From previous work [3, 21] and Chapter 3, we can identify that each internal unit in the trained generator controls a specific generation concept (e.g., dome, doors, and etc) included in the final generation. Especially, an artifact generation which include defective areas can also be regarded as a type of object. From these observations, we confirm that it is possible to detect the artifact generating internal units which induce the low visual fidelity of individual generations. To follow this spirit and expand the framework, we mainly focus on internal neurons considered as the basic element of an internal unit.

### 4.2.1 Quantification of Local Activation

We empirically identify that the internal neurons which related to the defective areas usually show a bounded activation pattern in the latent space. Figure 4.1 depicts an illustrative example for an artifact generation and two internal units (one is related to artifact and the other is related to plausible information) in PGGAN trained with CelebA-HQ. Internal unit 179 is expected to highly correlate to the defective areas (red background) in the generation and unit 229 is expected correlate to the mouth information, which has the high visual fidelity. The right column of Figure 4.1 depicts 3D visualization of the section of activation patterns in the latent space for two internal neurons of unit 229 and unit 179, respectively. The internal neuron from unit 229 (black box) shows high activation over a large area in the latent space. In contrast, the internal neuron from unit 179 (black box) related to the defective areas, shows high activation only in a restricted space across various pairs of latent axes. Figure 4.2 additionally supports how the activation values are different between the artifact-related internal neurons and the plausible internal neurons. In the second column of Figure 4.2, activation values of the artifact-related

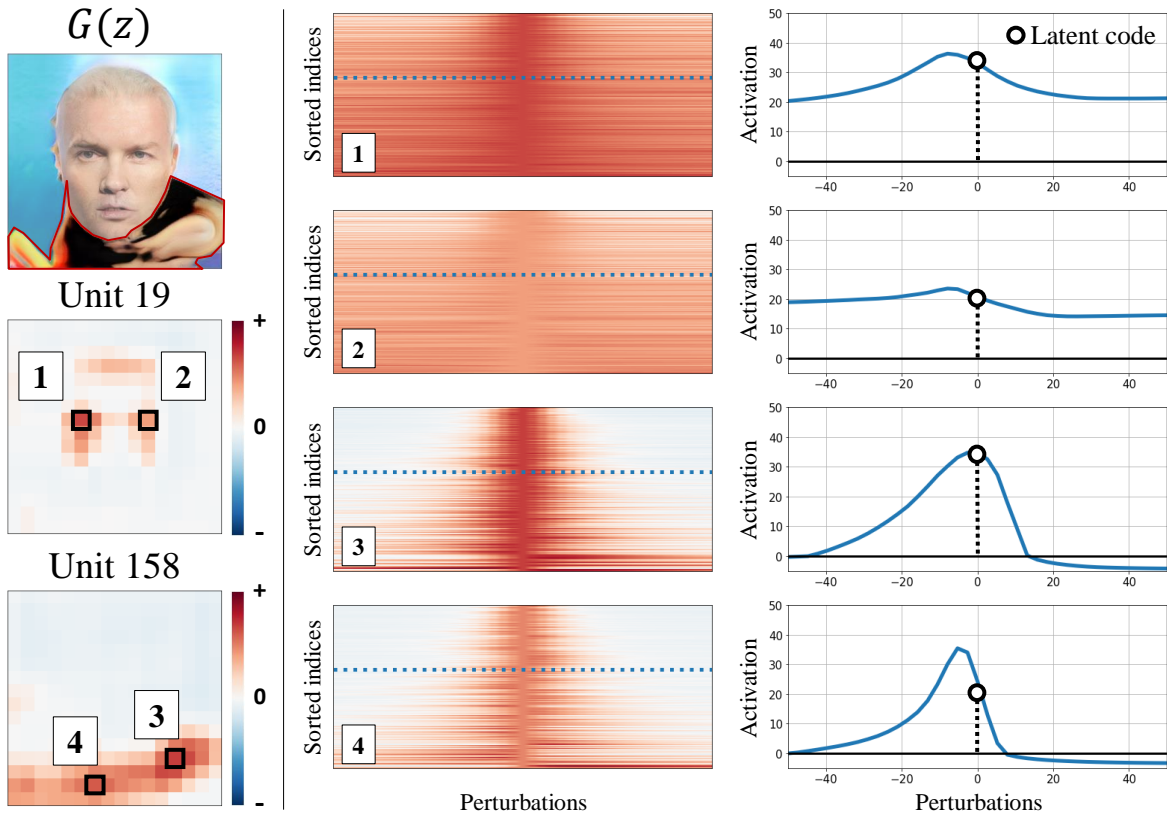


Figure 4.2: An illustrative example for activation patterns of the internal neurons in the manually selected internal units (unit 19 and unit 158) for the given latent code  $z$ . The internal neuron 3 and 4 are related to distorted jaw, while neuron 1 and 2 are related to eye information. The middle column depicts the visualization of heatmaps for the activation values nearby  $z$  where each row corresponds to each latent axis  $a$  in the latent space ( $a \in [1, 2, \dots, D_z]$ ). For intuitive visualization, we sort the rows based on the magnitude of activations in the descending order from top to bottom. The right column depicts the activation values for a specific axis (the dotted rows in the middle column) as the 2D visualization. X-axis denotes the perturbations and Y-axis denotes the corresponding activation value. The internal neurons related to the defective areas (3 and 4) are more locally activated compared to normal neurons.

internal neurons (neurons 3 and 4) are more sharply concave across the latent axes compared to the plausible internal neurons (neurons 1 and 2). The concave shape of the activation values suggests that the activations are bounded and concentrated around the given latent code.

As shown in Figures 4.1 and 4.2, we conjecture that the bounded activation pattern in the latent space may be related to the low visual fidelity of individual generation. We name this bounded activation pattern as *local activation* and the corresponding internal neuron as *locally activated neuron*. However, it is non-trivial to exactly quantify the local activation for an internal neuron in the latent space, because (1) commonly used generators have high dimensional latent space, and (2) the activation pattern is represented as a complicated non-convex shape in the



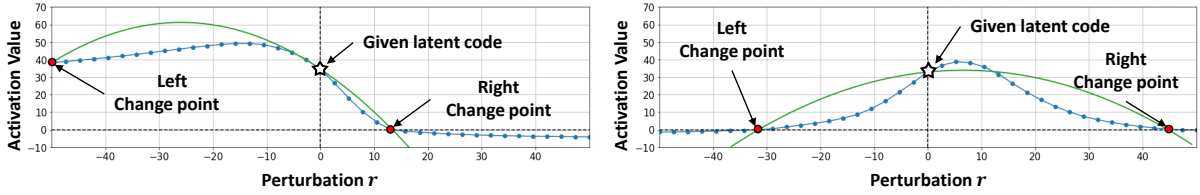


Figure 4.3: An illustrative examples for left/right change points and approximated curvature of activation values for the given activation values of perturbations. The white star denotes the activation value for the given latent code, and red dots denote the left/right change points. The green line denotes the quadratic approximation of the given activation values for perturbations.

latent space. To alleviate these challenges, we approximate the curvature of the local activation pattern with a line search for each latent axis within the empirical search bound.

Let the trained generator  $G$  with  $L$  layers be  $G(z) = g_L(g_{L-1}(\dots(g_1(z)))) = g_{L:1}(z)$ , where  $z$  is a vector in the latent space  $\mathcal{Z} \subset \mathbb{R}^{D_z}$ ,  $g_l(h_{l-1}) = \sigma(w_{g_l}^\top h_{l-1})$ ,  $h_{l-1} = g_{l-1:1}(z)$ , and  $\sigma(\cdot)$  is an activation function such as LeakyReLU or ReLU<sup>1</sup>. One can express the bias of each internal layer in the homogeneous representation with this unified equation by applying an additional dimension for the bias. For the  $i$ -th internal neuron  $g_{l:1}^i(z)$  of  $g_{l:1}(z)$ , we can compute an activation pattern with the line search over the perturbation range for each latent axis, and we define the left/right change points for each activation values as follows.

**Definition 8 (Change Point)** *Let the given latent code be  $z_0$ , the dimension index be  $d \in \{1, \dots, D_z\}$ , the search bound be  $R > 0$ , and the canonical basis be  $e_d = (0, \dots, 0, 1, 0, \dots, 0)^\top$  with the nonzero component at position  $d$ . For the set of change point  $P = \{r | g_{l:1}^i(z_0 + r \cdot e_d) = 0\} \cup \{-R, R\}$  for  $r \in [-R, R]$ , the right and the left change points of  $i$ -th neuron at the  $l$ -th layer are defined respectively as,*

$${}^r p = \min_{\forall r \in P; r \geq 0} (r) \quad \text{and} \quad {}^l p = \max_{\forall r \in P; r \leq 0} (r). \quad (4.1)$$

We note that if there are no points where signs of activation values are changed, the search bounds are regarded as the change points by Definition 1.

From the computed change points and the given latent code  $z_0$ , we approximate the curvature of the local activation values by calculating the curvature of the quadratic approximation of three points for each latent axis (the coefficient of the second degree coefficient). After obtaining the approximated curvature of each latent axis, we average these coefficients over the latent dimensions. The green and blue curves in Figure 4.1 and Figure 4.3 visualize the approximated quadratic functions to quantify the local activation.

**Definition 9 (Curvature of Local Activation (CLA))** *Let the given latent code be  $z_0$  and the left/right change points be  ${}^l p$  and  ${}^r p$  respectively as in Definition 8. The right slope is defined*

<sup>1</sup>There are various activation functions for deep neural networks, we only consider partial linear activation function in this chapter.



Figure 4.4: An illustrative examples for CLA based artifact generating internal unit identification. Bilinear upsampling for internal units are performed to overlay with the original generation. The red color denotes the high activation values.

as  $r_s = (g_{l:1}^i(z_0 + r_p \cdot e_d) - g_{l:1}^i(z_0))/r_p$  and the left slope is  $l_s = (g_{l:1}^i(z_0 + l_p \cdot e_d) - g_{l:1}^i(z_0))/l_p$  for a latent dimension  $d$ . With  $C_{i,l}(d, z_0) = (r_s - l_s)/(r_p - l_p)$ , the curvature of local activation for the  $i$ -th neuron in the  $l$ -th layer around the given latent code  $z_0$  is defined as,

$$\bar{C}_{i,l}(z_0) = \frac{1}{D_z} \sum_{d=1}^{D_z} C_{i,l}(d, z_0). \quad (4.2)$$

Although the definitions for CPs and CLA are constructed in the continuous space, we empirically use a grid search with search bound  $R = 30$ , dividing the search range by 20 for experiments throughout the paper. Details for the hyperparameter setting are discussed in Chapter 6.5.

Figure 4.4 depicts the internal units which have the highest average CLA over the internal neurons in each internal unit. We can identify that the activated areas for the internal units with a high CLA is semantically aligned with the defective regions in the given generation. For instance, the activations for the detected units are mainly related to the distorted hat on cat’s head.

#### 4.2.2 Dynamics for Local Activation during Training

In this section, we explore the dynamics of curvature of local activation (CLA) for the training process to verify the relationship between the low visual fidelity of generation and the magnitude of CLA. The experiments are performed on pre-trained snapshots of PGGAN trained with CelebA-HQ<sup>2</sup>. At first, we manually choose the internal unit which affects to the defective regions in layer 6  $\in \mathbb{R}^{512 \times 16 \times 16}$ . Then, we investigate the change of magnitude of CLA and the emerging process of defective regions during the training. Figure 4.5 depicts the emerging process of the defective region (low visual fidelity) in the generation and the corresponding magnitude of CLA.

<sup>2</sup>[https://github.com/tkarras/progressive\\_growing\\_of\\_gans](https://github.com/tkarras/progressive_growing_of_gans)



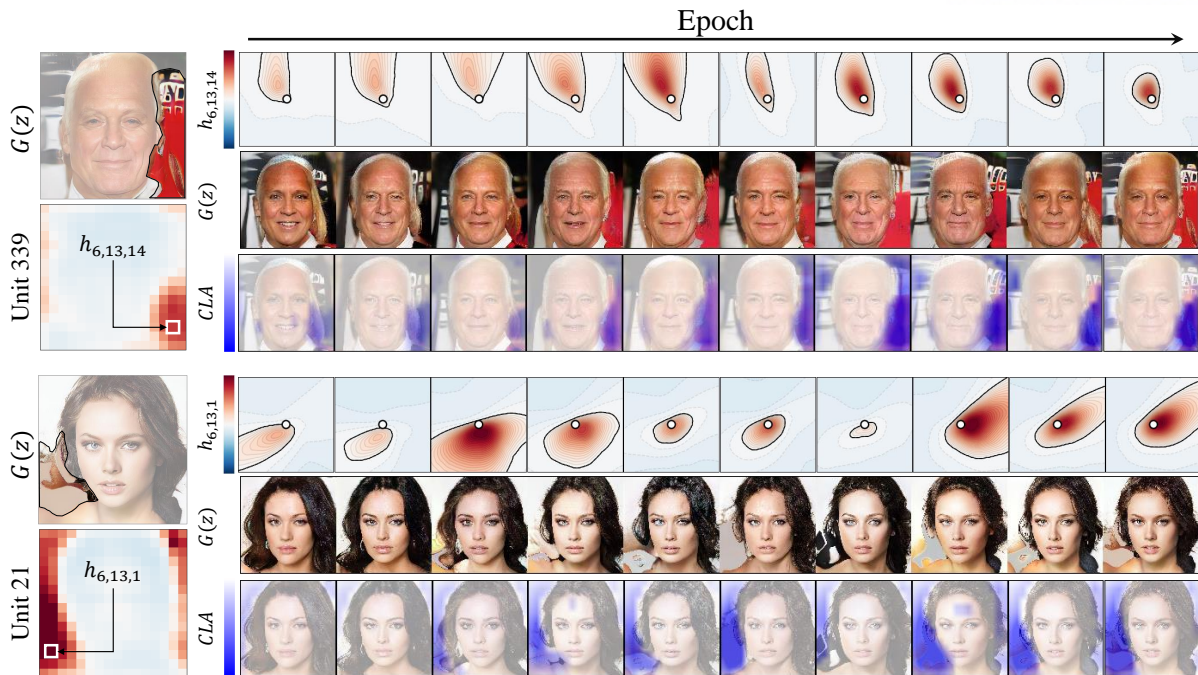


Figure 4.5: An illustrative example for the change of magnitude of CLA and emerging process of defective regions. (First row) The section of activation pattern in the latent space for the selected internal neuron (white box in each internal unit) with randomly selected axes. The red color denotes positively high value. (Second row) The generation  $G(z)$  for the fixed latent code. (Third row) Computed the magnitude of CLA for the selected internal neurons in each internal unit. The blue color denotes negatively high value. We can observe that increase of magnitude of CLA, change of activation pattern, and semantic information of generation are well-aligned during the training process.

In first example (the red object in right is defective region), we can observe that in the early stage of training, there is no artifact related object in the corresponding location and the magnitude of CLA is relatively low. However, during the training, the artifact related object appears and magnitude of CLA increase. We conjecture that the increase of magnitude of CLA is induced by shrinkage of activation area with increase of magnitude of activation value (see first row in Figure 4.5). From the second example, we also identify similar alignment as the first example. When the artifact related object appears, the magnitude of CLA increase. From observations, we empirically confirm that the magnitude of CLA can increase when (1) the activation region decreases as preserving the magnitude of activation value, or (2) the magnitude of activation value increases in a small activation area.

### 4.3 Experimental Analysis

This section provides experimental results for empirical relationship between the proposed concept, curvature of local activation (CLA), and low visual fidelity of individual generations. We

Table 4.1: The artifact detection and correction results on GANs trained with various dataset. We denote that higher/lower is better in RS/PPL, respectively.

Metric	Model	Dataset	Random	Low CLA	High CLA	Correction
RS	PGGAN	LSUN-Bedroom	1.028±0.003	<b>1.042</b>	1.017	1.002
		LSUN-Church	1.036±0.004	<b>1.059</b>	1.012	1.000
		CelebaA-HQ	1.076±0.004	<b>1.132</b>	1.011	1.018
	StyleGAN2	LSUN-Car	1.066±0.004	<b>1.084</b>	1.044	1.061
		LSUN-Cat	1.048±0.004	<b>1.071</b>	1.027	1.047
		LSUN-Horse	<b>1.056±0.004</b>	1.046	1.053	1.054
		FFHQ	1.075±0.004	1.077	1.069	<b>1.097</b>
PPL	PGGAN	LSUN-Bedroom	423.8±7.1	<b>243.9</b>	683.3	-
		LSUN-Church	356.4±7.9	<b>213.7</b>	558.0	-
		CelebaA-HQ	243.1±12.9	<b>114.9</b>	443.7	-
	StyleGAN2	LSUN-Car	1472.6±29.3	<b>920.7</b>	1938.9	-
		LSUN-Cat	1501.3±27.7	<b>1053.2</b>	2060.4	-
		LSUN-Horse	1207.4±21.5	<b>885.5</b>	1552.5	-
		FFHQ	484.9±19.2	<b>377.7</b>	596.2	-

select two different structure of GANs trained with various datasets. We use the pre-trained generator from the authors’ official github; (1) PGGAN trained with LSUN-Bedroom, LSUN-Church [15] and CelebaA-HQ<sup>2</sup>, and (2) StyleGAN2 trained with LSUN-Car, LSUN-Cat, LSUN-Horse [15] and FFHQ<sup>3</sup>. To evaluate visual fidelity for individual generations, we define CLA score as,

$$S_l(z) = \sum_i |\min(\bar{C}_{i,l}(z), 0) * \text{sign}(\max(h_{l,i}, 0)) + \max(\bar{C}_{i,l}(z), 0) * \text{sign}(\min(h_{l,i}, 0))| \quad (4.3)$$

where  $l$  is target internal layer and  $z$  is the given latent code. The CLA score mainly considers the degree of concavity/convexity for positive/negative activation, respectively. If the CLA score of the given generation  $G(z)$  is larger than other generations, we can expect that  $G(z)$  has a low visual fidelity.

