Development of Components for Generative Adversarial Networks in Medical Image Synthesis

Artur Sahakyan

A thesis presented for the degree of
Bachelor of Science in Computer Science

Supervisor
Varduhi Yeghiazaryan

Zaven and Sonia Akian College of Science and Engineering
American University of Armenia
Armenia
Spring 2024

Acknowledgments

My special thanks to Varduhi Yeghiazaryan for her professional supervision, sharing experience, inviting to speeches and helpful comments.

I am also grateful to AUA faculty for the knowledge and skills (Suren Khachatryan (Image Processing), Habet Madoyan (Data Science), Hayk Nersisyan (Differential Equations), Elen Vardanyan (Machine Learning)), AUA Student Union and Kevork Kojoyan (for supportive work environment at the campus)). Also extending thanks to DeepLearningAI (Sharon Zhou, Andrew Ng) for 3 courses on 'GANs specialization' and Armenian Code Academy (Vahan Huroyan, Nairi Hakobyan) for the 'Machine Learning' course.

Abstract

Medical imaging helps healthcare providers detect diseases, injuries. Given the availability of data, many tasks, such as tumor detection, disease classification, diagnosis can be automated with the help of machine learning. In many cases though, data is scarce, there are privacy issues or the modality is not suitable for the task. Given the rise of artificial intelligence, it has become possible to address those issues through the introduction of generative models. In this report, an adversarial approach for generating computer tomography (CT) scans is analysed. The project GANs model CT scans using training images, without providing conditional information, such as segmentation maps or edges. Components proposed are applied to larger generative adversarial networks and have been used for grayscale CT scan synthesis. The results are then evaluated with the help of statistical methods.

Contents

Chapter 1

Introduction

In general, image generation (synthesis) has seen a dramatic rise in fidelity, resolution during the last 3 years (2022–2024), with the advent of new approaches. Simultaneously, image generation methods have been applied for different tasks in medical imaging, such as magnetic resonance imaging (MRI) to computed tomography(CT) translation, brain tumor data augmentation, lesion in liver classification, segmentation of organs, detection of diseases, etc.

This report focuses on the adversarial approach of generative modeling of medical images for data augmentation purposes, where the output of the generator is evaluated (by a discriminator) and updated during training. Generative adversarial networks(GANs) are quite flexible. Researchers have introduced methods that allow GANs to synthesise high resolution, results very close to provided training images (in statistical measures [YZD21]).

Several issues persist though in medical image synthesis, such as poor quality of outputs when no additional information (such as segmentation masks or edge information) is provided to the neural network. Typical architectures do not output the desired result, have obvious problems and modifications are needed to achieve successful data augmentation.

This report showcases techniques, which enhance the process of image synthesis and output grayscale (single-channel) CT scans. The central goal is to create a GAN which can augment scarce data while preserving fidelity and diversity of images. For each method proposed it is then shown what type of improvement it provides with the help of numerical (statistical measures) scores and visualizations.

The methods proposed can then be applied to different types of GANs.

Chapter 2

Review of Methods

2.1 Generative Adversarial Networks

According to Goodfellow et al. [GPAM+14, GPAM+20] generative adversarial networks are a kind of artificial intelligence algorithm designed to solve the generative modeling problem. Discriminative models have become successful yet many difficulties with generative approach are unresolved [GPAM+14]. Their proposed method of GANs helps to overcome uncertainty around the estimation of maximum likelihood and intractable probabilistic computations. GANs sidestep those difficulties due to adversarial training (the generator gets evaluation from the discriminator). GANs learn the probability distribution from training examples and then generate more samples by modeling the learnt probability distribution.

GANs are among the most successful generative models (in terms of generating realistic images) [GPAM+20], but still present unique challenges and research opportunities. Given a task formulation, there is a wide range of GANs to select from, but custom modifications are still needed. Despite significant advances in constrained optimization of parameters of GANs, non-convergence of parameters remains an issue, "vanishing"(tends to 0) and "exploding"(tends to ∞) gradients may occur due to a large size (greater than 64×64 in width and height) of images and training data. GAN training stability still remains an open issue, for instance, if the size of images is larger (greater than 64 by 64 in width and height). State-of-the-art GANs (StyleGAN2 for instance) require large amounts of GPU memory.

GANs are applied in various tasks, and in the context of this report we shall be exploring their application in medical imaging. GAN applications in medical imaging raise issues of anatomical accuracy, edge preservation in images, mode collapse (during multi-modal synthesis). In comparison with another generative modeling approach–diffusion models (score-based models), GANs provide less diversity in outputs ([DN21]).

After the introduction of GANs researchers have come up with different types of GANs, such as Deep Convolutional GAN(DCGAN), Wasserstein GAN(WGAN),

StyleGAN, BigGAN, ProGAN, etc., trying to improve results or solve certain tasks.

2.1.1 GAN Structure

Generator and Discriminator

The typical GAN framework consists of a generator and a discriminator. The generator models a distribution, while the discriminator evaluates it (Figure 2.1). The result of evaluation can be binary (0,1), or any real number between 0 and 1.

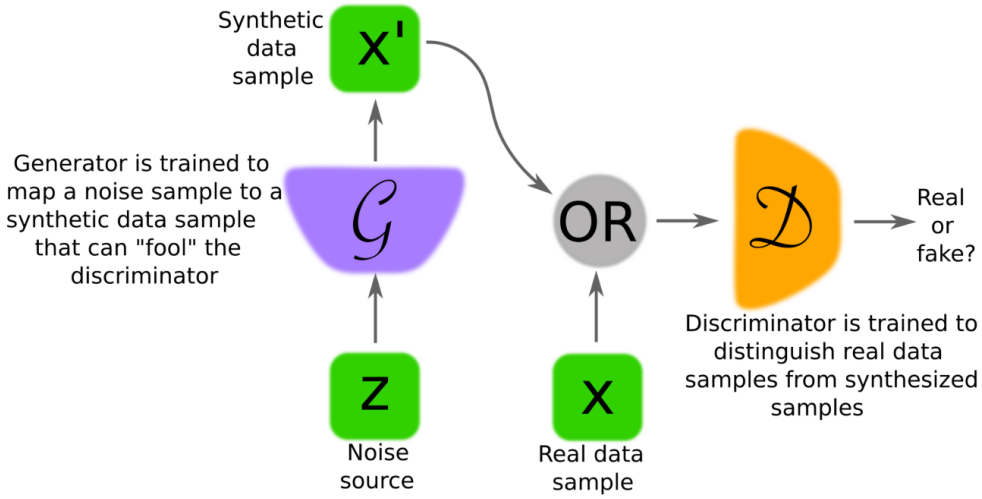

Figure 2.1: In this figure, the two models which are learned during the training process for a GAN are the discriminator (D) and the generator (G). Figure reproduced from [CWD+18].

Critic

Critic is a type of a discriminator which evaluates the proximity of fake distribution to the real distribution on the real line, not in range $[0, 1]$.

2.1.2 Loss Functions

Binary Cross Entropy Loss (BCE Loss)

In the first implementations of GANs BCE Loss (Equation A.1) is utilized to evaluate whether the image (distribution) is real or fake. The problem of evaluation is formulated as a binary classification task (fake (0) or real (1)).

BCE Loss as Min–Max optimization problem

For GANs the optimization problem is formulated as follows:

$$\mathbb{L}_{bce} = \min_{D} \max_{G} -[\mathbb{E}(\log(D(x))) + \mathbb{E}(1 - \log(D(G(z))))] \tag{2.1}$$

where D stands for discriminator, G for generator, z for input random noise.

The output of discriminator is optimized to be minimal, while generator output is maximized.

Wasserstein Loss or Earth Mover's Distance

$$EM(G, C, x, z) = \min_G \max_C [\mathbb{E}(C(x)) - \mathbb{E}(C(G(z)))] \tag{2.2}$$

G stands for generator, $C(x)$ is critic's (introduced below) output for real image, $C(G(z))$ is critic's output for generated image with input noise z.

The critic tends to maximize the distance, while the generator tends to minimize the distance. Intuitively, the formula shows how much 'mass' needs to be transported from generated distribution towards real distribution so that those two match.

2.1.3 Common Operations

Among common operations that are used to construct the generator and the discriminator are convolutional operations. A short explanation for them is provided below.

Convolution and Transposed Convolution

Linear convolution is an operation, which combines two functions of same dimensionality, either discrete or continuous.

For discrete, 2D functions I and H, the convolution operation is defined as

$$I^{'}(u, v) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(u - i, v - j) \cdot H(i, j) \tag{2.3}$$

and can be expressed with designated convolution operator $(*)$

$$I^{'} = I * H \tag{2.4}$$

Transposed convolution is a fractionally-strided convolution. It is similar to the convolution process with a modified input map used in upsampling of images and it tries to "reverse" the convolution process(which in the context of GANs is mainly used for downsampling of images in the discriminator or the critic). Transposed convolution is thought of as "deconvolution" or reverse convolution, since it is used to simulate the reverse process, but is not the true reverse of convolution and should not be confused with the term "deconvolution".

Visualizations of applications of the convolution operation can be observed in Figure 2.2 (δ Dirac function applied) and Figure 2.3 (image blurring).

Truncation trick

The truncation trick allows to select a 'portion' of a random normal distribution.

Graphically, it can be shown as in Figure 2.4 and for mathematical formulation please refer to Appendix A.

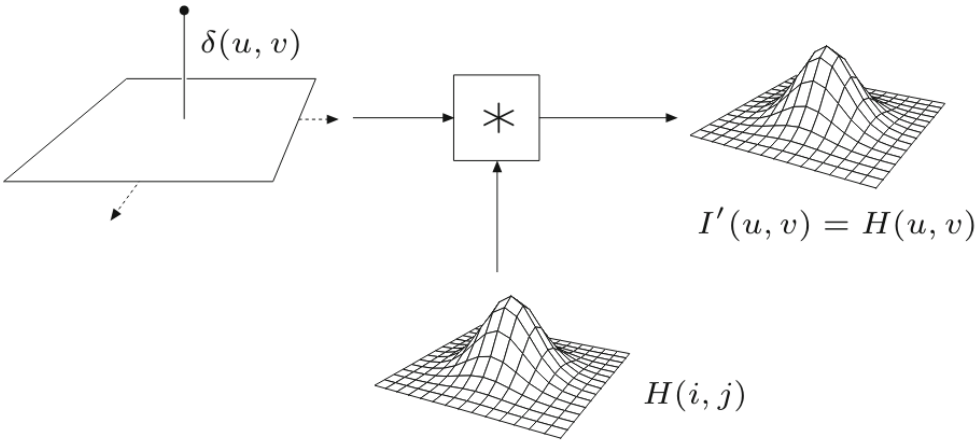

Figure 2.2: The linear filter H with the impulse δ as the input yields the filter kernel H as the result (Figure and caption adopted from [BB22]).