### 4.3.1 Qualitative Results

To demonstrate the relationship between CLA score and low visual fidelity of individual generations, we design artifact detection game. At first, we randomly sample 10k latent codes without truncation trick for each GAN, and compute the CLA score on layer 4  $\in \mathbb{R}^{512 \times 8 \times 8}$  for each generation. We select the bottom/high 1k samples as low/high CLA groups based on the CLA score, respectively.

<sup>3</sup>StyleGAN2:<https://github.com/NVLabs/stylegan2>

### Artifact Detection Game

At first, we compare the visual fidelity of each group qualitatively. Figure 4.6 shows the results of artifact detection game in GANs trained with various dataset. We confirm that the generations in the high CLA group have lower visual fidelity compared to generations in the low CLA group over almost dataset. For instance, in PGGAN trained with CelebA-HQ, we can observe that generations in the high CLA group usually have some defective visual pattern (e.g., distorted jaw or colored stain), while generations in the low CLA group represent clear face information. In StyleGAN2 trained with LSUN-Car, we can also verify that the generations in the high CLA group do not represent clear generation information for the car object, while generations in the low CLA group include clear car information. More results of the artifact detection game are available in the Chapter 6.7 - 6.13.

### Artifact Correction

To additionally demonstrate the relationship between locally activated neurons and the low visual fidelity of individual generations, we perform a sequential correction which discussed in Chapter 3 on the high CLA group. Instead of utilizing a pre-trained classifier with annotated dataset to detect the internal artifact generating units, we utilize the average magnitude of CLA over the internal neurons in each internal unit as the defective score. For sequential correction, we determine the hyperparameters as follows: stopping layer  $l = 4$ , the number of ablation units  $n = 20\%$ , and the maintain ratio  $\lambda = 0.9$ . We calculate the Realism Score (RS) after correction (see last column in Table 4.1.) to quantify the changed visual fidelity for individual generations. Figure 4.7 shows illustrative examples for the unsupervised correction, and the column *Correction* in Table 4.1 summarize the changed Realism Score for the high CLA group. From Figure 4.7, we can identify that when the generations contain severe defective visual pattern (e.g., simple constant color without clues for the object. See the first row in Figure 4.7), as in the cases of PGGAN trained with LSUN-Bedroom or LSUN-Church, we may need a more sophisticated method than sequential correction method to correct the artifact. Nevertheless, we can enhance the visual fidelity of individual generations qualitatively and quantitatively over most GANs in the experiments with the unsupervised sequential correction method. From results of the artifact detection game and the correction experiment, we believe that the locally activated neurons are strongly related to the low visual fidelity of the individual generations in GANs.

#### 4.3.2 Quantitative Results

To quantify the visual fidelity for categorized groups (low/high CLA groups), we calculate Realism Score (RS) and Perceptual Path Length (PPL) for each group; (1) low CLA group and (2) high CLA group and (3) random selection group (30 trials) which means the baseline. We utilize 30k real images for each dataset to calculate Realism Score with the neighborhood constant





Figure 4.6: Results of artifact detection game in GANs trained with various dataset. We visualize bottom-24 (low CLA score) and top-24 (high CLA score) generations for qualitative comparison. We verify that the generations with the high CLA score have lower visual fidelity compared to the the generations with the low CLA score. Chapter 6.7 - 6.13 provide more examples for results of artifact detection game.



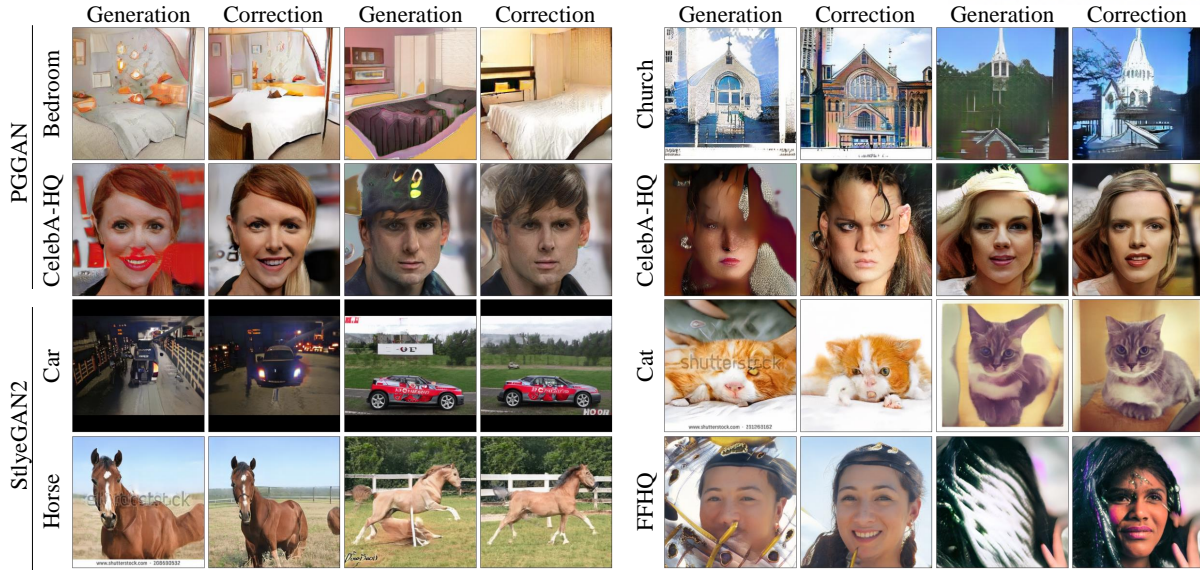


Figure 4.7: An illustrative examples for the artifact correction results on GANs trained with various dataset for high CLA group. More correction examples are available in Chapter 6.14.

$k = 3$  for computation. For Perceptual Path Length, we compute interpolation in the  $z$  latent space with  $\epsilon = 10^{-44}$ . Table 4.1 denotes the measured values for each group in GANs trained with various dataset. We can verify that (1) the high CLA groups have low Realism Score and high Perceptual Path Length, and (2) the low CLA groups have high Realism Score and low Perceptual Path Length compared to random groups in almost GANs. The results consistently demonstrate that the proposed concept, CLA score, can effectively evaluate the generations with the low visual fidelity. We emphasize that the proposed method only utilize the internal property to evaluate the visual fidelity of individual generations, while Realism Score and Perceptual Path Length require the additional resources.

### Diversity and Fidelity for Each Group

To perform comparison diversity and the visual fidelity in each group, we measure precision and recall [61] with the generations came from the truncation trick as the baseline<sup>5</sup>. In Figure 4.8, we can observe that the low CLA group shows the higher precision (visual fidelity) with slightly lower recall (diversity) comparing to the high CLA group. However, the low CLA group has much higher recall (diversity) comparing to the truncated generations for StyleGAN2. For PGGAN, the recall is higher on the truncated generation but the precision is higher on the low CLA group. From the observations, although we conjecture that there is trade-off between diversity and visual fidelity in evaluation criteria based on CLA score, we need further study to interpret this trade-off exactly.

<sup>4</sup>In general, for StyleGAN2, although  $w$  latent space is used to compute Perceptual Path Length, we compute the score in  $z$  latent space for the consistency for PGGAN case.

<sup>5</sup>The latent  $z$  space for PGGAN and the  $w$  space for StyleGAN2

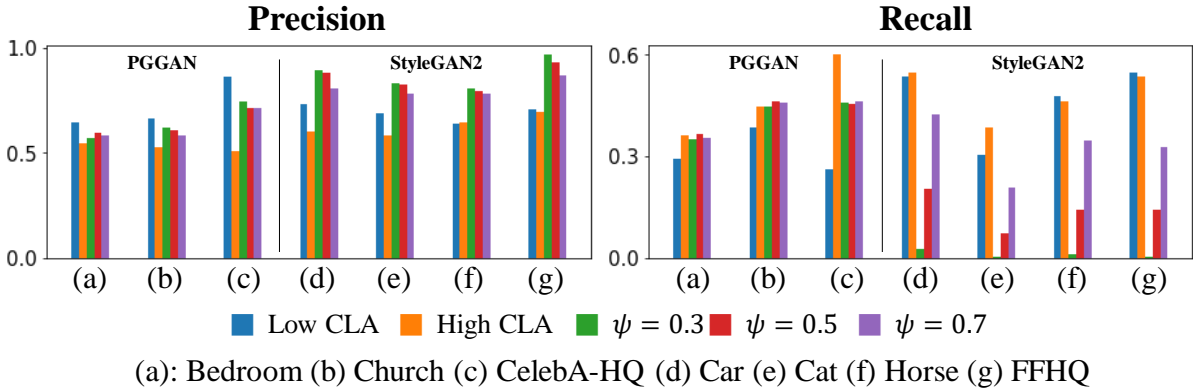


Figure 4.8: Precision and recall for each group in GANs trained with various dataset. Each color bar denotes each group, and each alphabet indicates each dataset. The  $\psi$  means the hyperparameter for truncation trick.

## 4.4 Discussion

### 4.4.1 Geometrical Interpretation of Local Activation

In this section, we discuss geometrical analysis to partially reveal the relationship between the visual fidelity of individual generations and the corresponding local activation in the restricted condition. The geometrical analysis is performed in terms of the competition between a generator and a discriminator. At first, we categorize the type of internal neurons in a perspective of the type of contribution for the output of the discriminator (positive or negative contribution). The discriminator  $D(x)$  in GANs can be similarly represented as the generator  $G(z)$  discussed in Section 4.2.1 where  $x$  is the target to generate such as a car image. The output of discriminator  $y$  for the given latent code  $z$  is expressed as  $y = D(G(z))$ . When we consider one latent code  $z_0$ , and the corresponding internal feature vector for the  $l-1$ -th internal layer  $\bar{h}_{l-1}$  in the generator  $G(z_0)$ , one can linearize the networks  $G$  and  $D$  with a partial linear activation function (e.g., LeakReLU or ReLU) which is commonly used in the modern GANs. Let  $\gamma \geq 0$  be the slope parameter for LeakyReLU (in ReLU case,  $\gamma = 0$ ), and  $w_{g_l, i}$  be the  $i$ -th column vector of weight parameter  $w_{g_l}$ . The corresponding linearized weight parameter  $\bar{w}_{g_l, i}$  is defined as,

$$\bar{w}_{g_l, i} = \begin{cases} w_{g_l, i} & \text{where } w_{g_l, i}^\top \bar{h}_{l-1} \geq 0 \\ \gamma \cdot w_{g_l, i} & \text{otherwise} \end{cases} \quad (4.4)$$

From this linearization, we can skip the non-linear activation function in a feed-forward pass for the given latent code  $z_0$ . After performing the linearization for entire layers, we can rephrase the linearized generator as  $\tilde{G}(z_0) = W_G^\top \bar{h}_l$  and the linearized discriminator as  $\tilde{D}(\tilde{G}(z_0)) = W_D^\top \tilde{G}(z_0)$  where  $W_G^\top = \bar{w}_{g_L}^\top \cdots \bar{w}_{g_1}^\top \in \mathbb{R}^{D_x \times D_l}$  and  $W_D^\top = \bar{w}_{d_L}^\top \cdots \bar{w}_{d_1}^\top \in \mathbb{R}^{1 \times D_x}$ . The output of the



discriminator  $\bar{y} = y$  is paraphrased as,

$$\bar{y} = \bar{D}(\bar{G}(z_0)) = W_D^\top W_G^\top \bar{h}_l = \sum_i W_D^\top W_{G,i}^\top \bar{h}_{l,i} \quad (4.5)$$

where  $W_{G,i}^\top$  is the  $i$ -th column vector of  $W_G^\top$  and  $\bar{h}_{l,i}$  is the  $i$ -th element of the vector  $\bar{h}_l$ . From the linearized output of the discriminator, we can identify that  $W_D$  is the normal vector of the decision boundary to evaluate the visual quality (real or fake) of the current generation  $W_G^\top \bar{h}_l = W_G^\top(\bar{z}_0)$ . And we also identify that the current generation can be represented as a linear combination of  $\{W_{G,i}^\top\}_i$  with the coefficients  $\{\bar{h}_{l,i}\}_i$ . The contribution of the  $i$ -th internal neuron in the  $l$ -th internal layer to the output of the discriminator  $\bar{y}$  will be  $W_D^\top W_{G,i}^\top \bar{h}_{l,i}$ . We determine that the  $i$ -th internal neuron in  $l$ -th internal layer has a negative/positive contribution if the activation value of the  $i$ -th internal neuron decreases/increases for the output of the discriminator, respectively. For instance, when  $W_D^\top W_{G,i}^\top \bar{h}_{l,i} < 0$ , the  $i$ -th internal neuron has a negative contribution to the output of the discriminator, as the activation value of  $\bar{h}_{l,i}$  reduce the magnitude of output of the discriminator.

In the vanilla GAN [4], we investigate how the type of contribution of the internal neurons for the output of the discriminator affects to the updates of weight parameters in the generator. Because the shape of the local activation pattern can be determined by the trained generative boundaries. Thus, we begin as describing how to perform the updates of weight parameters related to the type of the contribution of each internal neuron, and then interpret the results of updates. The loss function for the generator  $G$  and discriminator  $D$  is defined as,

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)}[\log f(D(x))] + E_{z \sim p_z(z)}[\log(1 - f(D(G(z))))] \quad (4.6)$$

where  $f(\cdot)$  is the sigmoid function. For  $w_{g_l}$ , the updated weight parameter  $w_{g_l}^+$  by the stochastic gradient descent of the given latent code  $z_0$  with linearized form is described as,

$$w_{g_l}^+ = w_{g_l} + \eta c_0 f'(\bar{y}) W_D^\top W_G^\top \bar{h}_{l-1} \quad (4.7)$$

where  $c_0 = (1 - f(D(G(z_0))))^{-1}$  and  $\eta > 0$  is learning rate. The  $i$ -th column vector  $w_{g_l,i}$  which induces activation of the  $i$ -th internal neuron in the  $l$ -th internal layer ( $\bar{h}_{l,i}$ ), is transformed by the direction  $\bar{h}_{l-1}$  with weight  $\delta_i = \eta c_0 f'(\bar{y}) W_D^\top W_{G,i}^\top \in \mathbb{R}$ . Figure 4.9 shows the geometrical illustrations of the update for four available cases. The available cases can be categorized by the type of contribution and the sign of activation value of  $\bar{h}_{l,i}$ , so we obtain a total of 4 update scenarios.

In Figure 4.9, we can identify that the perpendicular distance between the generative boundary (colored solid lines) and the magnitude of feature vector  $\bar{h}_{l-1}$  (colored dotted lines) decreases in the negative contribution cases when the learning rate is sufficiently small. Additionally, the magnitude of the activation value of  $\bar{h}_{l,i}$  also decreases in the negative contribution cases ( $\bar{h}_{l,i} \cdot \delta_i < 0$ ) as,

$$\bar{h}_{l-1}^\top w_{g_l,i}^+ = \bar{h}_{l-1}^\top (w_{g_l,i} + \delta_i \bar{h}_{l-1}) = \bar{h}_{l-1}^\top w_{g_l,i} + \delta_i \|\bar{h}_{l-1}\|^2. \quad (4.8)$$

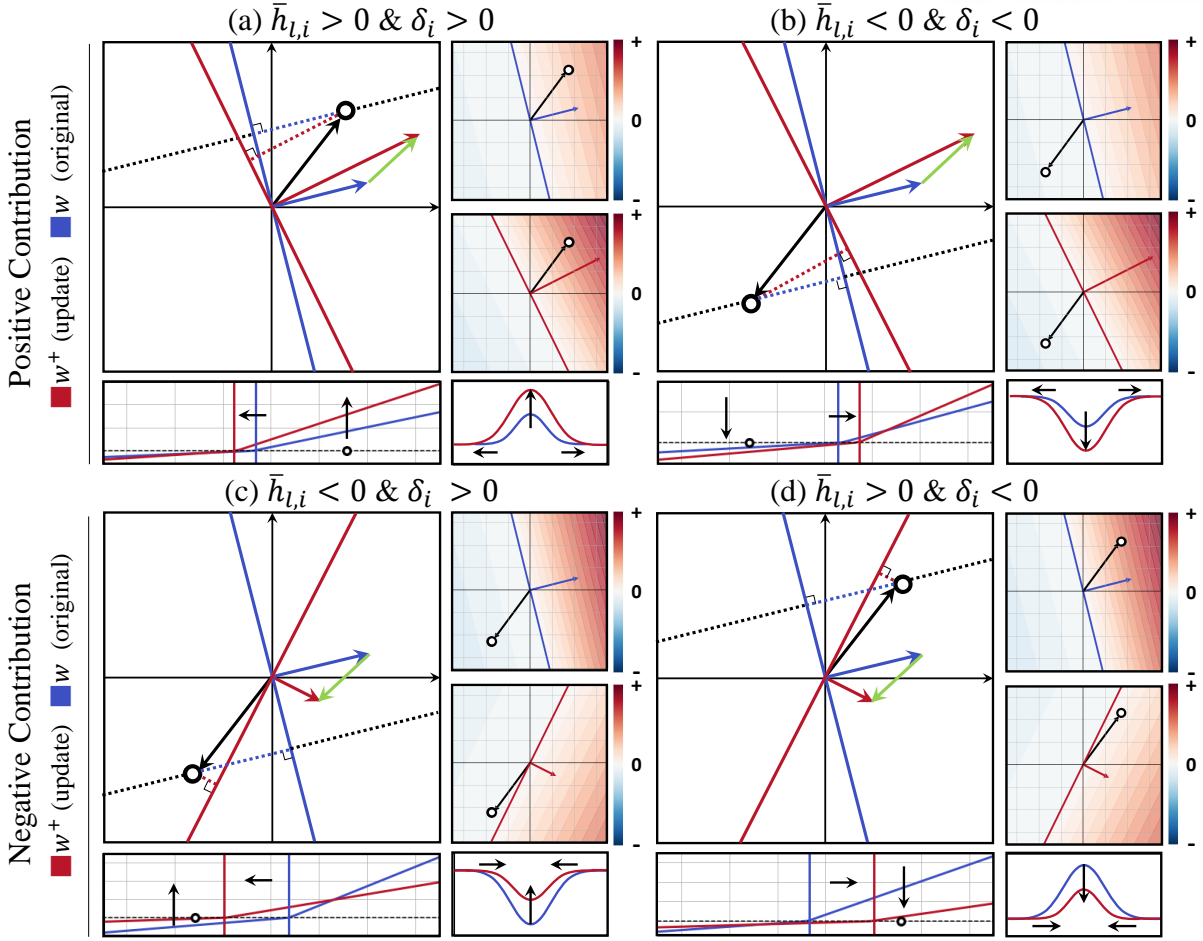


Figure 4.9: The geometrical illustrative examples of available weight parameter update cases ( $\bar{h}_{l-1} \rightarrow \bar{h}_{l,i}$ ). The black arrow and the black dot represent  $\bar{h}_{l-1}$ , the green arrow is  $\delta_i \bar{h}_{l-1}$  (update term), the blue arrow is  $\bar{w}_{g_{l,i}}$  (original parameter), and the red arrow is  $\bar{w}_{g_{l,i}}^+$  (updated parameter). The bottom-left plot denotes the activation values on the black dashed line before and after the update. The bottom-right plot shows the conceptual representation of the activation pattern change after the update ( $z_0 \rightarrow \bar{h}_{l,i}$ ).