.

2.1.4 Deep Convolutional Generative Adversarial Network

First GANs (Vanilla GAN) are composed of linear layers, mapping input noise to the first linear(affine) layer and then upsampling it in succeeding layers (Figure 2.5).

Radford et al. create Deep Convolutional GAN (DCGAN) [RMC15] introducing

- all convolutional neural network, which replaces deterministic spatial pooling functions (upsampling) with strided convolutions,
- global average pooling for eliminating fully connected layers on top of convolutional layers,
- batch normalization, which stabilizes learning by normalizing the input layers to have zero mean and unit variance.

2.1.5 Conditional GAN

Mirza et al. [MO14] extend the unconditional model to conditional by presenting additional information $\hat{y}$ to generator and discriminator. In generator the prior input noise $p_z(z)$ and $\hat{y}$ are combined in joint hidden representation. In discriminator x and $\hat{y}$ are presented as inputs. The objective function then becomes

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x}, \hat{\mathbf{y}})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z}, \hat{\mathbf{y}})))] \quad (2.5)$$

which is an extension of Equation 2.1.

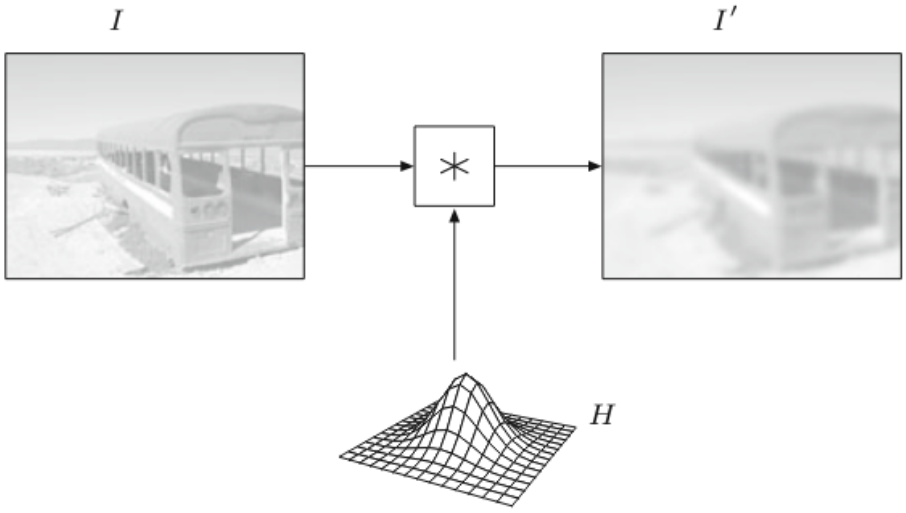

Figure 2.3: Original image subjected I to a linear convolution $*$ with the convolution kernel H outputs image $I^{'}$, long story short - image blurring applied (Figure adopted from [BB22]).

Image-to-Image Translation

Isola et al. [IZZE17] investigate conditional adversarial networks as a general-purpose solution to image-to-image translation. GANs, as they state, "learn a generative model of data, conditional GANs learn a conditional generative model".

For many image translation problems, there is a great deal of low-level information shared between the input and output, and it would be desirable to shuttle this information directly across the network.

That is why they add skip connections, where each skip connection concatenates all channels at layer i with those at layer $n - i$.

They also update the discriminator to work with $N \times N$ patches, calling it "PatchGAN", which they run convolutionally across the image, averaging all the results to provide the ultimate output of discriminator D.

2.1.6 Wasserstein GAN

Arjovsky et al. [ACB17] focus on loss functions, measures of distance between real and fake images. They list different formulas and they point out their disadvantages, trying to come up with a better loss function. Among those distance measures are Total Variation distance (Eq. A.3), Kullback-Leibler distance (Eq. A.4), Jensen-Shannon distance (Eq. A.5). Those measures have disadvantages for ensuring training stability.

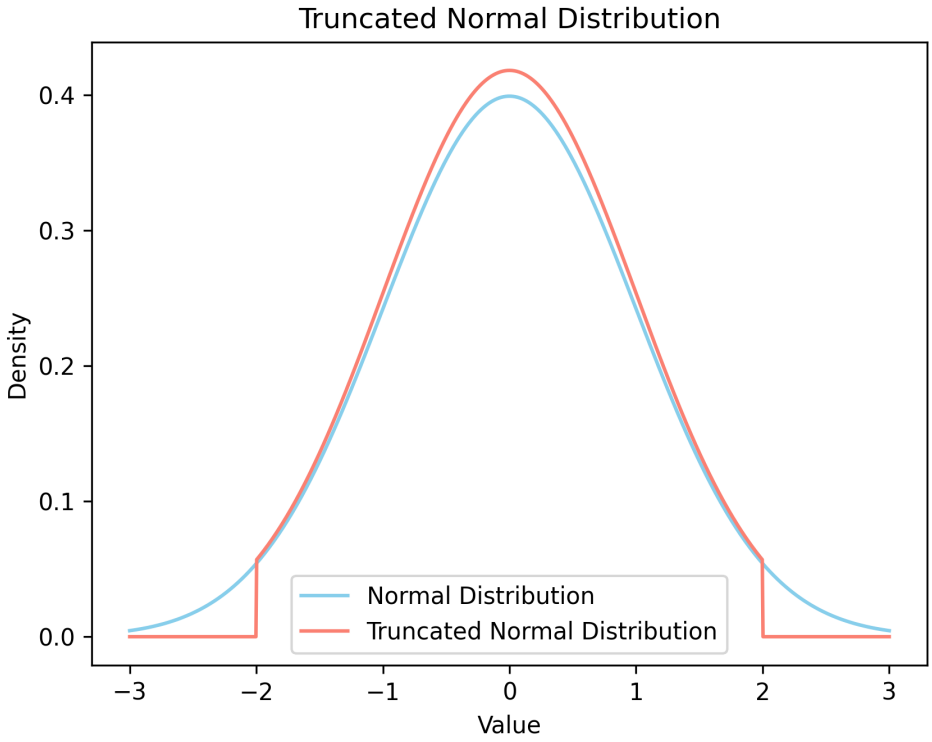

Figure 2.4: Plot of truncated normal distribution

Arjovsky et al. [ACB17] formulate the loss function as

$$W(Pr, P_\theta) = \sup_{\|f\|_{L\leq 1}} \mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_\theta}[f(x)] \tag{2.6}$$

which is another way of writing the Earth Mover's distance (Equation 2.2), P_r stands for real distribution, P_θ for generated distribution.

In general, by replacing $\|f\|_{L\leq 1}$ with $\|f\|_{L\leq K}$ we can define K-Lipschitz functions (those are functions which have bounded derivatives, for further information please read [BS00]), thus for $f : \mathbb{A} \to \mathbb{R}$ we have a uniformly continuous function on $\mathbb{A}$ [BS00].

This way the problem of vanishing gradients is treated.

Gulrajani et al. [GAA+17] introduce a few changes to ensure stable training of WGANs.

Initial WGANs use weight clipping to enforce 1-Lipschitz constraint, address the issue of "exploding" and "vanishing" gradients. But the experiments show critics trained with weight clipping fail to capture higher moments of the data distribution. Gradient norms with weight clipping either explode or vanish on a toy dataset.

Gulrajani et al. propose using a gradient penalty term in the Wasserstein loss function (Equation 2.2), therefore the loss function becomes

$$W(Pr, P_\theta) = \mathbb{E}_{x\sim P_r}[D(x)] - \mathbb{E}_{x\sim P_\theta}[D(x)] + \lambda \cdot \mathbb{E}_{\hat{x}\sim \mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2] \tag{2.7}$$

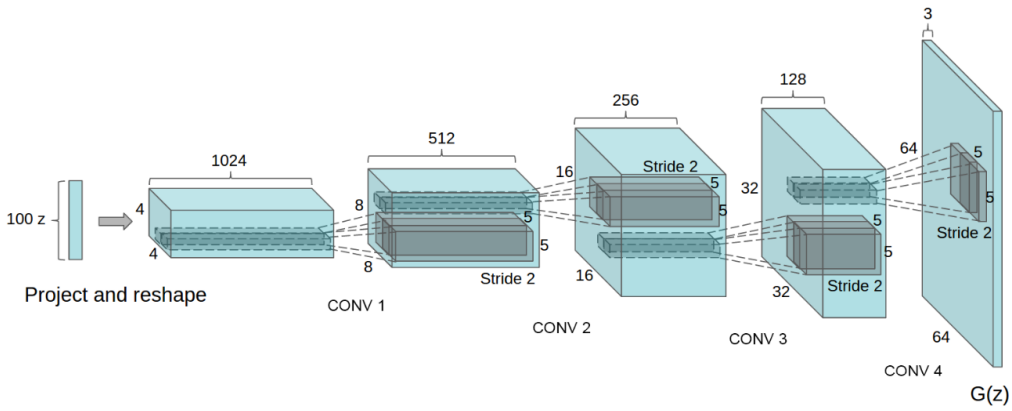

Figure 2.5: An example of DCGAN generator structure. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64 × 64 pixel image. No fully connected layers, pooling layers are used.(hence the name—deep convolutional). (Figure adopted from [RMC15].)

where D denotes the discriminator (critic for WGANs) function. $\mathbb{P}_{\hat{x}}$ denotes the sampling distribution, which is sampled uniformly along straight lines between pairs of points sampled from $\mathbb{P}_r$ and $\mathbb{P}_\theta$.

Experiments show that gradient penalty ensures training stability.

2.1.7 BigGAN

Andrew Brock, Jeff Donahue and Karen Simonyan [BDS18] focus on closing the gap in fidelity and variety between images generated by GANs and real world images from ImageNet.

Their research shows

- GANs benefit dramatically from scaling (4 times more parameters and larger batch size improve generated outputs),
- models become amenable to the 'truncation' trick (truncation of normal distribution 'cuts' input noise range and reduces likelihood of 'bad' samples) (see Equation A.6, Figure 2.6),
- large scale GANs have characteristic to them training instabilities (for some layers spectral norms keep growing),
- Complete training stability can be achieved at a performance cost (steps needed to optimize large amount of parameters during backpropagation with the help of Jacobians (automatic differentiation) [AMB+94]).

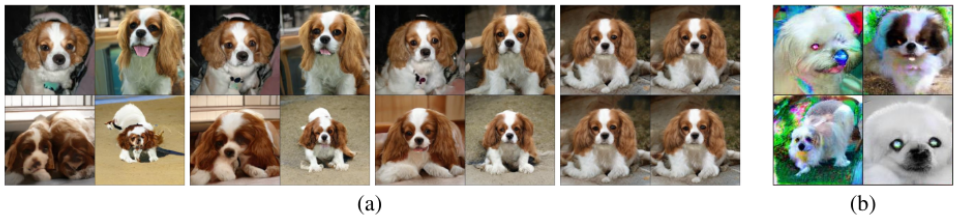

Figure 2.6: a) From left to right the effects of truncation are shown; b) Saturation artifacts from applying truncation to a poorly conditioned model. (Figure and caption adopted from [BDS18])

2.1.8 ProGAN

Karras et al. [KALL17] focus on the issue of GANs being problematic in high resolution image generation. Higher resolution image generation is difficult because it is easier to distinguish fake from real, which drastically amplifies the gradient problem.

Their primary contribution is a training methodology for GANs, where they start with low resolution images and then progressively increase the resolution by adding layers to the networks as visualized in Figure 2.7.

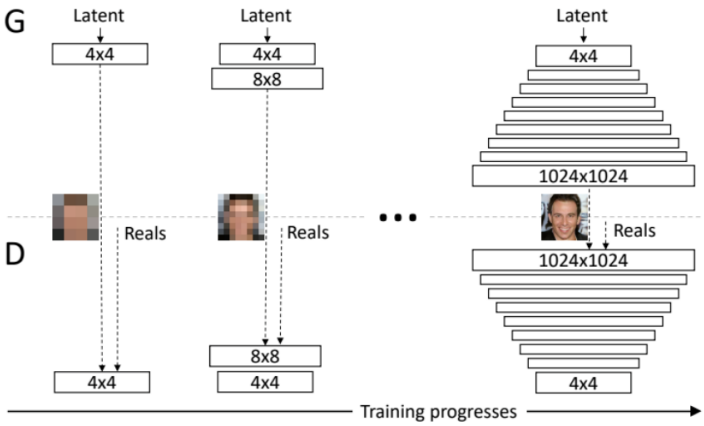

Figure 2.7: ProGAN architecture for 1024×1024 image (Figure adopted from [KALL17]).

2.1.9 StyleGAN

Tero Karras, Samuli Laine, Timo Aila present a novel approach to the Generator architecture [KLA19] inspired by style transfer literature. Their architecture in the first version of StyleGan is based on progressive growth of GANs, linear mapping networks, adaptive instance normalization(Equation A.2) for smoother transition of one image's style to another image's style. The architecture (Figure 2.8) is capable of generating higher resolution images and "competes" with BigGAN in that sense.

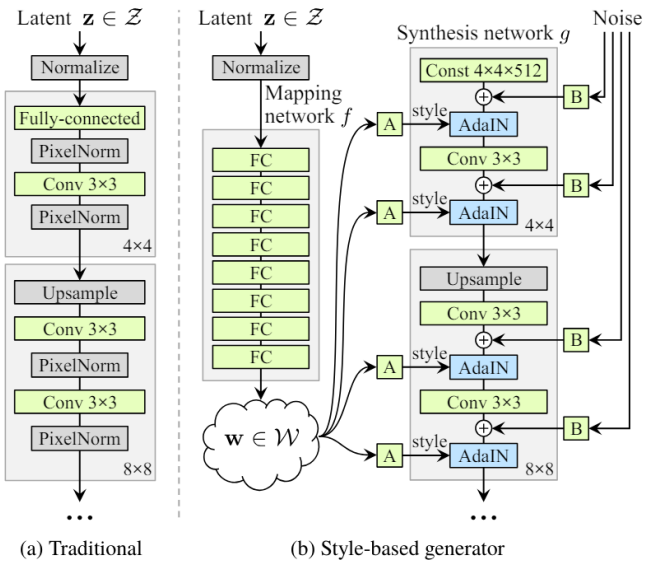

Figure 2.8: While a traditional generator a) feeds the latent code though the input layer only, we first map the input to an intermediate latent space W, which then controls the generator through adaptive instance normalization (AdaIN) at each convolution layer. Gaussian noise is added after each convolution, before evaluating the nonlinearity. Here "A" stands for a learned affine transform, and "B" applies learned per-channel scaling factors to the noise input. The mapping network f consists of 8 layers and the synthesis network g consists of 18 layers— two for each resolution ($4^2 - 1024^2$). The output of the last layer is converted to RGB using a separate 1×1 convolution. (Figure and caption adopted from [KLA19].)

2.2 GAN Application in Medical Imaging

Recently, multiple methods have been proposed in medical image synthesis, which solves the problem of data scarcity for researchers. It also provides enchanced datasets for neural networks tasked to automate detection of diseases, etc:

Another major issue is data privacy, protection of patients' data. Synthesized data solves that issue and encourages data-sharing between institutions. GANs have different applications in the field of medical imaging. In scope of this report the following categories have been researched

- unconditional image synthesis,
- conditional image synthesis,
- segmentation,
- reconstruction,
- registration,
- classification.

2.2.1 Unconditional Image Synthesis

An example of using deep convolutional GANs for medical image synthesis can be observed in the paper by Nie et al. [NTL+18]. Nie et al. present a DCGAN inspired approach for MRI generation (Figure 2.9).

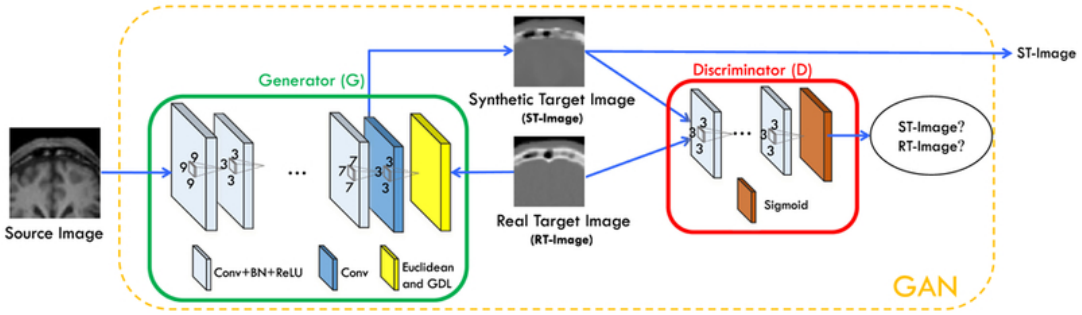

Figure 2.9: Architecture used in the deep convolutional adversarial setting for estimation of the synthetic target image. (Figure adopted from [NTL+18])

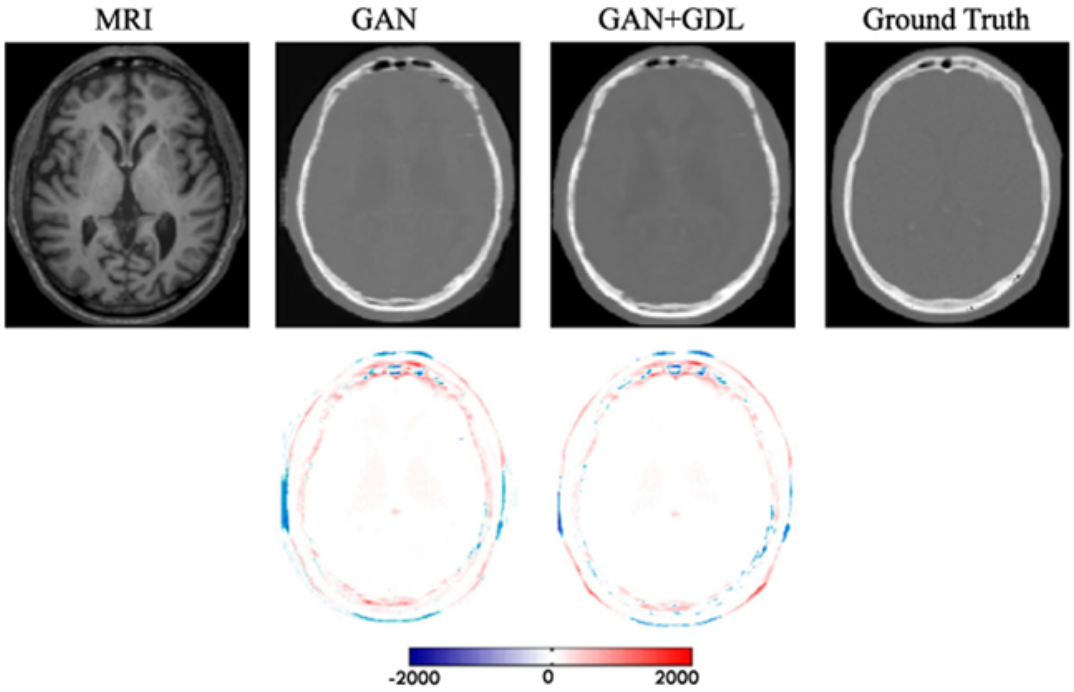

Figure 2.10: Visual comparison for impact of using the gradient difference loss (GDL). The first row shows the input MRI, two synthetic CT by two different methods, and the ground-truth CT. The second row shows difference maps between each synthetic CT and the ground-truth CT. (Figure and caption adopted from [NTL+18])

Additionally, authors define loss function using gradient difference: Equation 2.8.

$$\mathbb{L}_{GDL}(Y, \hat{Y}) = ||\nabla Y_x| - \nabla \hat{Y}_x||^2 + ||\nabla Y_y| - \nabla \hat{Y}_y||^2 + ||\nabla Y_z| - \nabla \hat{Y}_z||^2 \tag{2.8}$$

where $Y, \hat{Y}$ stand for real and generated images. Gradient difference loss helps preserve sharpness of edges (Figure 2.10).