In positive contribution cases (in first row of Figure 4.9), we can identify that the perpendicular distance to the generative boundary increase (red dotted line is longer than blue dotted line), and the magnitude of the activation value of  $\bar{h}_{l,i}$  increase (see the 2D representation in bottom-left plot).

From the investigation for update scenarios on the vanilla GAN, we can presume that when an internal neuron in the generator negatively contributes to the output of the discriminator, the generator tries to deactivate the internal neuron by (1) reducing the magnitude of the activation value, and (2) distance between generative boundary and feature vector from the previous internal layer, to deceive the discriminator. Because this penalization can be performed for the arbitrary latent code  $z$  in the latent space, if an internal neuron has the negative contribution consistently during the training, the corresponding activated region in the latent space will

shrink. And this shrinkage of the activation region can cause the locally activated region, which can be connected to high curvature of local activation (CLA). From the fact that the discriminator evaluate the visual quality of generations comparing the real images, we can expect that the negative contribution for the output of the discriminator is related to low visual fidelity of individual generations. As a result, if the locally activated region is not fully removed during the training, the corresponding internal neuron may induce artifact generations when highly activated.

Although we partially interpret the relationship between the local activation and the visual fidelity of individual generations in a perspective of the adversarial framework, the performed geometrical analysis is mainly focus on the point-wise analysis. This means that it is non-trivial to expand the local explanations for the global explanation directly, and the theoretical reasons for the observed relationship between local activation and visual fidelity of individual generations are still remained as an open question.

#### 4.4.2 Comparisons between Perceptual Path Length and CLA

In this section, we compare the differences between Perceptual Path Length and CLA. Perceptual Path Length is known as the quantification of smoothness of the latent space of the trained generator, and in terms of smoothness, curvature of local activation (CLA) is also related. The first key difference is that Perceptual Path Length needs an external network (e.g., pre-trained VGG-16 with ImageNet) to quantify the smoothness in the latent space. The smoothness is computed as the distance in the feature manifold obtained by this external network for small perturbations of the given latent code. The dependency on the external network can not only requires additional resources but also raises a limitation which the reliability of scoring depends on the capability of the external network. In other words, if the class of generations is not well-aligned to the pre-trained external network, it becomes difficult to guarantee the reliability of the quantified perceptual distance in the feature manifold of the external network. For instance, our generator is trained a special dataset which is not related to ImageNet, it is hard to trust the Perceptual Path Length for individual generations. In seconds, Perceptual Path Length computes the first order derivative, while our method approximate the second order derivative for the smoothness of latent space. Although the path length regularization is applied in StyleGAN2 to regularize the first order derivative, we can still observe high curvature of local activation.

## 4.5 Related Work

**Deep Generative Neural Networks** In general, deep neural networks based generative models mainly map the input distribution to the given a target data distribution. Representative architectures include variational autoencoder (VAE) [63], neural language models [64–66] and generative adversarial networks (GANs). Especially, the adversarial training framework with

deep neural networks (i.e., GANs) which utilize the relationship between a generator and a discriminator [4] has shown remarkable performance in image generation [5, 6, 32, 62].

**Analysis for Interior Mechanism of GANs** GAN Dissection [3] devises a method to analyze the generative concepts handled by each internal unit of generator in GANs. The authors perform zero-ablation for artifact generating internal units related to defective visual pattern to improve the visual fidelity of individual generations. Another work [13] trains a linear classifier (e.g., support vector machine) depending on annotation of generations (artifact generations or not), and removes artifact generations by manipulating the latent code toward the hyper-plane induced by the trained classifier. A explorative sampling method considering the trained generative boundaries and region, was introduced to interpret shared semantic attributes in a perspective of the generator [21]. Classifier-based artifact generating internal unit identification was discussed in Chapter 3. We can improve the visual fidelity of individual generations with sequential control of the generation information for the detected internal units from the trained classifier. Investigations for latent space of the trained generator in GANs were also performed to manipulate the attributes of the generation [14, 67]. The authors empirically identify that the rich semantic information of dataset is embedded in the latent space of GANs from the manipulation of latent codes. Our work mainly focuses on the internal generative mechanism, and the relationship between artifact generations and the properties of internal neurons represented in the latent space of GANs.

**Metric for Generative Model** Various metrics have been introduced to assess the performance of generative models, and properties of each metric are well-summarized in [68]. Although Fréchet Inception Distance (FID) [59] and Inception Score (IS) [58] are commonly used to evaluate the generative model from the robustness to image distortion, they sometimes assign high scores (indicate high visual fidelity) for generations with low visual fidelity. Precision and Recall (P&R) is a surrogate metric to quantify mode dropping and inventing based on training data [60, 61] in the generative models. They mainly focus sample-wise comparison in the feature manifold obtained from the pre-trained network such as VGG-16, while the FID and IS are related to distributional comparison. The authors also devised Realism Score (RS) to evaluate the visual fidelity of individual generations by computing similarity of feature embeddings with training data. Perceptual path length (PPL) is another metric that quantifies the smoothness of the latent space with a hypothesis that the region in the latent space for artifact generations has a small volume compared to normal generations [32]. It measure the distance in the feature space of pre-trained classifier between small perturbations of the given latent code in the latent space. Although our work is mainly related to the metric for GANs, the key difference is that the proposed method can be performed in unsupervised manner, while the previous metrics require additional supervision such as pre-trained network or a large amount of training dataset.

## 4.6 Final Remark

In this chapter, we introduce the novel concept, *local activation*, on the internal neurons of generator in GANs to assess the visual fidelity of individual generations in unsupervised manner. We partially discuss a geometrical analysis to reveal the relationship between the proposed concept, curvature of local activation (CLA), and visual fidelity of individual generations under the restricted condition. We provide various experimental studies to empirically verify the observed relationship on artifact detection and correction settings. From the results of experiments, we confirm that the proposed CLA score shows satisfactory performance without external resources such as pre-trained network or a large amount of training data as the supervision. As the introduced internal properties is related to the basic component (neuron) of deep neural networks, we believe that the approach based on the CLA can be extend to a wide range of deep learning models.

## Chapter 5

# Conclusion and Future Work

This chapter concludes this thesis with summary and discuss potential future work.

### 5.1 Summary of Contribution

One of the main contribution in this thesis is to propose an efficient sampling strategy which can guarantee the neural representation sharing in a perspective of the network. Because of exploration considering the internal generative boundaries in the latent space, it supports obtaining the reliable examples to understand internal generative mechanism of the network.

The second contributions of this thesis is to devise the automatic detection and correction of internal artifact generating units to improve the reliability of the generator. Along with defective unit detection and correction procedures, we identify that the well-trained GANs still have internal units which cause the low visual fidelity in internal layers, and the artifacts can be repaired as ablating these detected units.

The third contributions of this thesis is to provide unsupervised way to understand internal artifact generating units in GANs. The proposed metric based on the local activation pattern can be applied to quantify the defectiveness of the individual generations without additional supervisions. This thesis also presents the partial geometrical analysis to discuss the relation between the local activation and the low visual fidelity in a perspective of the adversarial competition.

### 5.2 Future Work

As presented in this thesis, the full-identification for internal generative mechanism of deep generative neural networks is usually hard. Although the approach in this thesis provides remarkable examples to understand the internal generative mechanism indirectly, it is still non-trivial to analyze directly. One promising direction is to expand the geometrical analysis (i.e., generative boundary and region comprised of the internal neurons) related to the dynamics of training for the generator. This idea is based on the partial discussion in Section 4.4.1 of Chapter 4 and the geometrical analysis for deep classifier [26, 26, 40, 69]. Because preliminary work for deep



classifier has achieved to interpret the decision mechanism effectively, this approach will answer about the internal generative mechanism more concretely.

Score-based generative models [70–72] have shown high performance recently, and the model beats GANs which this thesis mainly focus as the target deep generative neural networks in experiments. Despite these two models use similar structure to build the network for generation, there are many interesting open questions, as training scheme for each model is largely different: (1) how the score-based model controls the neural representation sharing (in this case, the neural representation sharing will be related to the score vector), and generates defective score vectors which can cause the artifacts as the final generation; (2) can the proposed example-based methods be applicable to understand the internal generative mechanism of score-based model? These are all challenging problems required careful treatments.

### 5.3 Conclusion

In conclusion, I hope that this thesis can provide a fresh perspective to understand the internal generative mechanism of deep generative neural networks, in particular, for artifact generations in GANs. The proposed methods can support to improve the reliability of the deep generator for real-world applications. The theoretical understandings for internal causes related to artifacts in this thesis are still preliminary, but potentially helpful for interpreting the internal generative mechanism of not only GANs but also other deep generative neural networks such as the score-based generative model.

## Chapter 6

# Appendix

This chapter includes the supplemental materials for each chapter.

### 6.1 Details of Human Evaluations

#### 6.1.1 Evaluation Results

To perform human evaluation experiment, at first, we prepare 500 artifact generations, and corresponding corrected generations. To quantify the correction performance for the artifact generations, we design the criteria for each dataset to guide the decision of raters. The human evaluation process is twofold: (1) Raters perform re-labeling for the corrected generations and (2) Determine the improvement for generations compared to original generations based on the pre-defined criteria. The summary of human evaluation is described in Table 6.1.

Table 6.1: Summary for results of human evaluation on corrected generations in PGGANs trained with various dataset. The number in the parentheses denotes the standard deviations over raters.

Dataset	Corrected (%)	Improved (%)
CelebA-HQ	53.00 (4.20)	96.00 (2.00)
LSUN-church	54.50 (0.90)	86.10 (6.30)
LSUN-bedroom	46.80 (8.60)	95.50 (1.10)

Figure 6.1 denotes the stacked bar charts for the results of human evaluation in PGGANs trained with various dataset *Strongly normal* denotes to the portion of the unanimous agreement among 500 generations that the corrected generations is normal class, and *Strongly artifact* refers to the portion of the unanimous agreement that corrected generations is still an artifact class. If it is not unanimous agreement, the generations falls into *Neither normal nor artifact* portion.

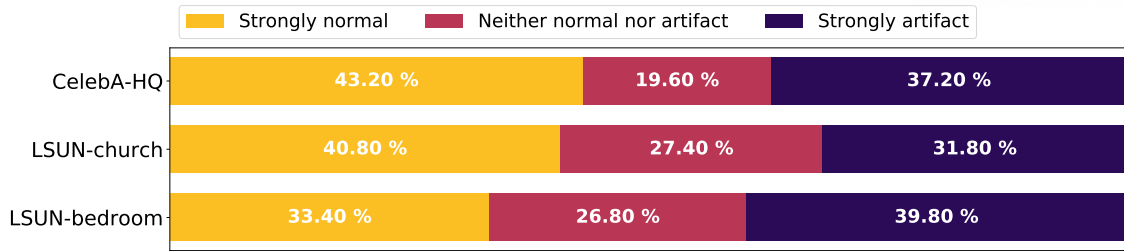


Figure 6.1: Detailed results for human evaluation of the re-labeling on the corrected generations in PGGAN trained with CelebA-HQ, LSUN-Church, and LSUN-Bedroom datasets

### 6.1.2 Evaluation Criteria

This section presents the criteria to determine the given generation is artifact or not for human evaluations to guide raters. At first, we define the criteria to categorize individual generation manually. Figure 6.2 shows the examples of artifact generations according to criteria for each dataset. We define different criteria for each dataset to cover the common types of artifact generation for each network. The detailed descriptions for the types of artifact generations and the criteria are described in Table 6.2.

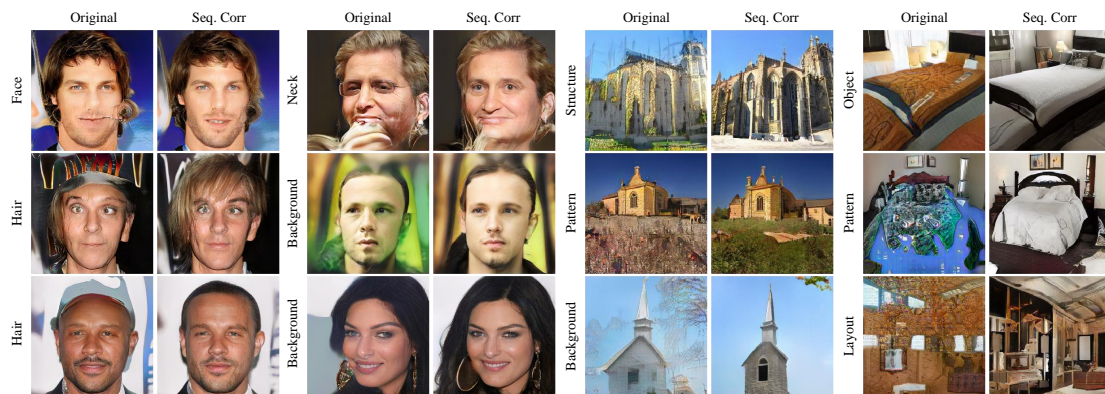


Figure 6.2: Illustrative examples for artifact generations according to the descriptions in the criteria for each dataset. The one generation can be included in multiple criteria. For instance, left-bottom generation satisfies the hair and background type of artifact at the same time.

Table 6.2: Types of artifact generations and its description

Dataset	Type	Description
CelebA-HQ	Face	<ul style="list-style-type: none"> <li>• There exist defective segments (i.e., eyes, nose, mouth) on the face.</li> <li>• The color-tone and texture of the face is not realistic.</li> </ul>
	Hair	<ul style="list-style-type: none"> <li>• The hair which is fused in the background.</li> <li>• The head is replaced with patterns.</li> <li>• There exist empty parts in hair.</li> </ul>
	Neck	<ul style="list-style-type: none"> <li>• The neck is missing.</li> <li>• Neck and shoulder line is not clear.</li> </ul>
	Background	<ul style="list-style-type: none"> <li>• There exist clear patterns in the background.</li> </ul>
LSUN-Church	Structural Defects	<ul style="list-style-type: none"> <li>• Each part of church, i.e., roofs, windows, walls, doors and outlook are not separated each other or from the background.</li> <li>• Church is transparent or overlapped.</li> <li>• The structure of a church is not clear or impossible.</li> </ul>
	Pattern	<ul style="list-style-type: none"> <li>• There exists a clear pattern (i.e., checkerboard pattern) which is unnatural.</li> <li>• The color of an object is unnatural.</li> </ul>
	Background	<ul style="list-style-type: none"> <li>• There exists standing out, unrecognizable object.</li> <li>• There are transparent objects in the sky.</li> </ul>
LSUN-Bedroom	Object	<ul style="list-style-type: none"> <li>• There is no bed in bedroom.</li> <li>• The structure or edge of an object is physically not feasible.</li> </ul>
	Pattern	<ul style="list-style-type: none"> <li>• The texture of the images is like a painting.</li> <li>• There is a transparent patterns.</li> <li>• There is a clear pattern crossing the boundary of an object.</li> </ul>
	Layout	<ul style="list-style-type: none"> <li>• The layout of the bedroom is not clear and realistic (i.e., The walls are not connected).</li> <li>• No objects or room at all.</li> </ul>

## 6.2 Artifact Correction Results

In this section, we provide the artifact correction results for each correction method. We set FID-score based internal unit detection and single-layer zero-ablation as the baseline. We refer DS to represent the defective score based internal unit detection and single-layer zero-ablation. These two correction methods are performed in layer 6. The sequential correction method is performed from layer 1 to 6 (The layer 0 denotes the first dense layer with reshape.). Figure 6.3 shows the artifact correction result for random generations. Although we can confirm that all of methods preserve normal generation information, the sequential correction method shows effective repair for the defective areas. For instance, the stain on the forehead is removed in the sequential correction method (see top-right in Figure 6.3.). We note that entire correction methods are performed in the global manner to improve the performance of the generator not the correct per sample (the given generation). Figure 6.4 - 6.9 shows the correction results for each class (artifact or normal) in PGGAN trained with various dataset.