Shin et al. emphasize the role of synthetic medical images in anonymization and protection of patients' personal health information (PHI) [STR+18]. Even removing DICOM metadata is not enough to achieve that goal. Shin et al. emphasize the role of GANs for that purpose, as well as for providing data diversity for training classifiers.

2.2.2 Conditional Image Synthesis

A very interesting implementation of conditional modeling has been done by Yu et al. [YZW+19] in their paper "Ea-GANs: edge-aware generative adversarial networks for cross-modality MR image synthesis". The authors use sobel edge detector to "feed" the information of edges as condition to the generator. As a result the network becomes edge-aware (Figure 2.11).

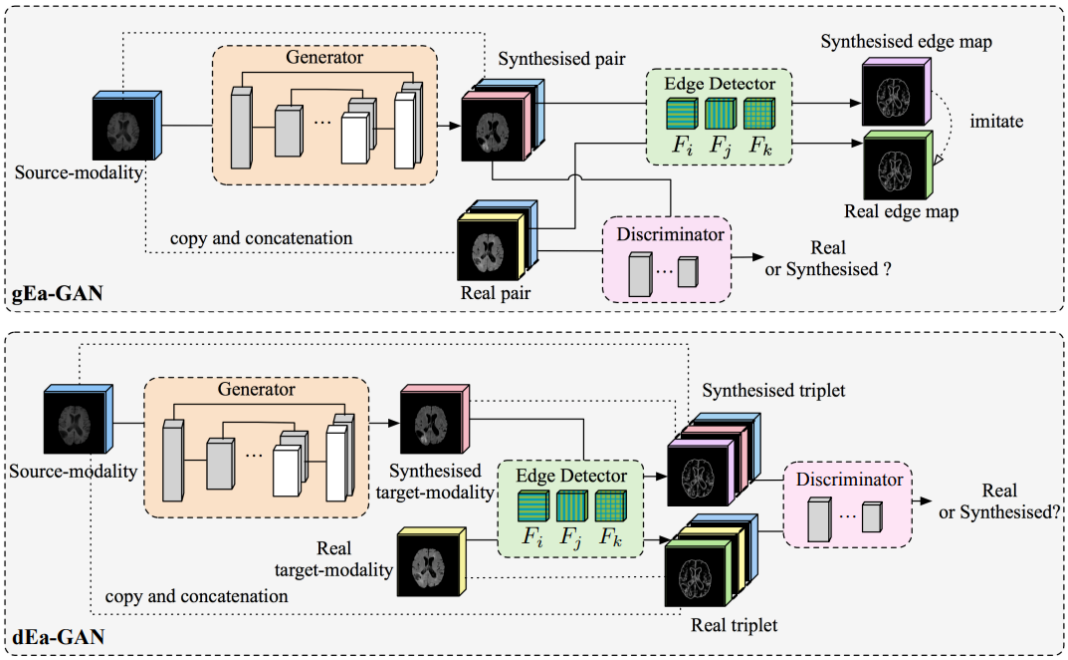

Figure 2.11: The generator G is trained to synthesise a realistic target-modality image with its edge map detected by the Sobel edge detector S, while the discriminator D is learned to distinguish between the synthesised and real pair/triplet for sharp synthesis. In the back-propagation step of training, the generator G of gEa-GAN is affected by the gradients from the dissimilarity between the synthetic and real edge maps, while both generator G and discriminator D of dEa-GAN are affected by the detected edge maps. (Figure and caption adopted from [YZW+19])

Dar et al. [DYK+19] note that in image-to-image translation tasks, the synthesized image has statistical dependence on the source image. The resulting network can then be trained with the following loss function:

$$\mathbb{L}_{condgan} = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,z}[\log(1 - G(D(x,z)))] \tag{2.9}$$

Their synthesis approach, per authors' claim, can help improve quality and versa-

tility of multi-contrast MRI exams without the need for prolonged examinations.

Sun et al. implemented image-to-image translation for medical image synthesis for lesion detection [SWH+20]. Sun et al. create an abnormal-to-normal GAN (Figure 2.12) for generating a normal counterpart of an MRI image where the patient had tumor. This way they try to solve the issue of labeling lesions in images for classification by detecting tumors with the difference of normal and abnormal images.

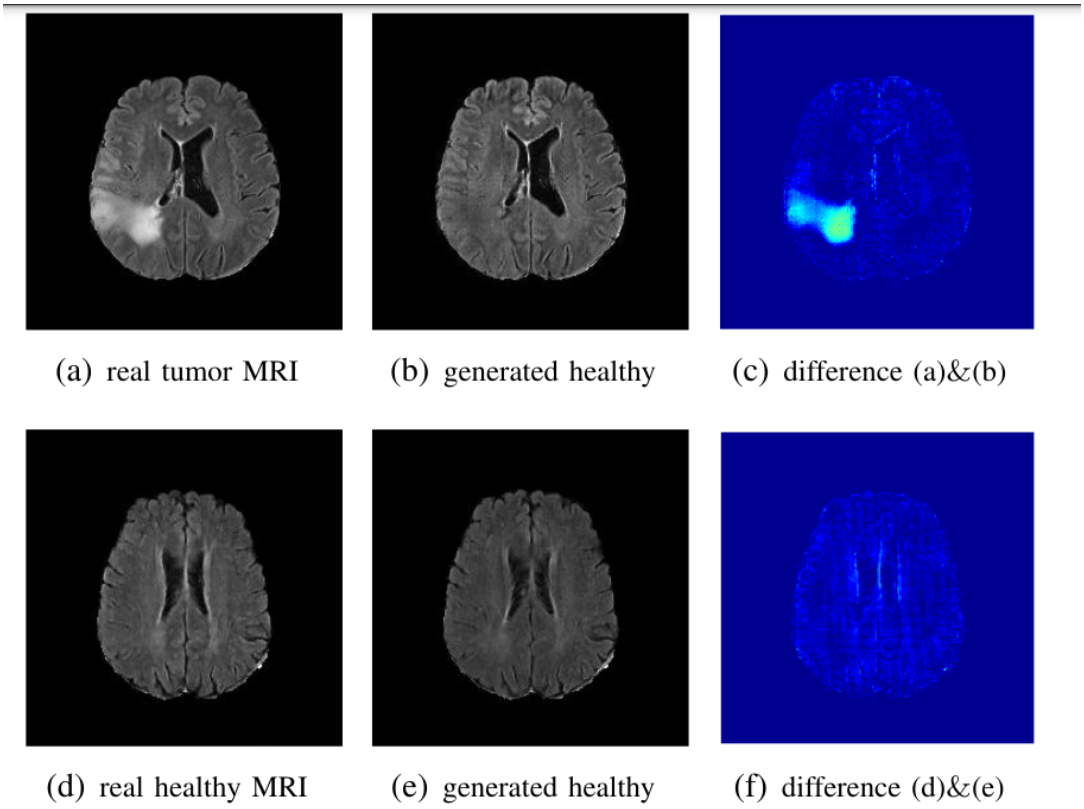

Figure 2.12: Results produced by the model. Lesions are isolated, while healthy regions pass through with minimal modification. (Figure and caption adopted from [SWH+20])

Nie et al. focus on the need of CT scans [NTL+17] , while at the same time pointing out the problem with CT acquisition, due to exposure of patients to radiation and the potential risks coming with it (such as development of cancer as a result of CT scanning).

The proposed solution is image-to-image translation using GANs to convert MRI modality to CT modality. To alleviate the issue of blurriness in generated samples, Nie et al. propose image gradient difference loss, then apply auto context mechanism(ACM) strategy to make the GAN context-aware [NTL+17].

Nie et al. state "One way to enlarge the context during the training is by using the ACM which is commonly used in the task of semantic segmentation and has been shown to be very effective". This way they make the GAN context-aware ("informed" about edges). In this work, Nie et al. show that the "ACM can also be

applied successfully to the regression tasks."

In their experiments the authors show that adversarial training is beneficial in terms of rise in fidelity, compared with fully convolutional neural networks.

2.2.3 Segmentation

Image segmentation is a computer vision technique, which extracts particular regions from an image—to inform object detection and related tasks. Medical image segmentation is for provision of a precise and accurate representation of the objects of interest within the image, mainly for the purposes of diagnosis, treatment planning, and quantitative analysis.

Universal models are demanded increasingly more for medical image segmentations, models that are trained once and applied to a large range of tasks related to segmentation. Current segmentation models have limited capabilities, due to difference between natural images and medical images [MHL+24].

2.2.4 Reconstruction

The mathematical process of generating tomographic images from X-ray projections acquired at different angles around the patient is called image reconstruction.

Image reconstruction impacts the image quality, and therefore the radiation dose. 3D-YNet-GAN and 3D-DenseU-Net reconstruction framework based GAN reconstructs thin slice infant MR image [GLW+19].

2.2.5 Registration

Spatial alignment of two or more image datasets, taken at different times, from different perspectives, different cameras, sensors, is called image registration. According to Suganthi et al. [S+21], parameter dependency and heavy optimization load is the major disadvantage of the registration process. An improved version of WGAN with its image transformation capabilities has been used as an image alignment algorithm in MR-TRUS image registration.

2.2.6 Classification

Image classification is a fundamental computer vision task with an aim of categorizing image as a whole under a specific label. Adversarial training can improve the classification accuracy [JTA+22].

For diagnosing cardiovascular diseases, a GAN architecture called Semi-coupled-GAN has been proposed. It is a combination of DCGAN's generated images and real Chest X-Ray given to deep convolutional neural network to classify abnormali-

ties [ZGF17]. GAN based synthetic data augmentation has a beneficial impact for liver lesion classification with an increase of classification accuracy [FAKA+18].

Chapter 3

Contributions

3.1 General Concept

The architecture of generator presented in this report consists of one affine (linear) layer, with convolutional layers following it.

The affine layer, reshaped into a `batch_size`$\times 1 \times 4 \times 4$ tensor, has been convoluted by the learnable kernel.

The following convolutional layers have been connected with their previous layers, which have been scaled through interpolation, as presented in different versions (for different resolutions of outputs).

3.1.1 Learnable Kernel

The main motivation for applying a random normal kernel (a learnable parameter) is to preserve edges. For the GANs proposed, this filter has been applied on larger GAN's Version 1 and Version 2, while on small GAN its benefits cannot be verified (by statistical measure the results are far from real).

The kernel coefficients depend on the range of values, making this filter a range filter.

The idea of applying this learnable parameter (kernel) is to have both domain and range filters (that are convoluted and learnable). Tomasi and Manduchi [BB22] proposed to combine Gaussian kernels into a common edge-preserving smoothing filter. This combination of filters is usually referred to as bilateral filter, which can be expressed in the form

$$I^{'}(u,v) = \frac{1}{W_{u,v}} \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} H_d(i-u, j-v) \cdot H_r(I(i,j) - I(u,v)) \tag{3.1}$$

where $H_r(I(i,j) - I(u,v)) = w_{i,j}$ (the weights).

In case of Gaussian filters, the bilateral filter takes this form

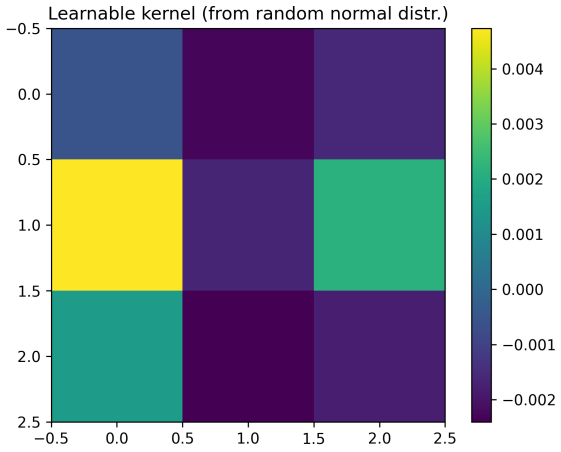

Figure 3.1: Visualization of learnable kernel (initialized from random normal distribution) tensor, which has been applied on generator's first layer

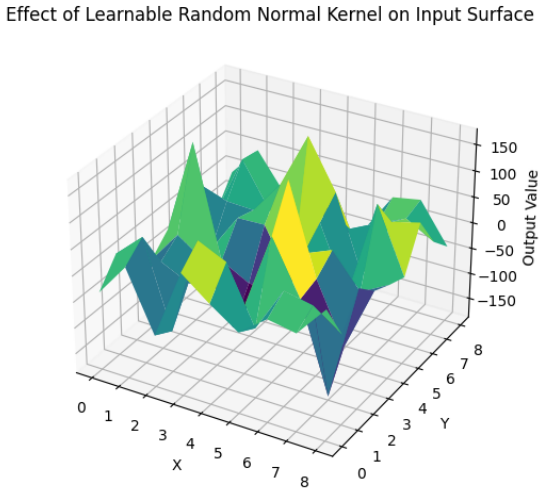

Figure 3.2: Toy example of convolution of a surface with learnable kernel

$$I^{'}(u,v) = \frac{1}{W_{u,v}} \sum_{m=-D}^{D} \sum_{n=-D}^{D} \left[\exp\left(-\frac{m^2+n^2}{2\sigma_d^2} \right) \cdot \exp\left(-\frac{(I(u+m,v+n) - I(u,v))^2}{2\sigma_r^2} \right) \right] \tag{3.2}$$

where $W(u,v) = \left[\exp\left(-\frac{m^2+n^2}{2\sigma_d^2} \right) \cdot \exp\left(-\frac{(I(u+m,v+n)-I(u,v))^2}{2\sigma_r^2} \right) \right]$ and $D = [3.5, \sigma_d]$. The width of σ_d depends on the desired amount of spatial smoothing.

Similarly, the learnable parameter introduced serves as a bilateral filter and has increased the generated image quality (grayscale 128×128 images).

The learnable kernel (Figure 3.1) has transformed random noise (its effect on a surface 3.2) and its effects are visually observed. In Figure 3.3 the small gan's first output before training is shown in grid, in Figure 3.4 the larger gan's first output without learnable kernel applied is shown in grid.

After adding the kernel to larger GAN Version 1 and Version 2 outputs before

learning are shown in Figure 3.5 and Figure 3.6 respectively.

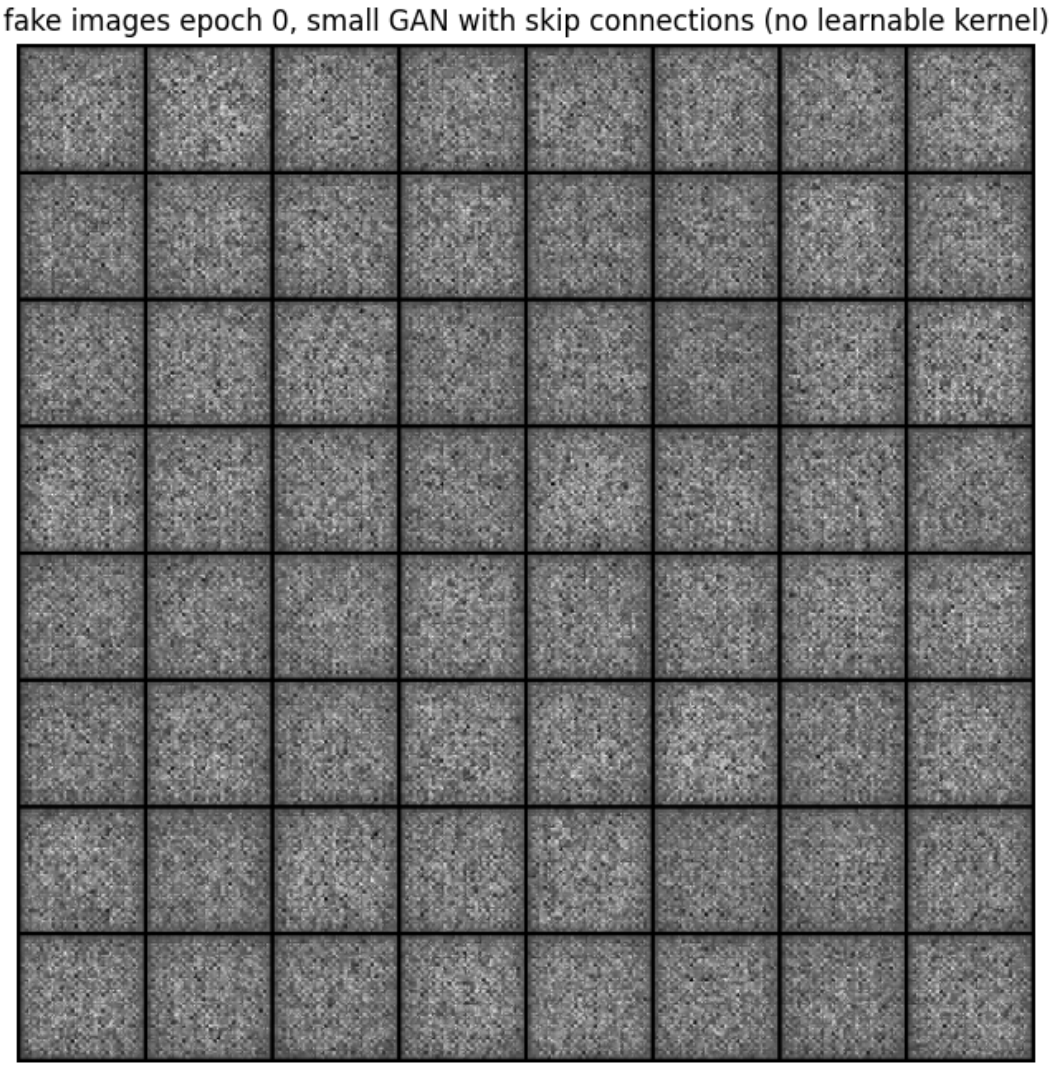

Figure 3.3: Generator output for small gan without learnable kernel applied

3.1.2 Layer Additions

The architecture has been connected in form of

$$F(x) = \begin{cases} x_1 = F_1(x) + x \\ x_1 = x_1 * (learnable\ kernel) \\ x_2 = F_2(x_1) + x_1 + x \\ x_3 = F_3(x_2) + x_2 + x_1 \\ x_4 = F_4(x_3) + x_3 + x_2 \\ x_5 = F_5(x_4) + x_4 + x_3 \\ x_6 = F_6(x_5) + x_5 + x_4 \end{cases} \tag{3.3}$$

When computing the gradient penalty, the fact that the early layers of the network and their weight parameters can produce significant changes in gradient has