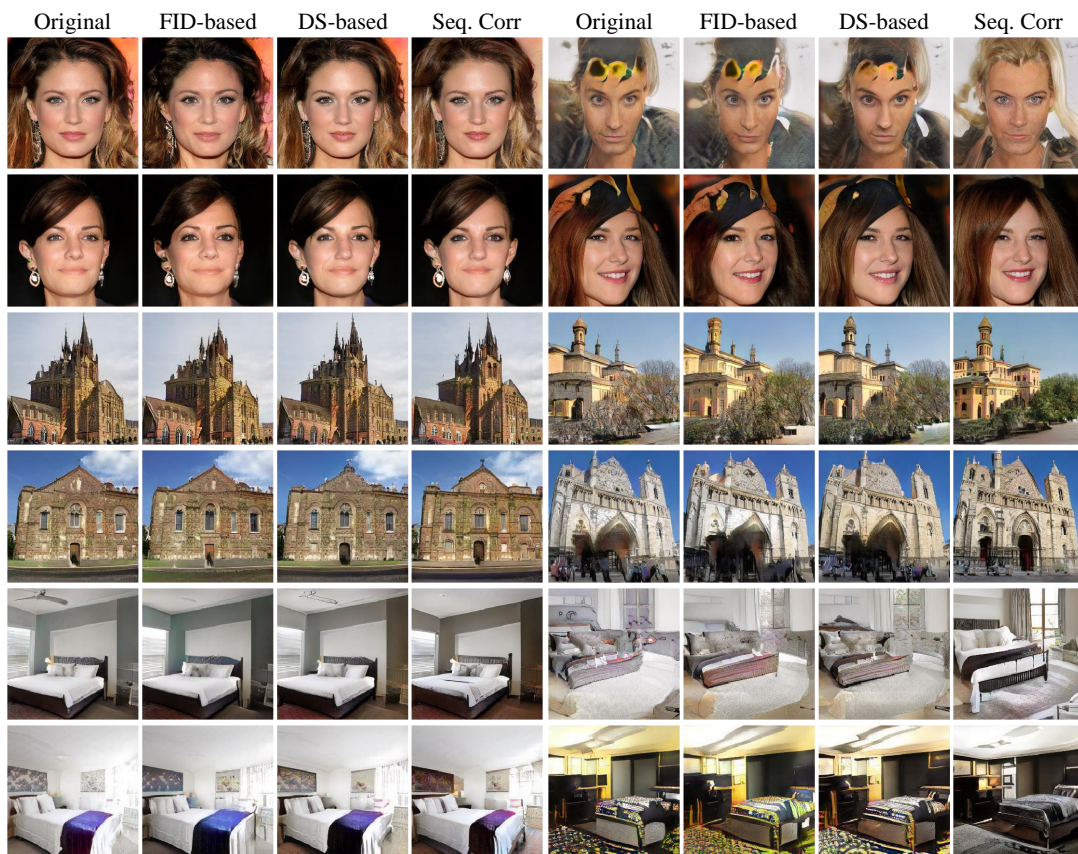


Figure 6.3: An illustrative examples for correction result for each correction method in PGGAN trained with various dataset. Although the entire correction method can preserve the plausible areas, the proposed sequential correction method can repair the defective areas effectively. See the improved quality of church in 4th row.



6.2.1 Correction Results in PGGAN trained with CelebA-HQ: Artifact

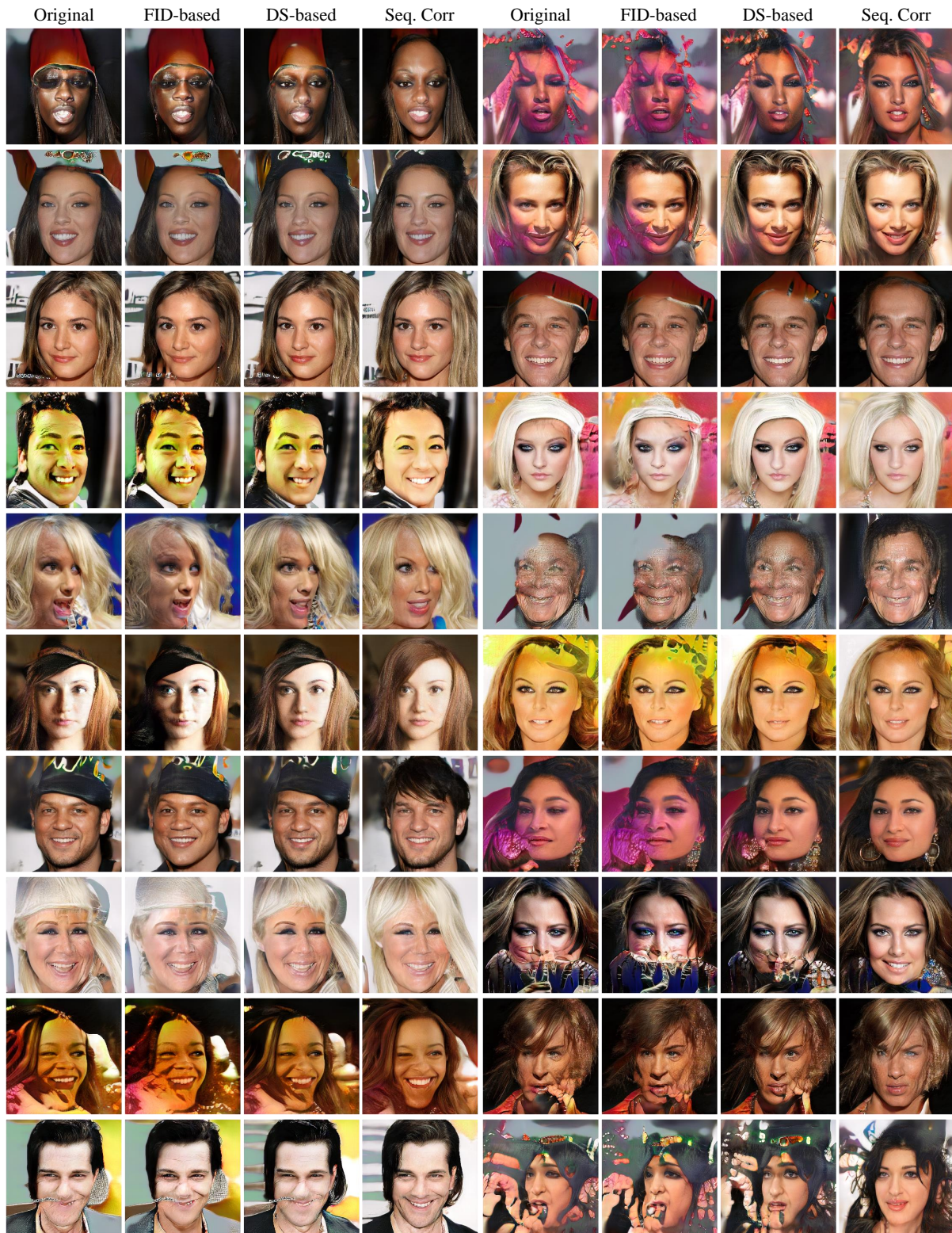


Figure 6.4: Correction results for the artifact generations in PGGAN trained with CelebA-HQ.



6.2.2 Correction Results in PGGAN with CelebA-HQ: Normal

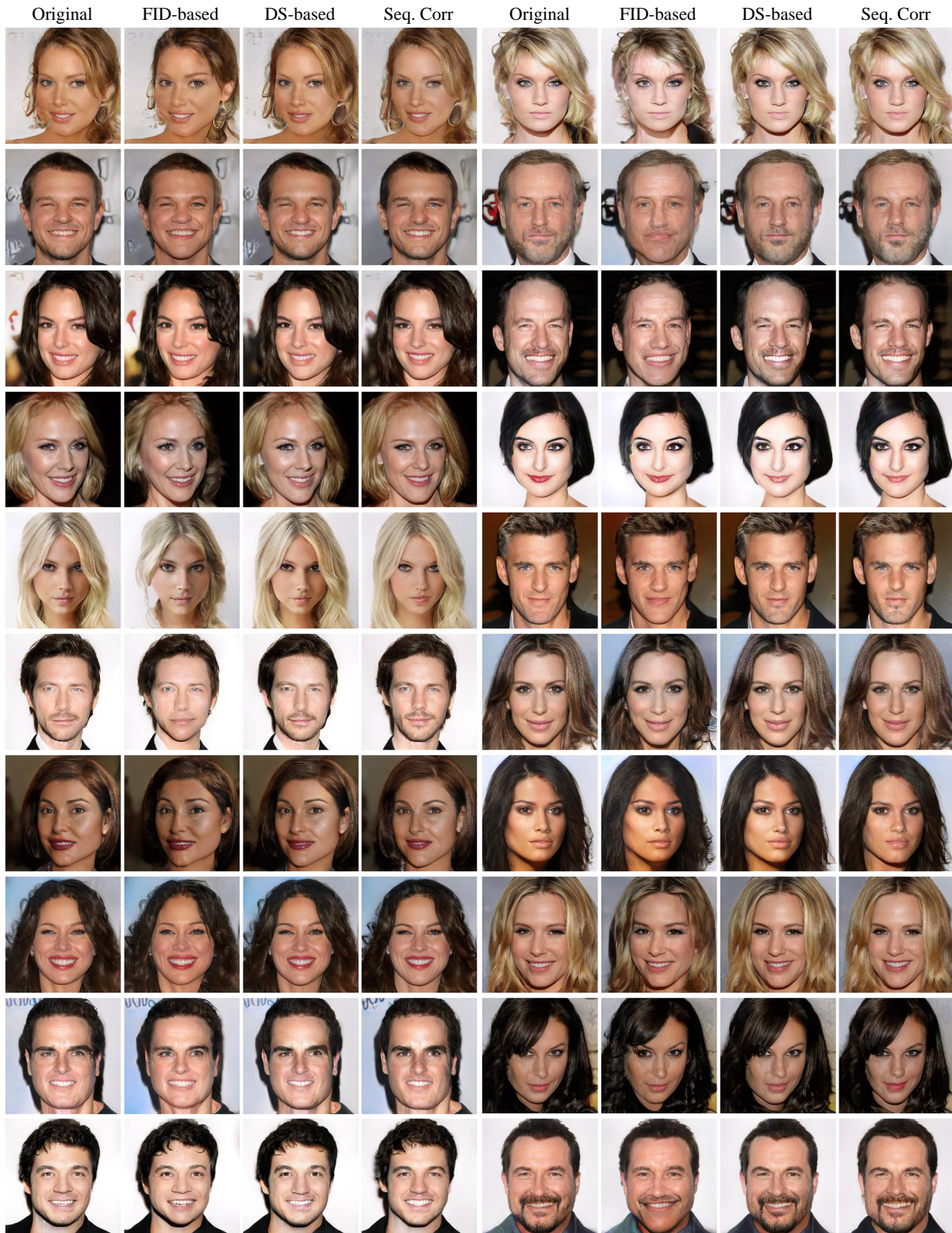


Figure 6.5: Correction results for the normal generations in PGGAN trained with CelebA-HQ.



6.2.3 Correction Results in PGGAN trained with LSUN-Church: Artifact

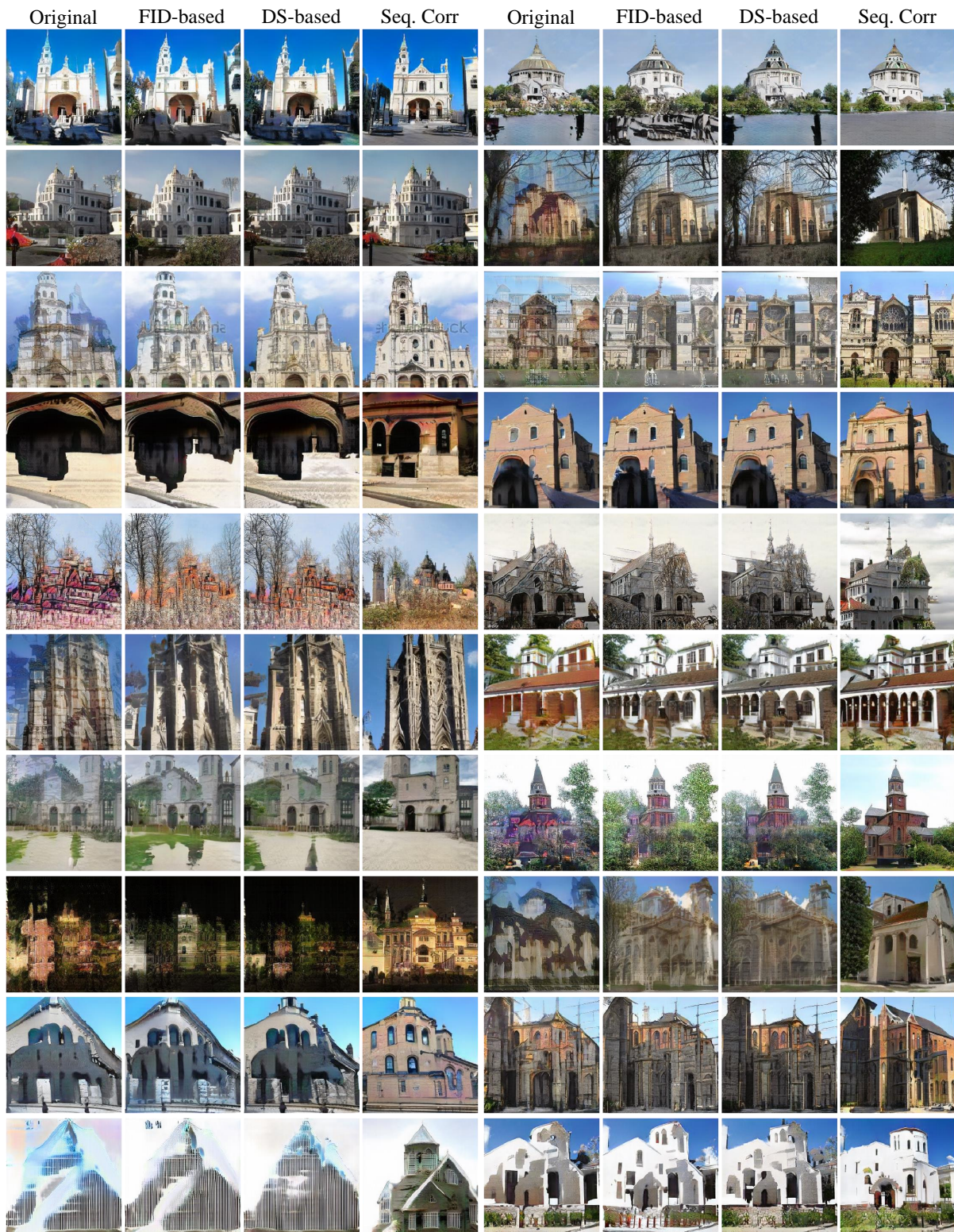


Figure 6.6: Correction results for the artifact generations in PGGAN trained with LSUN-Church.



6.2.4 Correction Results in PGGAN trained with LSUN-Church: Normal

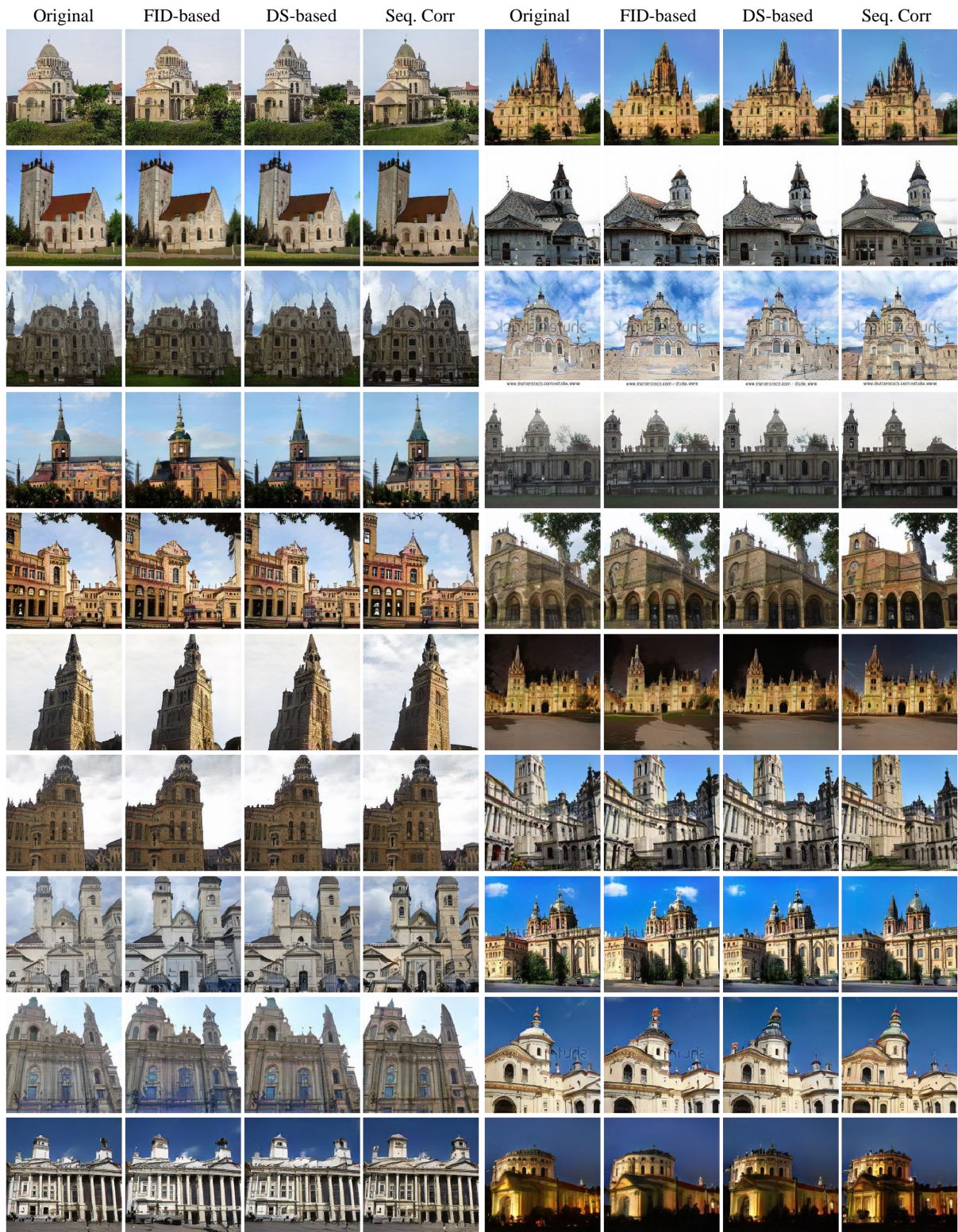


Figure 6.7: Correction results for the normal generations in PGGAN trained with LSUN-Church.