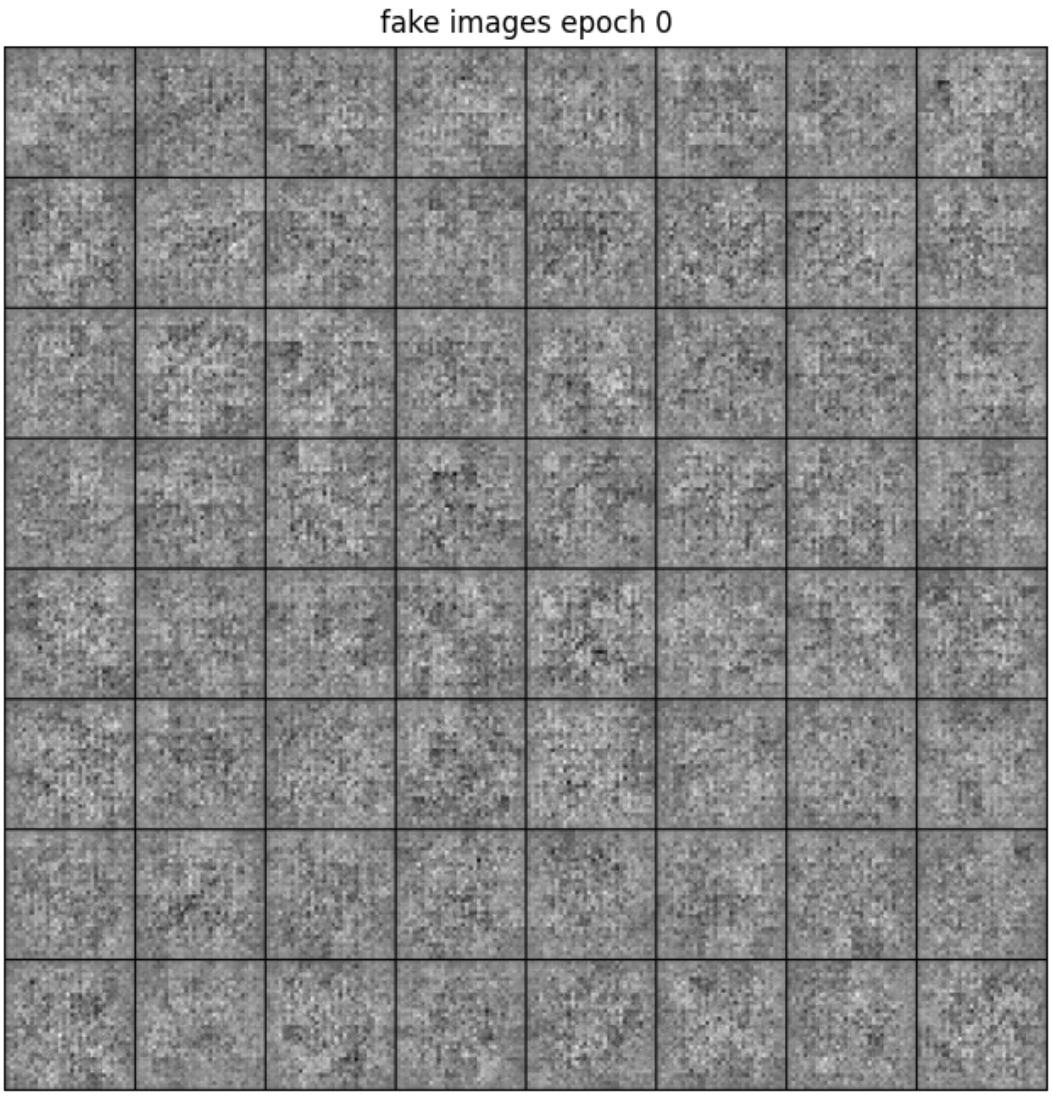

Figure 3.4: Generator output for Larger GAN of Version 1 without learnable kernel applied

been taken into account.

A familiar reader may consider those as being skip connections, used in ResNet, UNet, but the way those connections have been implemented is different.

There is no concatenation of channels. Each previous layer has been interpolated and then convoluted to have the same number of channels as the convolutional upsampling. Afterwards, the two layers are added and activated with a ReLU function.

The transposed convolutional upsamplings might "lose" information during the forward propagation. Interpolated upsamplings provide an additional insight during generation process. Adding convolutional and interpolated layers together results in improvement of outputs (as shown in Section 4.4).

In fact, without them experiments show that deep convolutional GAN with Wasserstein loss function, or linear then convolutional GAN with Wasserstein loss function doesn't produce meaningful results, shown in detail in Section 4.5.

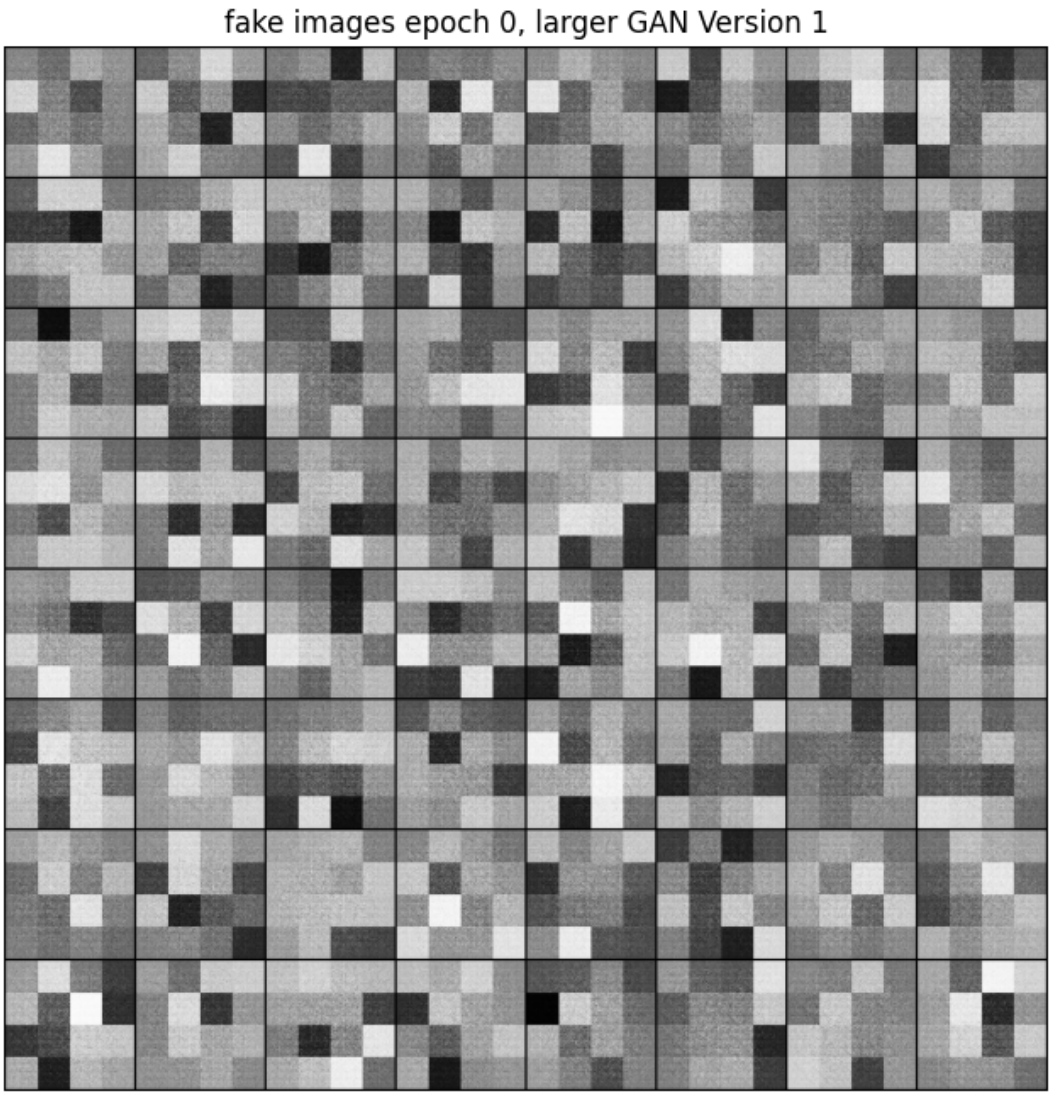

Figure 3.5: Generator output (before learning) for Larger GAN of Version 1 with the learnable kernel applied

3.2 Generator Architecture

3.2.1 Architecture for Small GAN

The architecture of this small (naming 'small' since it outputs grayscale 64×64 images) generative network is composed of 5 layers (Figure 3.7).

The first layer is a linear layer of size `batch_size`$\times 128$.

That layer is then reshaped into a `batch_size`$\times 1 \times 4 \times 4$ tensor.

In the tensor shape above, `batch_size` stands for batch size (set to 64 in our experiments), 1 for number of channels, 4×4 for width and height respectively (spatial dimensions).

After that consecutive convolutional layers follow, which upsample the spatial dimensions of image from a 4×4 matrix, to a 64×64 matrix (width and height).

Parallel to the convolutional layers, interpolated upsamplings have also been

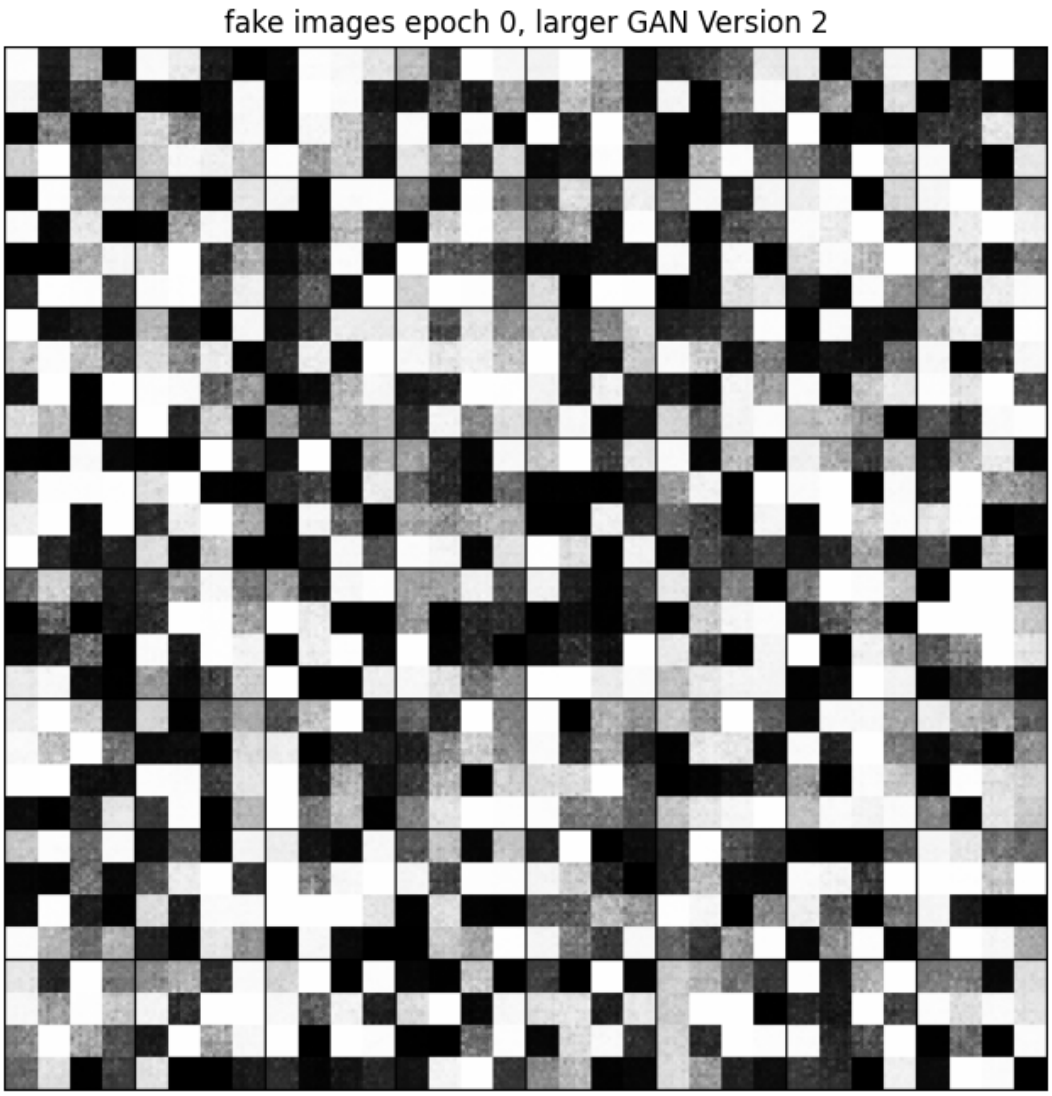

Figure 3.6: Generator output (before learning) for Larger GAN of Version 2 with the learnable kernel applied

calculated and added to already computed tensors.

For computing the error function gradient we use the chain rule of calculus, and those connections address the issue of "shattered" gradients after ReLU activations of sequential layers [BB23].

3.2.2 Architecture for Larger GAN, Version 1

The existing architecture for single-channel (grayscale) 64×64 images has been extended to single-channel 128×128 version. For this variant, the layers preceeding convolutional layers have been added twice, as shown in Figure 3.8.

3.2.3 Architecture for Larger GAN, Version 2

The existing architecture for single-channel 128×128 images has been modified with respect to its connections. For this variant, each convolutional layer i has been

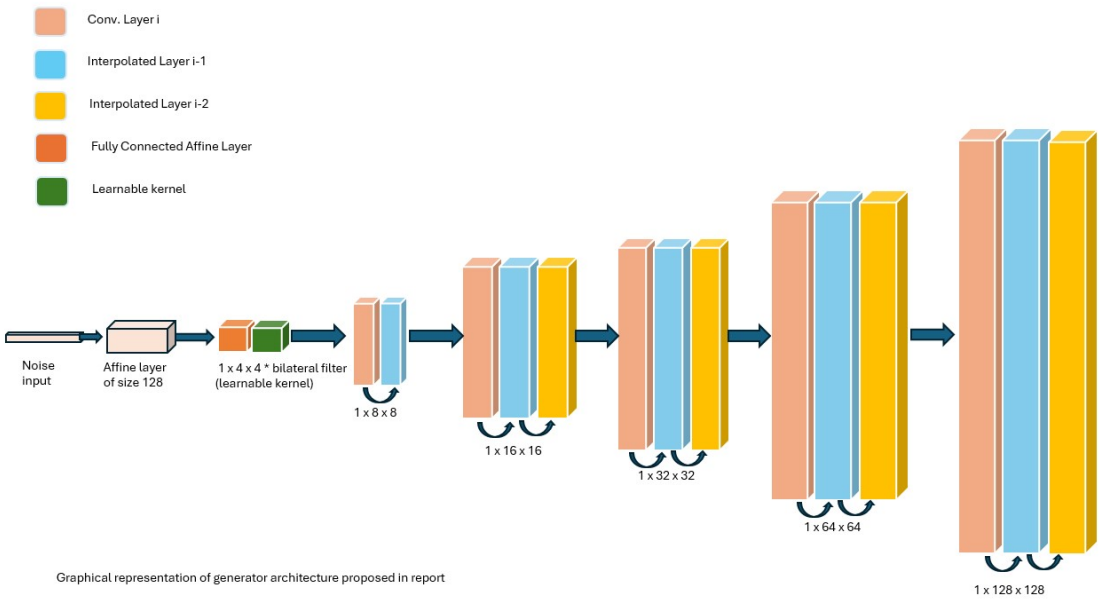

Figure 3.7: Visualization of Small GAN Generator Setup

added to layer $(i-1)$ and to layer $(i-2)$. Layers $(i-1)$, $(i-2)$ have been scaled through interpolation, layer $(i-1)$ with a scale factor of 2, layer $(i-2)$ with a scale factor of 4, as shown in Figure 3.9.

3.3 Image Post Processing

3.3.1 Edge Detection for Medical Analysis

Analysing the edges of generated images can be helpful from the point of evaluating the results.

A simple algorithm for viewing the edges has been applied and can be observed in Figure 3.10.

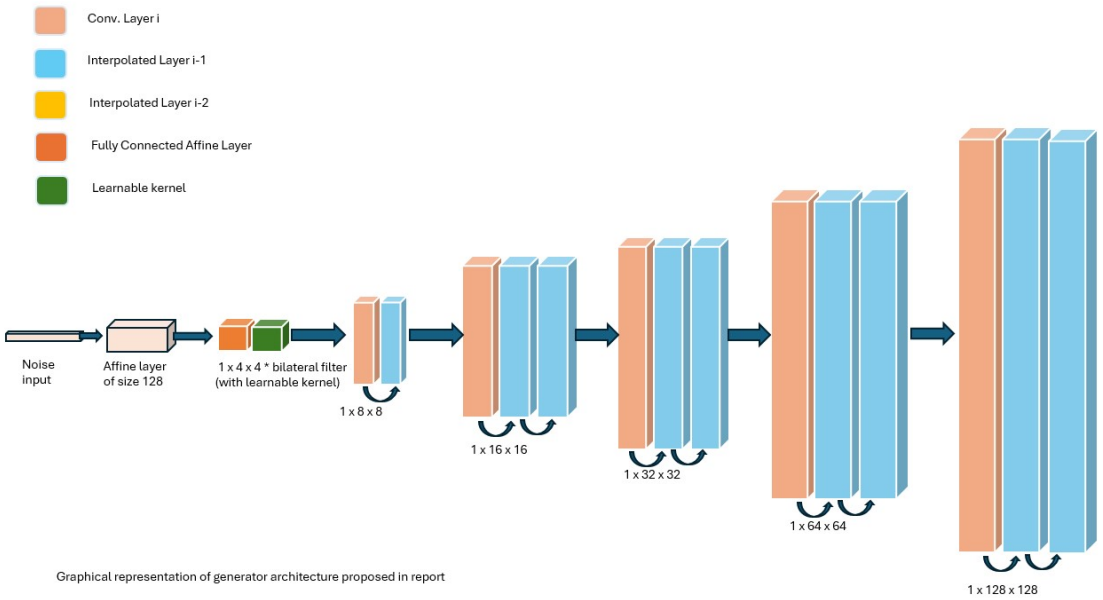

Figure 3.8: Visualization of first version of 128×128 GAN, which adds layer $i-1$ (if present) twice to layer i

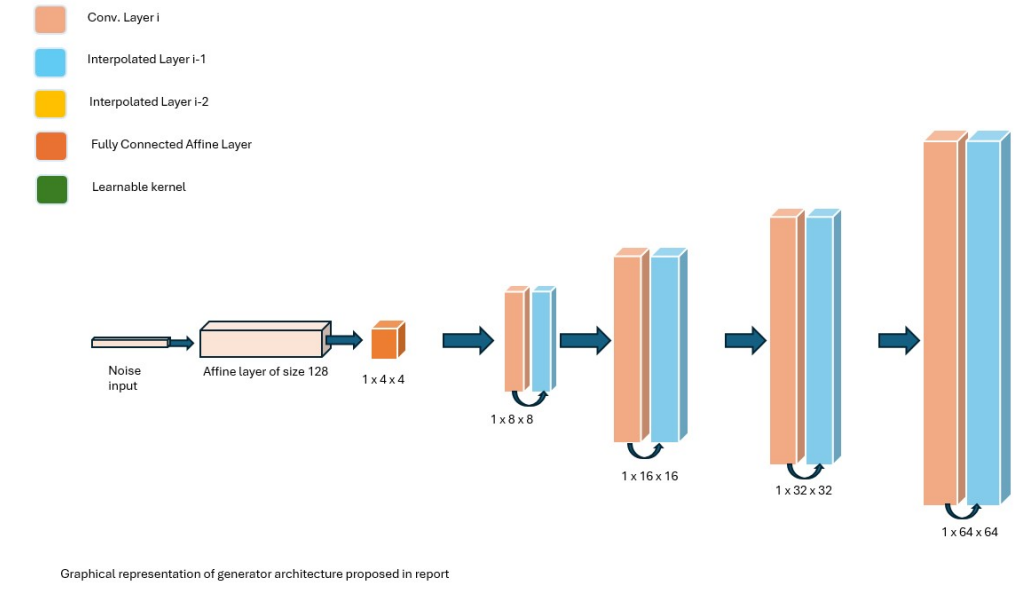

Figure 3.9: Visualization of second version of 128×128 GAN, which adds layers $i-1, i-2$ (if those are present) to layer i