6.2.5 Correction Results in PGGAN trained with LSUN-Bedroom: Artifact

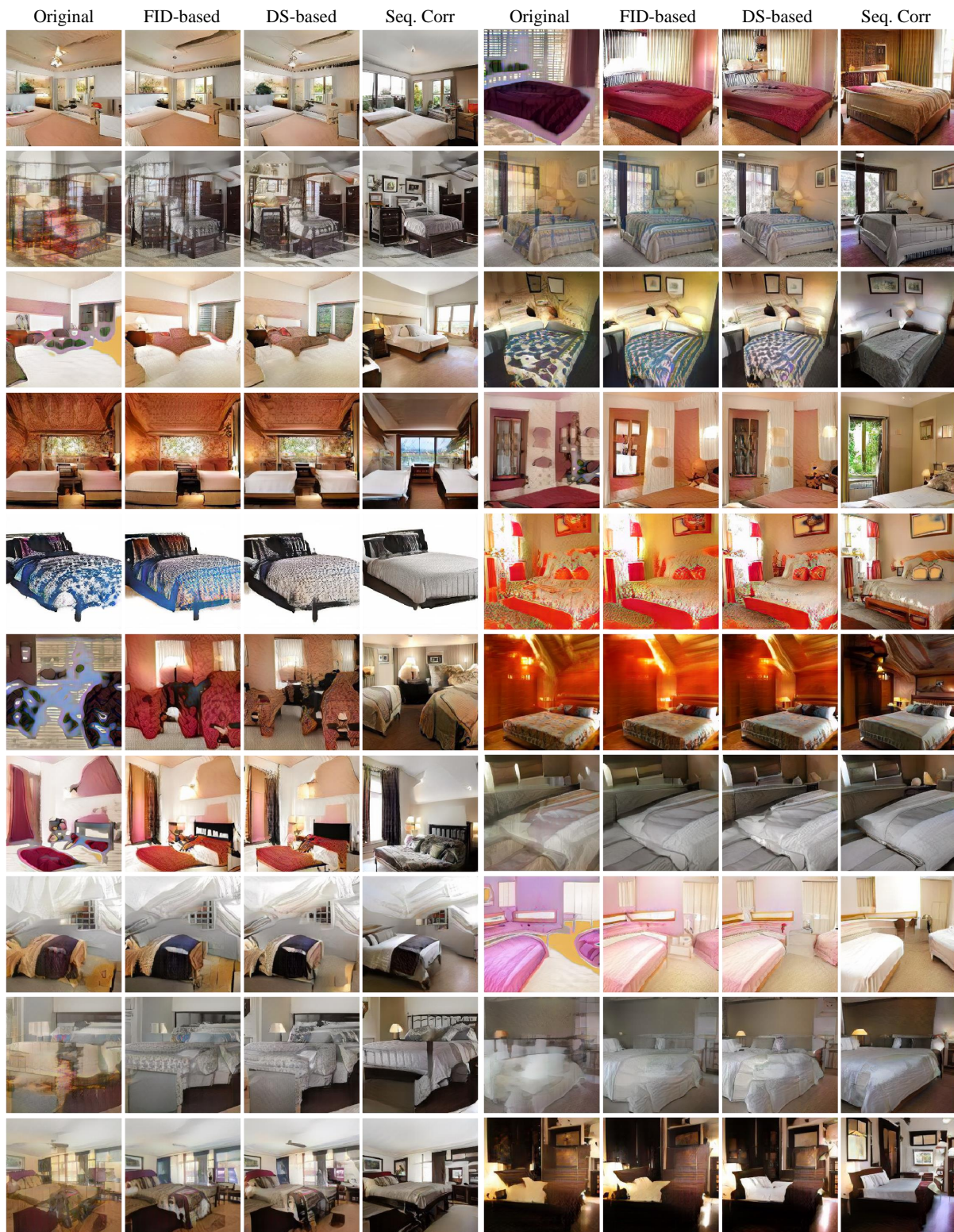


Figure 6.8: Correction results for the artifact generations in PGGAN trained with LSUN-Bedroom.



### 6.2.6 Correction Results in PGGAN trained with LSUN-Bedroom: Normal

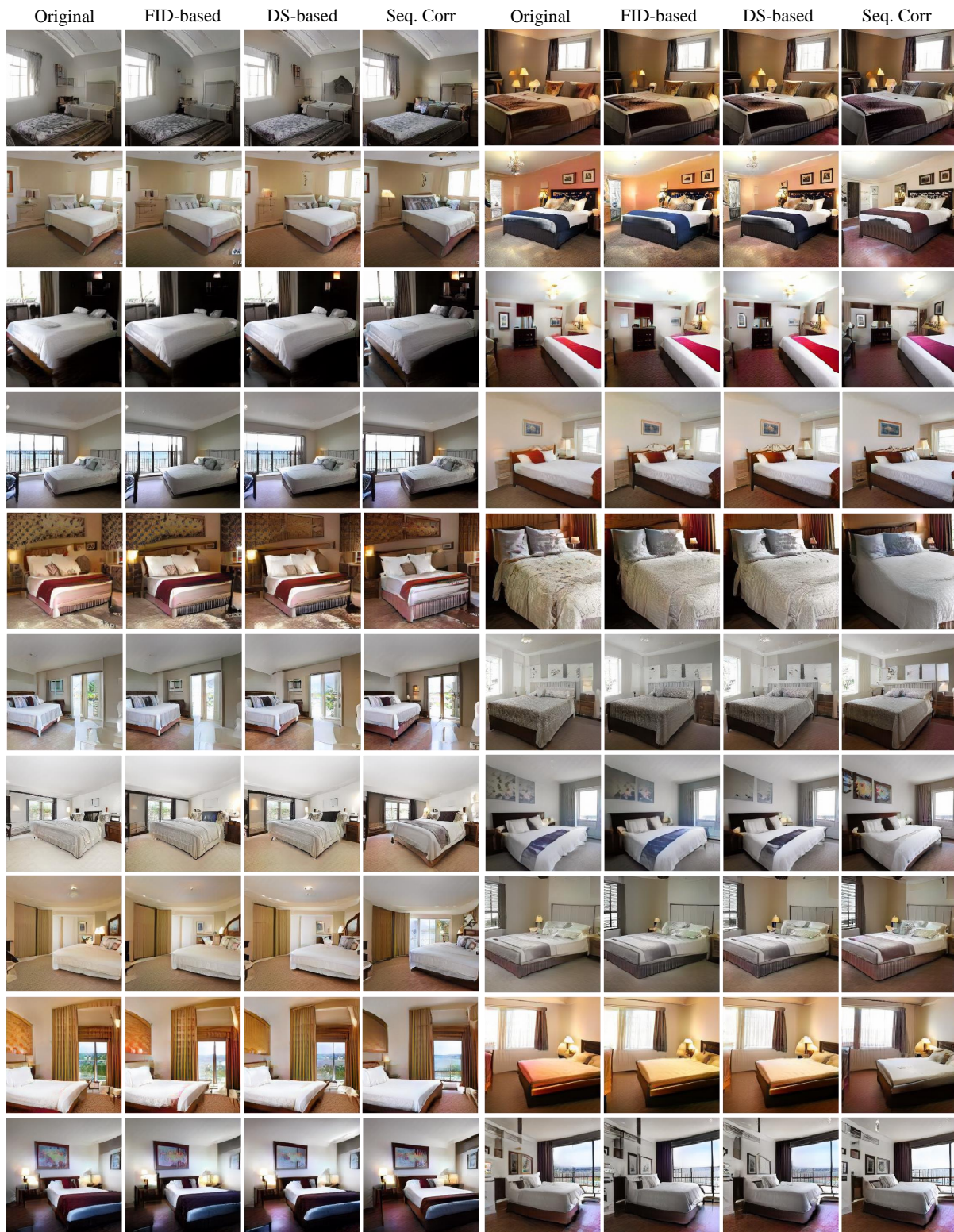


Figure 6.9: Correction results for the normal generations in PGGAN trained with LSUN-bedroom.



### 6.3 Sequential Correction in StyleGAN2 and U-net GAN

Because the structural difference between PGGAN and StyleGAN2 (e.g. stochastic variation induced at each convolutional layer), the internal unit detection and correction in the global framework is not trivial. However, we can obtain the artifact mask for the given generation from Grad-CAM method, the proposed internal unit detection and correction can be applied with slight modification (global sense  $\rightarrow$  sample-based). If a generation is predicted as artifact class from the trained classifier, we acquire an artifact mask for defective areas. Similar to our global unit identification for PGGAN, we can assign relative defective score using sample-based artifact mask. Figure 6.10 shows the result of sample-based sequential correction method for generations in StyleGAN2 trained with FFHQ.

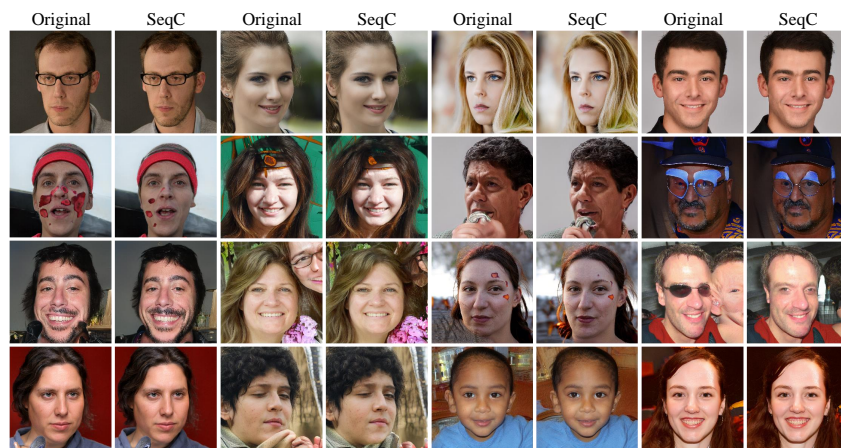


Figure 6.10: Correction results for the generations in StyleGAN2 trained with FFHQ.

We also adopt U-net GAN which the generator module is same as BigGAN with some modifications. To investigate the correction results, we follow the same procedure as StyleGAN2 for sample-based sequential correction method.

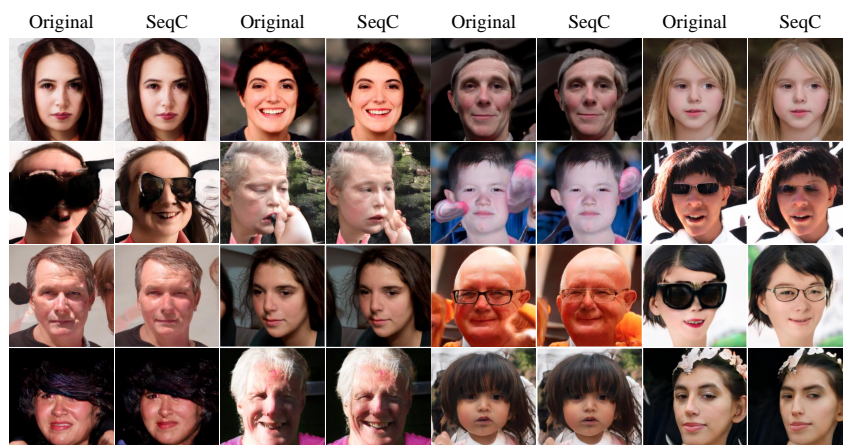


Figure 6.11: Correction results for the generations in U-net GAN trained with FFHQ.

## 6.4 Representative Generation and Highly Activated Generations for Internal Units

This section provides the representative image and highly activated generations for detected internal units in layer 6 of generator in PGGAN trained with various dataset to investigate generation concepts handled in the specific internal unit. Figure 6.13 - 6.15 shows the visualization results for top-15 units in each scoring method (left: FID-score and right: defective score (DS)). Figure 6.12 shows the procedure for obtaining the highly activated generations and synthesizing the representative image for the selected internal unit.

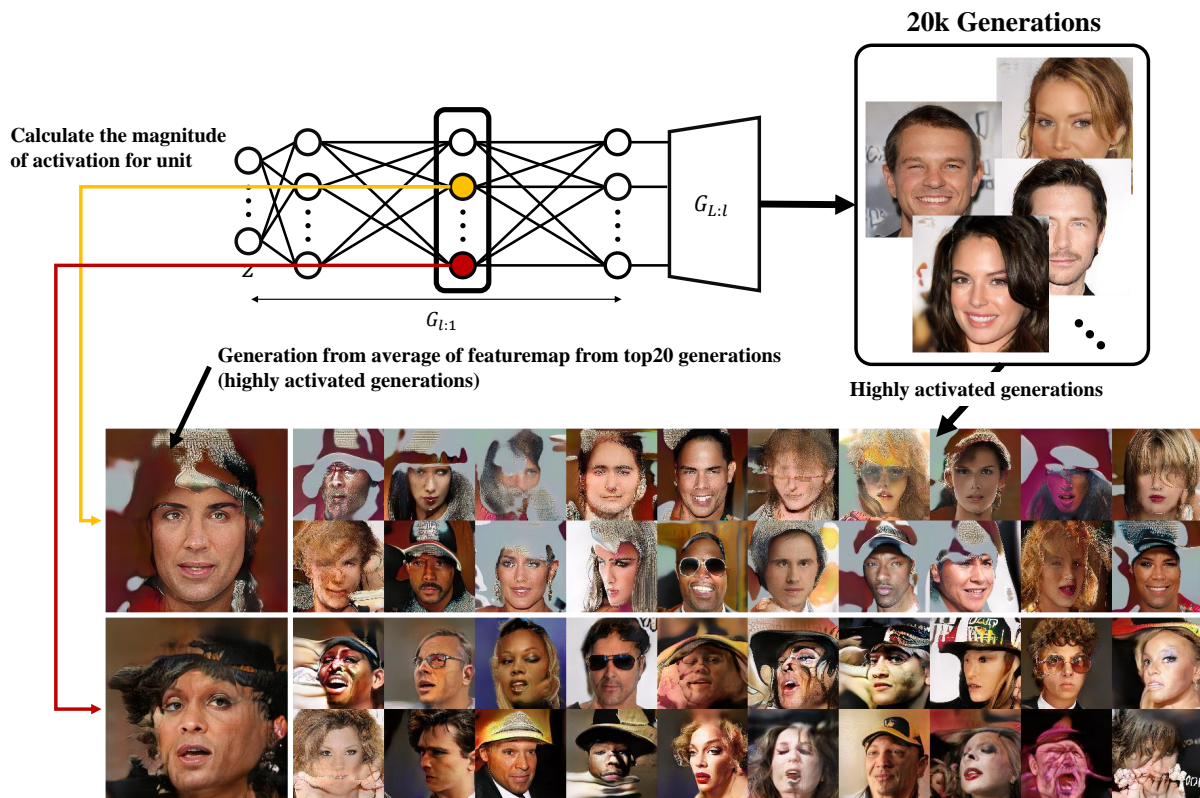


Figure 6.12: An illustrative example for how to obtain highly activated generations and synthesize the representative image.





Figure 6.13: Representative image and highly activated generations for each internal unit in layer 6 of PGGAN trained with CelebA-HQ. (Left) FID-score based unit generations. (Right) defective score based unit generations.



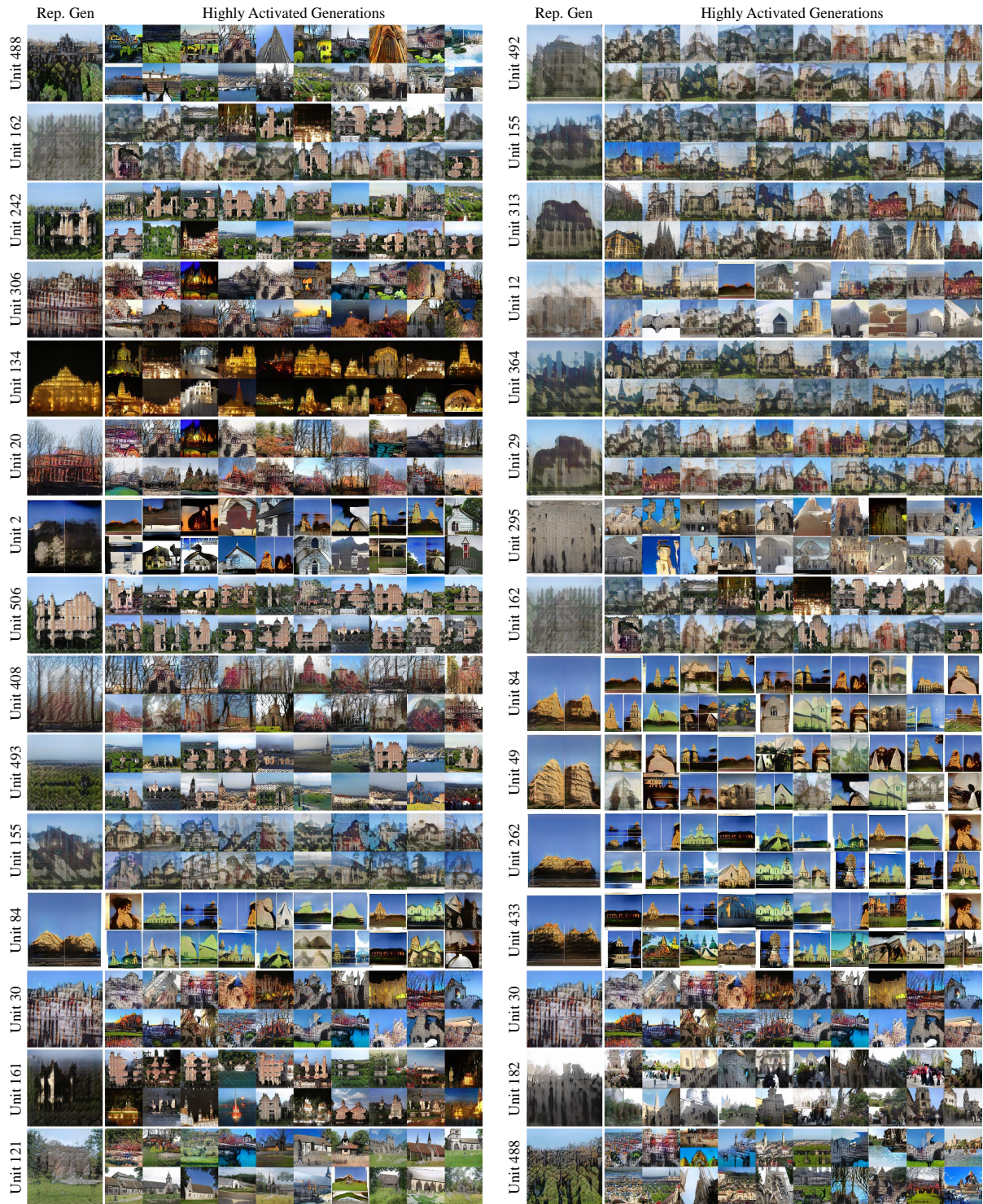


Figure 6.14: Representative image and highly activated generations for each internal unit in layer 6 of PGGAN trained with LSUN-Church. (Left) FID-score based unit detection. (Right) defective score based unit detection.



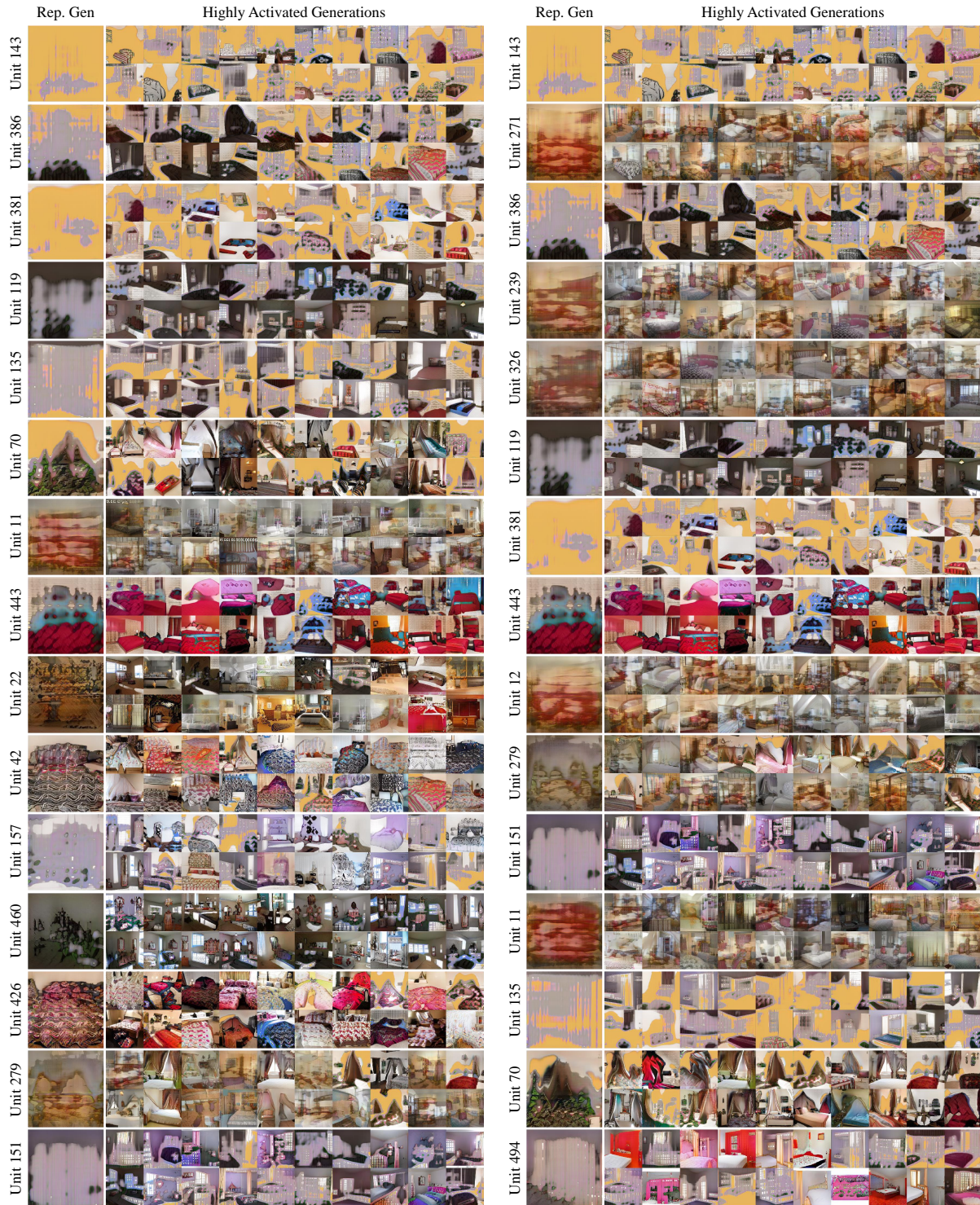


Figure 6.15: Representative image and highly activated generations for each internal unit in layer 6 of PGGAN trained with LSUN-Bedroom. (Left) FID-score based unit detection. (Right) defective score based unit detection.

## 6.5 Exploration of Hyperparameter for CLA

To determine the hyperparameter (search bound  $R$ , division  $n$ , and target internal layer  $l$ ) for computation of CLA, we explore with simple grid search. At first, we choose 200 generations for each group (high and low CLA groups) in randomly selected 1k generations for PGGAN trained with CelebA-HQ, and measure the average of Realism Score and Perceptual Path Length of each group for various hyperparameter settings. Figure 6.16 shows the computed scores for each hyperparameter setting. In Figure 6.16, the red/blue color denotes high/low CLA group respectively.

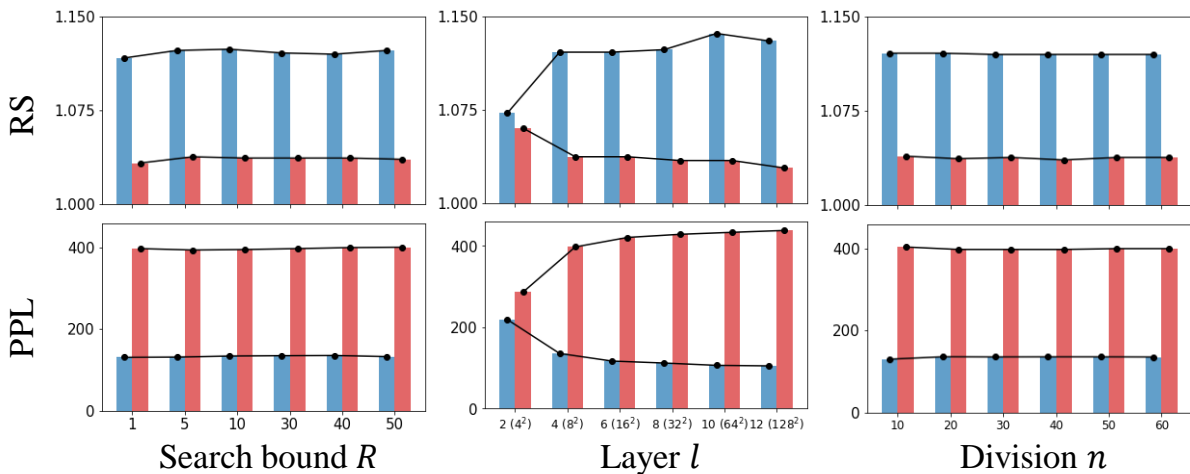


Figure 6.16: Realism Score and Perceptual Path Length for the various hyperparameter settings. (First column) division  $n = 20$  and target internal layer  $l = 4$  are fixed. (Second column) search bound  $R = 30$  and division  $n = 20$  are fixed. (Third column) search bound  $R = 30$  and target internal layer  $l = 4$  are fixed.

From first and third columns of Figure 6.16, we can identify that it is non-trivial to observe the large difference of artifact detection performance according to the various magnitude of  $R$ . For applicable to GANs trained with various dataset, we prefer the larger value of search bound  $R$  to guarantee the existence of the left/right change points, which make the computation of CLA more accurate. However, the larger value of search bound  $R$  may require the larger value of division  $n$ , which requires a expensive computation cost. To consider this trade-off, we determine search bound  $R = 30$  and division  $n = 20$  for artifact detection game. In the second column, we can observe that the artifact detection performance increases depending on the depth of layer, although the increase is small enough after layer 4. The number of inspection neurons exponentially increases when the depth of layer increase, which denotes that the computation time exponentially increases. As a result, we determine layer 4 as the target internal layer for the artifact detection experiment as considering both performances of detection and computation time.



## 6.6 Computational Complexity for Computation of CLA

We can roughly approximate computational complexity as,

$$O(|D_z| * n) + O(|D_l| * |D_z| * n) \quad (6.1)$$

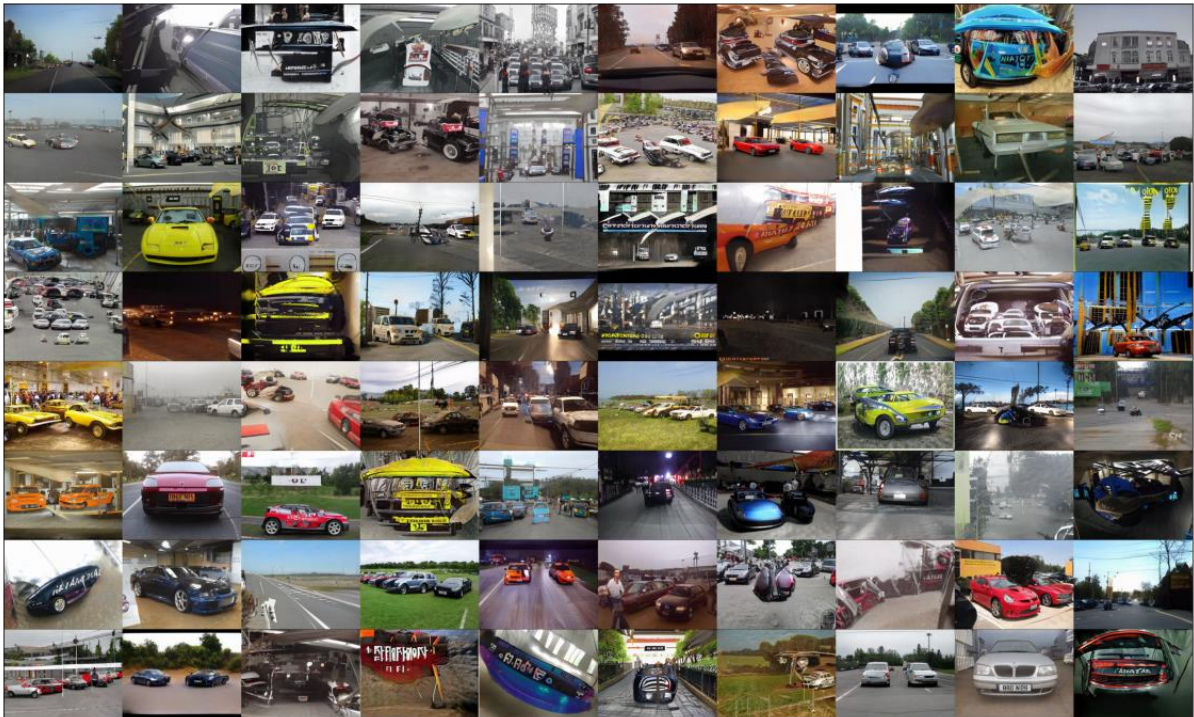
where  $D_z$  is the dimension of latent space,  $n$  is the value of division, and  $D_l$  is the dimension of feature vector in the target internal layer  $l$ . The first part in Equation 6.1 denotes the computational complexity for the perturbations of each latent axis with division constant  $n$ , and it is related to the forward pass in the generator. The second part in Equation 6.1 denotes the computational complexity for the computation of CLA in  $l$ -th internal layer. To measure the computation time for CLA score at the specific internal layer, we utilize 1 GPU (RTX 2080 Ti) with 100 iterations. In PGGAN trained with CelebA-HQ, the average computation time of CLA score for one latent code is about  $5.6 \pm 0.31$  sec at the target internal layer (512x8x8). In StyleGAN2 trained with FFHQ, the average computation time of CLA for the one latent code is about  $26.4 \pm 0.41$  sec at the target layer (512x8x8).

Although the computation time of CLA score seems to high for realtime application, we think that the computation time can reduce. This is because the calculation of CLA on the latent axes are independent of each other. The calculation can be implemented by the parallel computing (multi GPUs) for each latent axis to boost the speed. For instance, if we utilize 8 GPUs for calculation of CLA from divided latent axes, the total calculation time decreases by 1/8. We believe that the computation can be additionally improved by optimizing the code.

## 6.7 Artifact Detection in StyleGAN2 trained with LSUN-Car



(a) Bottom 25-104 generations in low CLA group.



(b) Top 25-104 generations in High CLA group.

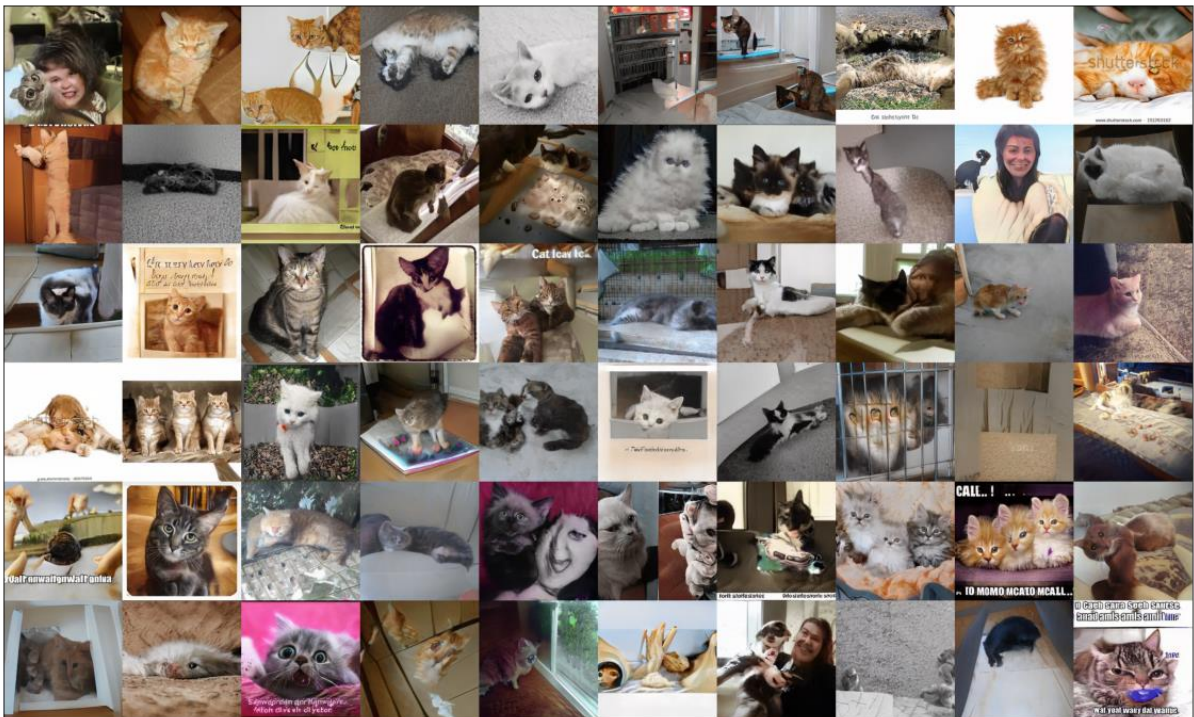
Figure 6.17: Generations depending on the magnitude of CLA in StyleGAN2 trained with LSUN-Car. We identify low CLA group has higher visual fidelity compared to high CLA group.



## 6.8 Artifact Detection in StyleGAN2 trained with LSUN-Cat



(a) Bottom 25-84 generations in low CLA group.



(b) Top 25-84 generations in High CLA group.

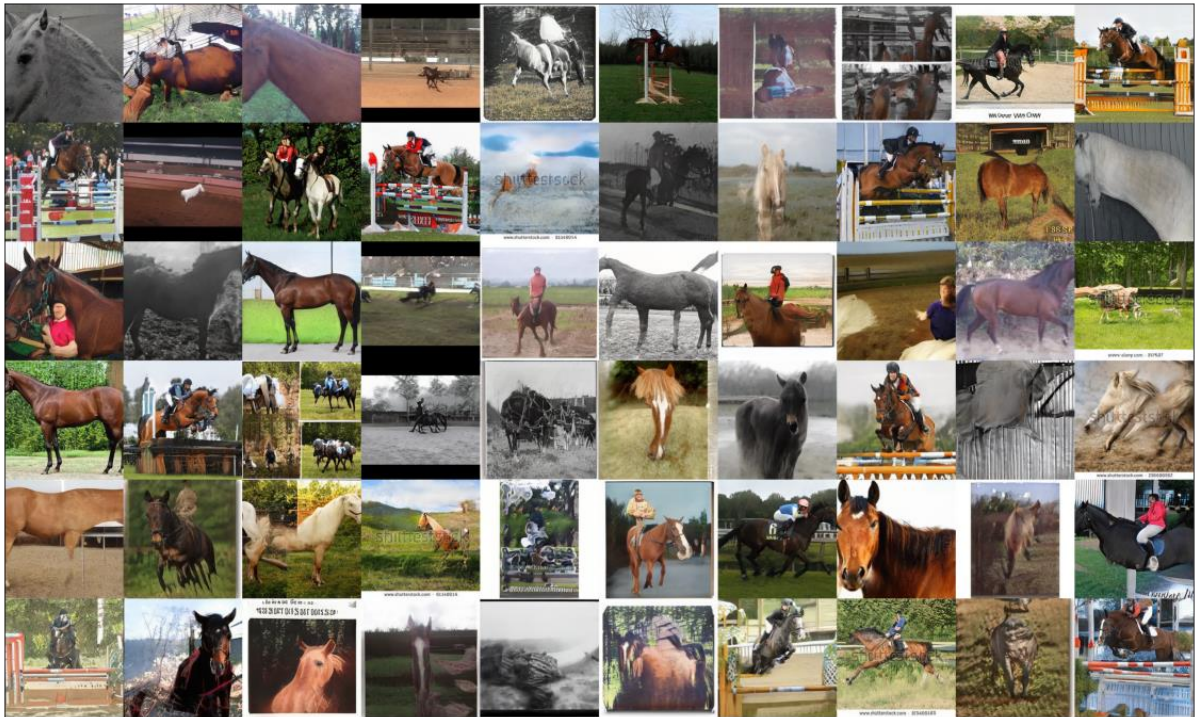
Figure 6.18: Generations depending on the magnitude of CLA in StyleGAN2 trained with LSUN-Cat. We identify low CLA group has higher visual fidelity compared to high CLA group.



## 6.9 Artifact Detection in StyleGAN2 trained with LSUN-Horse



(a) Bottom 25-84 generations in low CLA group.

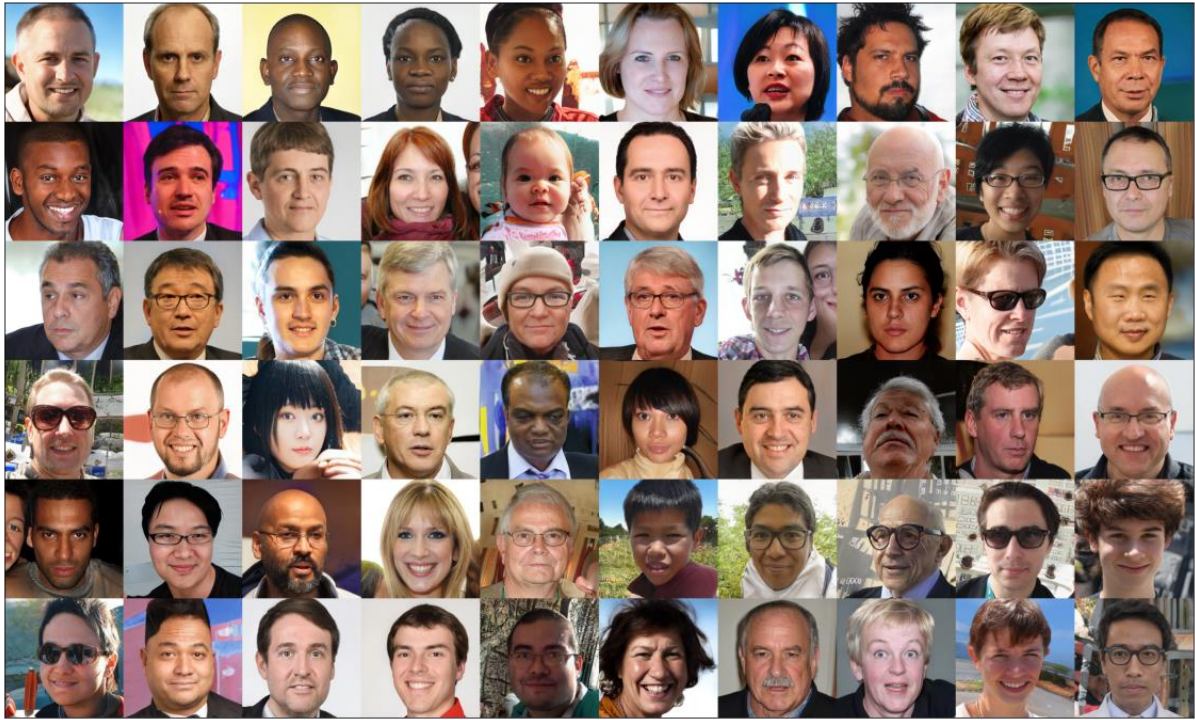


(b) Top 25-84 generations in High CLA group.

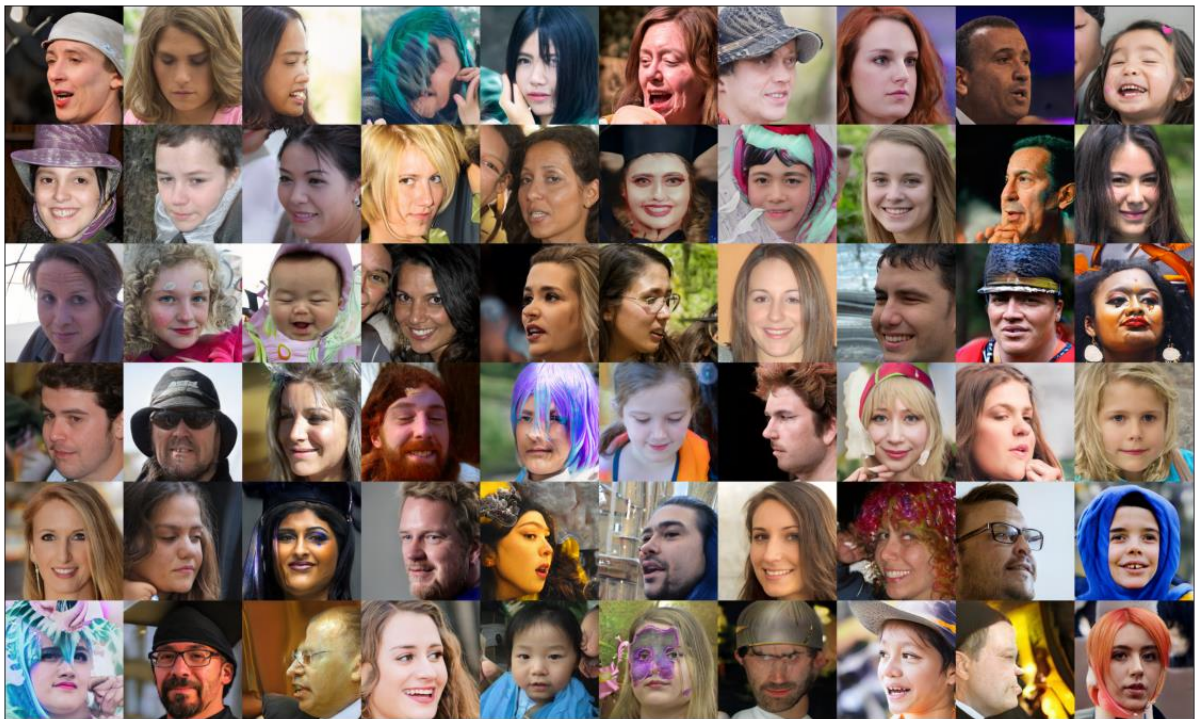
Figure 6.19: Generations depending on the magnitude of CLA in StyleGAN2 trained with LSUN-Horse. We identify low CLA group has higher visual fidelity compared to high CLA group.



### 6.10 Artifact Detection in StyleGAN2 trained with FFHQ



(a) Bottom 25-84 generations in low CLA group.

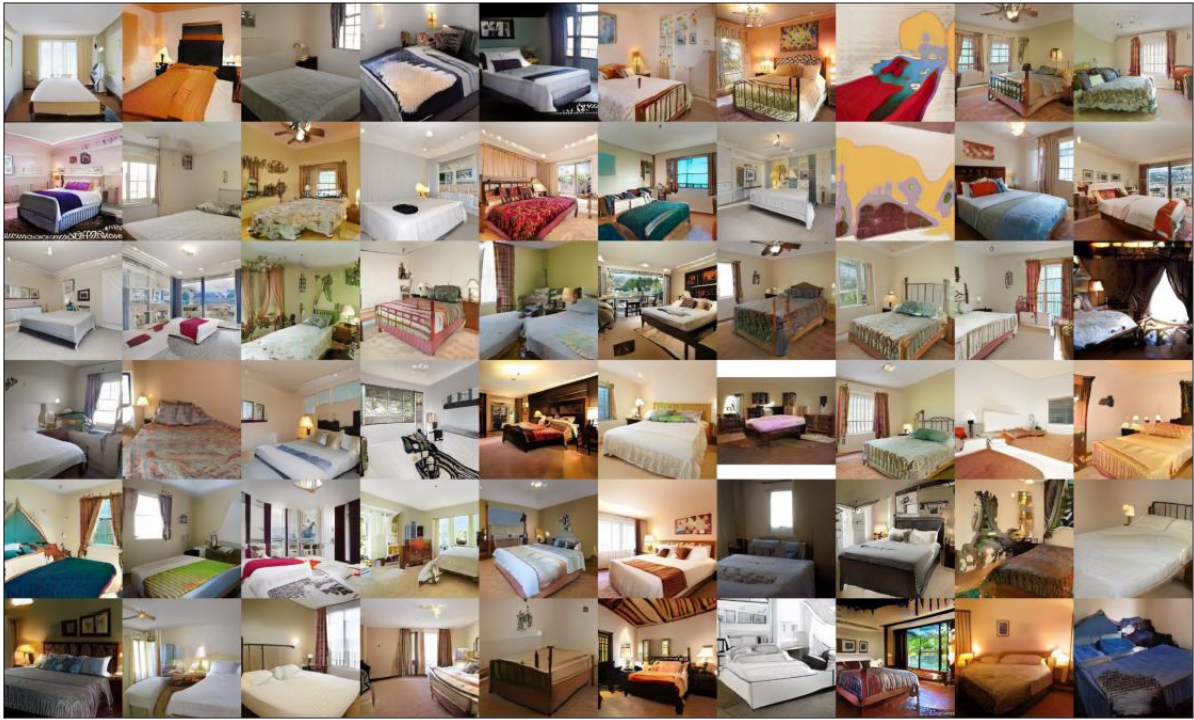


(b) Top 25-84 generations in High CLA group.

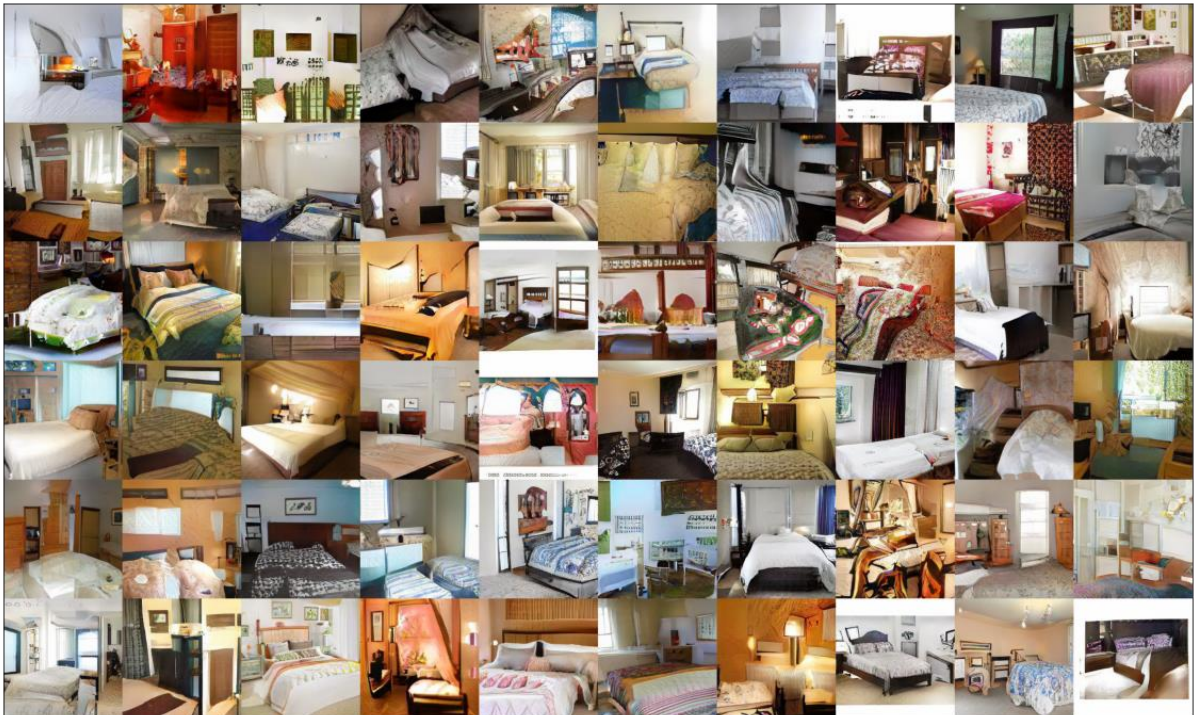
Figure 6.20: Generations depending on the magnitude of CLA in StyleGAN2 trained with FFHQ. We identify low CLA group has higher visual fidelity compared to high CLA group.



### 6.11 Artifact Detection in PGGAN trained with LSUN-Bedroom



(a) Bottom 25-84 generations in low CLA group.

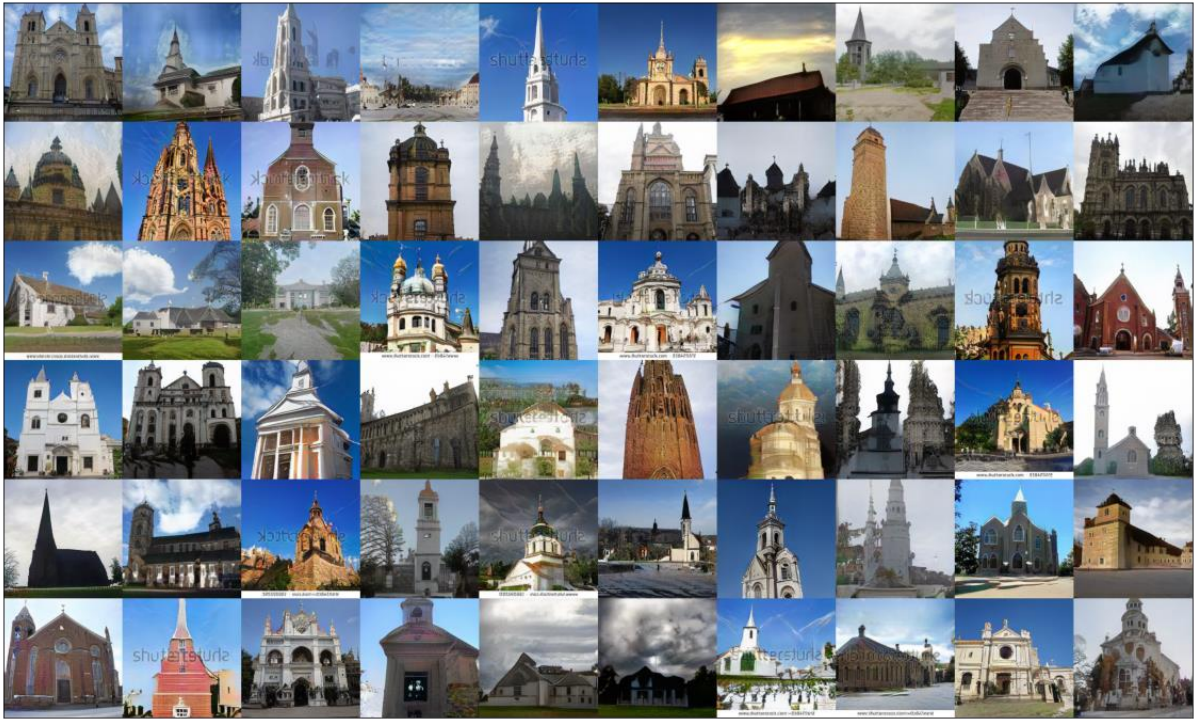


(b) Top 25-84 generations in High CLA group.

Figure 6.21: Generations depending on the magnitude of CLA in PGGAN trained with LSUN-Bedroom. We identify low CLA group has higher visual fidelity compared to high CLA group.



## 6.12 Artifact Detection in PGGAN trained with LSUN-Church



(a) Bottom 25-84 generations in low CLA group.

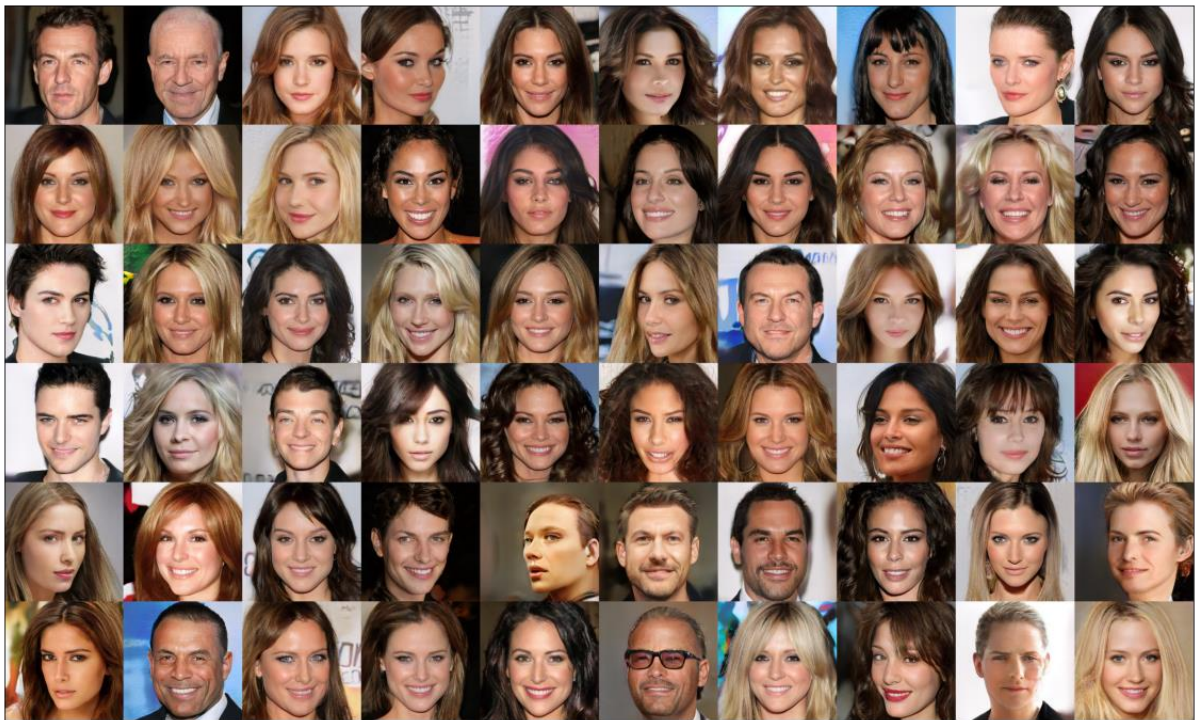


(b) Top 25-84 generations in High CLA group.

Figure 6.22: Generations depending on the magnitude of CLA in PGGAN trained with LSUN-Church. We identify low CLA group has higher visual fidelity compared to high CLA group.



### 6.13 Artifact Detection in PGGAN trained with CelebA-HQ



(a) Bottom 25-84 generations in low CLA group.



(b) Top 25-84 generations in high CLA group.

Figure 6.23: Generations depending on the magnitude of CLA in PGGAN trained with CelebA-HQ. We identify low CLA group has higher visual fidelity compared to high CLA group.



## 6.14 Artifact Correction with CLA

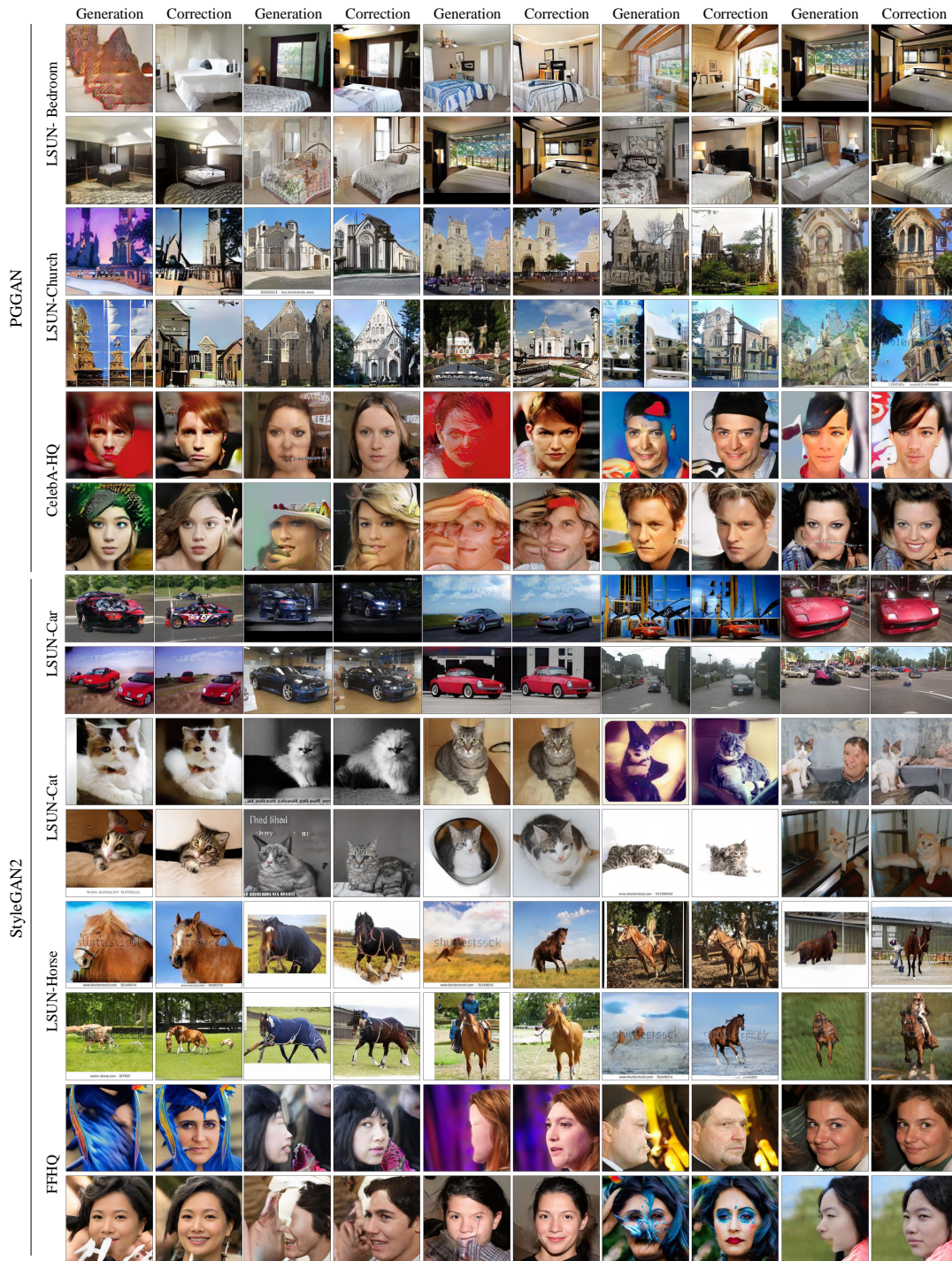


Figure 6.24: An illustrative examples for the results of artifact correction for high CLA group on GANs trained with various dataset. We note that the correction is performed with only internal information of the trained generator.

## References

- [1] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [2] Y. LeCun, C. Cortes, and C. Burges, “Mnist handwritten digit database,” *AT&T Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, vol. 2, p. 18, 2010.
- [3] D. Bau, J.-Y. Zhu, H. Strobelt, Z. Bolei, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, “Gan dissection: Visualizing and understanding generative adversarial networks,” in *International Conference on Learning Representations*, 2019.
- [4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Conference on Neural Information Processing Systems*, 2014, pp. 2672–2680.
- [5] A. Brock, J. Donahue, and K. Simonyan, “Large scale GAN training for high fidelity natural image synthesis,” in *International Conference on Learning Representations*, 2019. [Online]. Available: <https://openreview.net/forum?id=B1xsqj09Fm>
- [6] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” in *International Conference on Learning Representations*, 2018. [Online]. Available: <https://openreview.net/forum?id=Hk99zCeAb>
- [7] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *CVPR*, 2017.
- [9] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng, “Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks,” in *Proceedings of the International Conference on Artificial Neural Networks*, 2019.
- [10] H. Kim, Y. Choi, J. Kim, S. Yoo, and Y. Uh, “Exploiting spatial dimensions of latent in gan for real-time image editing,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

- [11] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, “Network dissection: Quantifying interpretability of deep visual representations,” *arXiv preprint arXiv:1704.05796*, 2017.
- [12] C. Olah, A. Mordvintsev, and L. Schubert, “Feature visualization,” *Distill*, 2017, <https://distill.pub/2017/feature-visualization>.
- [13] Y. Shen, J. Gu, X. Tang, and B. Zhou, “Interpreting the latent space of gans for semantic face editing,” in *CVPR*, 2020.
- [14] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, “Ganspace: Discovering interpretable gan controls,” in *Proc. NeurIPS*, 2020.
- [15] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao, “Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop.” *CoRR*, 2015.
- [16] C.-H. Chang, E. Creager, A. Goldenberg, and D. Duvenaud, “Explaining image classifiers by adaptive dropout and generative in-filling,” *arXiv preprint arXiv:1807.08024*, 2018.
- [17] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” Computer Science Department, Iowa State University, Tech. Rep., 1998.
- [18] D. Erhan, A. Courville, and Y. Bengio, “Understanding representations learned in deep architectures,” *Department dInformatique et Recherche Operationnelle, University of Montreal, QC, Canada, Tech. Rep*, vol. 1355, p. 1, 2010.
- [19] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, and M. Detyniecki, “The dangers of post-hoc interpretability: Unjustified counterfactual explanations,” *arXiv preprint arXiv:1907.09294*, 2019.
- [20] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *IEEE International Conference on Computer Vision*, 2017, pp. 618–626.
- [21] G. Jeon, H. Jeong, and J. Choi, “An efficient explorative sampling considering the generative boundaries of deep generative neural networks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 4288–4295.
- [22] A. Tousi, H. Jeong, J. Han, H. Choi, and J. Choi, “Automatic correction of internal units in generative neural networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.
- [23] H. Jeong, J. Han, and J. Choi, “An unsupervised way to understand artifact generating internal units in generative neural networks,” *arXiv preprint arXiv:2112.08814*, 2021.
- [24] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” *arXiv preprint arXiv:1312.6199*, 2013.



- [25] T. Karras, T. Aila, and Laine, “Progressive growing of gans for improved quality, stability, and variation,” *arXiv preprint arXiv:1710.10196*, 2015.
- [26] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, “On the number of linear regions of deep neural networks,” in *Conference on Neural Information Processing Systems*, 2014, pp. 2924–2932.
- [27] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. S. Dickstein, “On the expressive power of deep neural networks,” in *International Conference on Machine Learning*, 2017, pp. 2847–2854.
- [28] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz, and M. Botvinick, “On the importance of single directions for generalization,” *arXiv preprint arXiv:1803.06959*, 2018.
- [29] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *International Conference on Machine Learning*, 2017, pp. 214–223.
- [30] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, “Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop,” *arXiv preprint arXiv:1506.03365*, 2015.
- [31] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *International Conference on Computer Vision*, 2015.
- [32] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.
- [33] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, “Generative visual manipulation on the natural image manifold,” in *European Conference on Computer Vision*, 2016, pp. 597–613.
- [34] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.
- [35] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, “Explaining nonlinear classification decisions with deep taylor decomposition,” *Pattern Recognition*, vol. 65, pp. 211–222, 2017.
- [36] M. T. Ribeiro, S. Singh, and C. Guestrin, “Why should i trust you?: Explaining the predictions of any classifier,” in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.
- [37] M. Ravanelli and Y. Bengio, “Interpretable convolutional filters with sincnet,” *arXiv preprint arXiv:1811.09725*, 2018.

- [38] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*, 2014, pp. 818–833.
- [39] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, “Synthesizing the preferred inputs for neurons in neural networks via deep generator networks,” in *Conference on Neural Information Processing Systems*, 2016, pp. 3387–3395.
- [40] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto, “Empirical study of the topology and geometry of deep networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [41] N. Lei, Z. Luo, S.-T. Yau, and D. X. Gu, “Geometric understanding of deep learning,” *arXiv preprint arXiv:1805.10451*, 2018.
- [42] K. Dvijotham, R. Stanforth, S. Gowal, T. A. Mann, and P. Kohli, “A dual approach to scalable verification of deep networks.” in *UAI*, vol. 1, no. 2, 2018, p. 3.
- [43] A. Odena, V. Dumoulin, and C. Olah, “Deconvolution and Checkerboard Artifacts,” *Distill*, vol. 1, no. 10, p. e3, Oct 2016.
- [44] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium,” in *Advances in neural information processing systems*, 2017, pp. 6626–6637.
- [45] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, “Network dissection: Quantifying interpretability of deep visual representations,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [46] R. Fong and A. Vedaldi, “Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [47] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, “Improved precision and recall metric for assessing generative models,” *CoRR*, vol. abs/1904.06991, 2019.
- [48] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of StyleGAN,” in *Proc. CVPR*, 2020.
- [49] E. Schonfeld, B. Schiele, and A. Khoreva, “A u-net based discriminator for generative adversarial networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8207–8216.
- [50] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q.

- Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: <https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf>
- [51] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein Generative Adversarial Networks,” in *International Conference on Machine Learning*. PMLR, Jul 2017, pp. 214–223. [Online]. Available: <http://proceedings.mlr.press/v70/arjovsky17a.html>
- [52] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/file/892c3b1c6dcd52936e27cbd0ff683d6-Paper.pdf>
- [53] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham, 2014, pp. 818–833.
- [54] A. Karpathy, J. Johnson, and L. Fei-Fei, “Visualizing and Understanding Recurrent Networks,” *arXiv*, Jun 2015. [Online]. Available: <https://arxiv.org/abs/1506.02078v2>
- [55] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, and W. Samek, “The lrp toolbox for artificial neural networks,” *Journal of Machine Learning Research*, vol. 17, no. 114, pp. 1–5, 2016. [Online]. Available: <http://jmlr.org/papers/v17/15-618.html>
- [56] X. Zhang, S. Karaman, and S.-F. Chang, “Detecting and simulating artifacts in gan fake images,” in *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2019, pp. 1–6.
- [57] Y. Li and S. Lyu, “Exposing deepfake videos by detecting face warping artifacts,” in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.
- [58] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” in *Proceedings of the Conference on Neural Information Processing Systems*, 2016.
- [59] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium.” in *Proceedings of the Advances in Neural Information Processing Systems*, 2017.
- [60] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly, “Assessing generative models via precision and recall.” in *Proceedings of the Advances in Neural Information Processing Systems*, 2018.



- [61] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, “Improved precision and recall metric for assessing generative models,” in *Proceedings of the Advances in Neural Information Processing Systems*, 2019.
- [62] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- [63] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” in *Proceedings of the International Conference on Learning Representations*, 2014.
- [64] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018.
- [65] J. D. M.-W. C. Kenton and L. K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2019.
- [66] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” in *Proceedings of the Advances in Neural Information Processing Systems*, 2020.
- [67] W. Peebles, J. Peebles, J.-Y. Zhu, A. Efros, and A. Torralba, “The hessian penalty: A weak prior for unsupervised disentanglement,” in *Proceedings of the European Conference on Computer Vision*, 2020.
- [68] A. Borji, “Pros and cons of gan evaluation measures,” *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, 2019.
- [69] H. He and W. Su, “The local elasticity of neural networks,” in *International Conference on Learning Representations*, 2020. [Online]. Available: <https://openreview.net/forum?id=HJxMYANtPH>
- [70] Y. Song and S. Ermon, “Generative modeling by estimating gradients of the data distribution,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [71] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-based generative modeling through stochastic differential equations,” in *International Conference on Learning Representations*, 2021. [Online]. Available: <https://openreview.net/forum?id=PxDIG12RRHS>

- [72] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

## Acknowledgements

First of all, I would like to express my special thanks of gratitude to my advisor Jaesik Choi who teaches me not only the knowledge about eXplainable Artificial Intelligence (XAI) or how to write a scientific paper but more importantly the way how to become a researcher. Secondly I would like to appreciate Giyoung Jeon and Jiyeon Han, for all the countless efforts to build and develop the work with me besides the gentle discussions. I would also like to give many thanks to my colleagues in Statistical Artificial Intelligence Lab who support me by making comments, having discussions, giving warm encourages in many other ways. For the last but not the least, I would like to thank my parents and friends who helped me a lot emotionally and physically to finish my thesis.