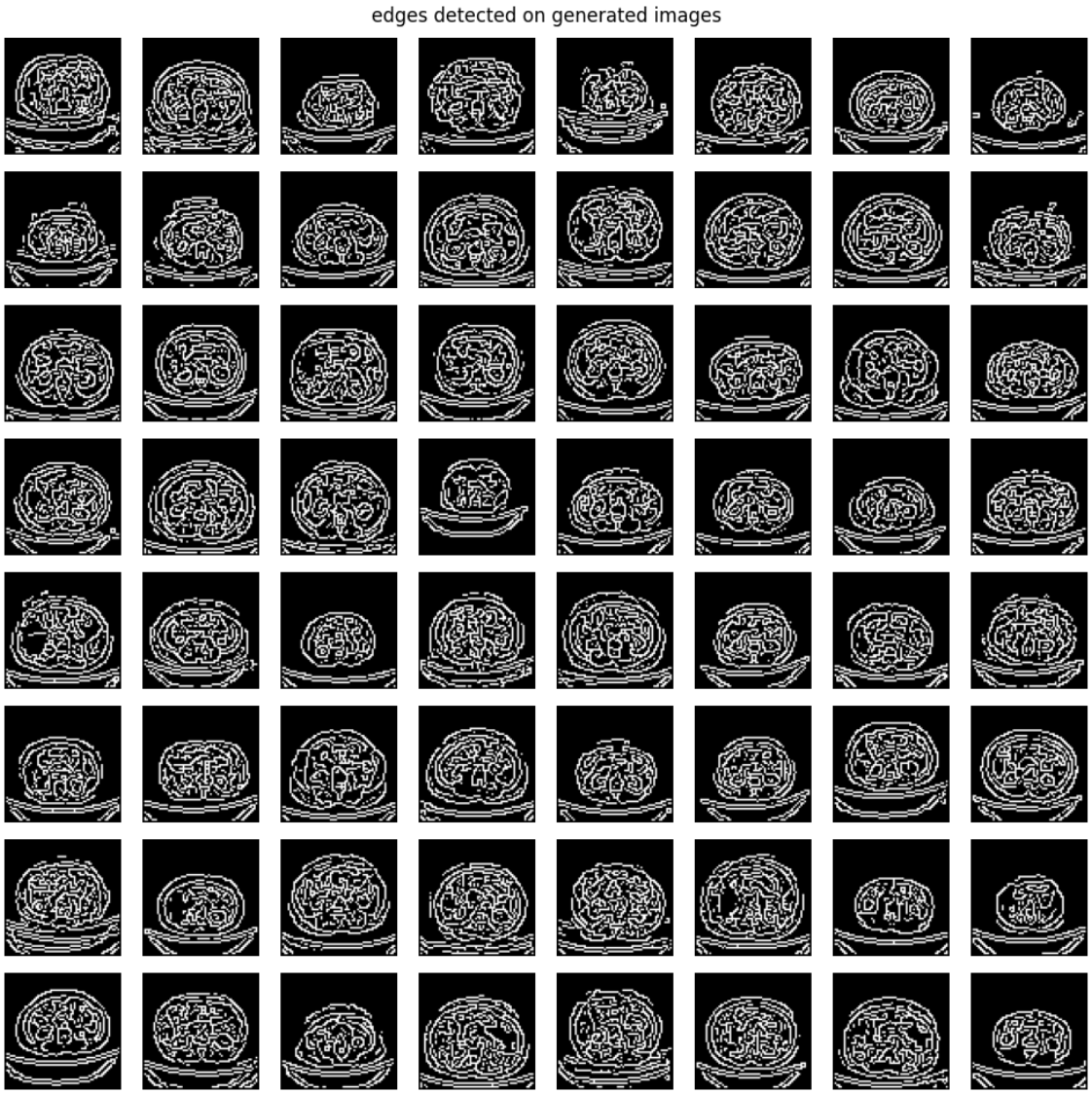

Figure 3.10: Edges found on the generated images

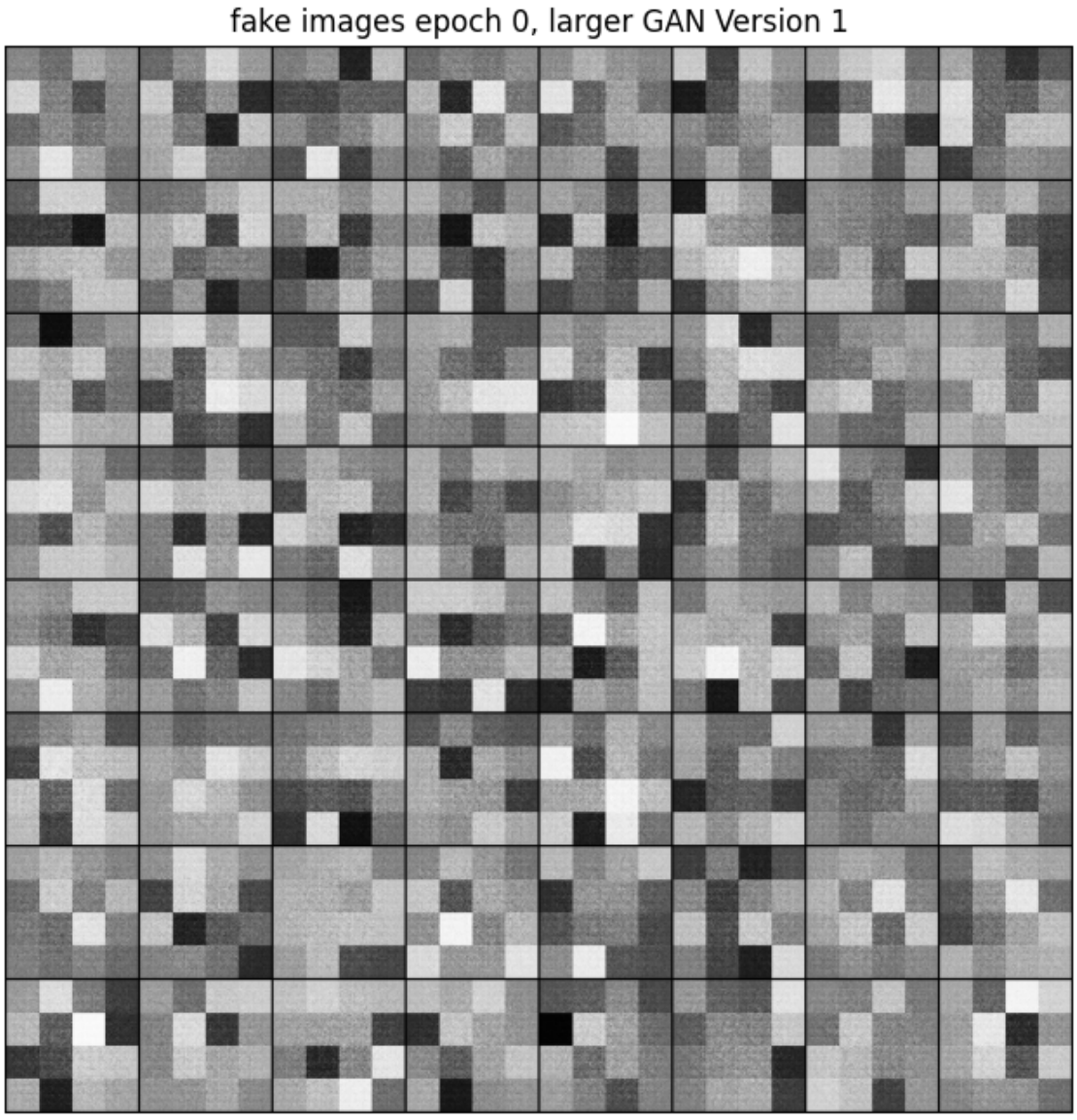

Figure 3.11: Learnable kernel's impact

Chapter 4

Evaluation and Results

4.1 Dataset

The dataset utilized for training is "CT KIDNEY DATASET: Normal-Cyst-Tumor and Stone" [IHH+22]. It was collected from PACS (Picture archiving and communication system) from Dhaka hospitals, in Bangladesh. Patients were diagnosed with tumors, cyst, stone findings and luckily for the patients, there were normal scans as well.

The dataset contains 12,446 unique data within it in which the cyst count is 3,709, normal 5,077, stone 1,377, and tumor 2,283.

4.2 Implementation Details

4.2.1 Framework

For implementation PyTorch framework has been utilized. Pytorch has powerful components, such as *torch*, which is like NumPy and has GPU support, *torch.autograd* - an automatic differentiation library supporting all differentiable tensor operations, *torch.nn* - a neural networks library, *torch.utils* - dataloader and other utilities' functional library.

4.2.2 General Description

Several code chunks, such as convolutional blocks, are in form of functions. `Generator` (a generator implementation) and `Critic` (a critic implementation) are defined as classes, and their instances are then created. Parametric variables' scope of definition is global, such as the input latent size, hidden layer size. Activation functions, layers are then initialized within the classes of generator and critic.

4.2.3 Dataset Loader

Pytorch provides a data loader, but the need for custom data loading object has risen (for loading .png images from folder, transforming them to tensors, normalising and resizing to specific size). The custom dataloader object loads images from folder and then transforms those images to tensors and provides them in batches. The transformer normalizes, as well as resizes all images' width and height to 64×64, 128×128 (depending on the scale of outputs and GAN variant).

4.2.4 Hardware Characteristics

The devices and their characteristics crucial for AI development are the following

- MSI GF63 Thin 9SCSR laptop–9th Gen. Intel® Core™ i7-9750H Processor, NVIDIA® GeForce® GTX 1650 Ti With Max-Q Design, 4GB GDDR6
- Local personal computer (home)–Intel® Core™ i5-9400F Processor, NVIDIA® GeForce® GTX 1050 Ti
- Google Colaboratory–NVIDIA Tesla K80 GPU with 12GB of VRAM (Video Random-Access Memory)

Google Colaboratory provides GPU Runtime with a memory of 12GB, which allows for larger in memory space tensor operations. This in turn, is critical for the development of larger models. But the free version takes longer time than the MSI laptop (4.2.4) for training per epoch, and after 500 epochs Google Colab. no longer allowed to continue the training, prompting message of limit of usage.

4.3 Numerical and Visual Tests

4.3.1 Frechet Inception Distance

Inception-v3 classifier has been used to extract features, also called embeddings. Afterwards, distance is calculated between the multivariate distributions with mean and variance of embeddings of generated samples, and real samples.

Frechet Inception Distance gives the measure of distance between two distributions. Graphically it can be shown as distance between the two curves.

Frechet Inception Distance is formulated mathematically as follows

$$FID = \|\mu_X - \mu_Y\|^2 + Tr(\Sigma_X + \Sigma_Y - \sqrt{\Sigma_X \Sigma_Y}) \tag{4.1}$$

After obtaining the features, those are then appended into a list. The resulting matrices are concatenated. Mean vector and covariance matrix are computed from

feature matrices, then those two parameters are input in multivariate normal distribution. Afterwards, some N amount of samples are taken from the multivariate normal distributions (from mean and covariance of fake and real embeddings).

4.3.2 FID Scores and Embedding Visualizations

FID and Inception Features for 64×64 Images

For analysing samples from multivariate normal distribution (from mean and covariance of fake embeddings of small GAN) principal component analysis has been applied to reduce 2048 dimensions to 3 (Figure 4.1), 2 (Figure 4.2) dimensions.

The two distributions' kernel density estimates (KDE) are plotted side by side in Figure 4.3 and separately in Figure 4.4, Figure 4.5.

The pairwise samples from multivariate normal distributions are selected and plotted in Figure 4.3. FID score after 7000 epochs of training for small gan (without Gaussian kernel added is 64, while with Gaussian kernel is 264, thus the benefits of adding it cannot be verified (visually those seem better))

FID and Inception Features for 128×128 Images, Version 1

For analysing samples of the multivariate normal from covariance matrix of embeddings (features) for grayscale 128 × 128 images (generated by larger GAN, Version 1) principal component analysis has been applied to reduce 2048 dimensions to 3 (Figure 4.1), 2 (Figure 4.2) dimensions.

The two distributions' kernel density estimates are plotted side by side in Figure 4.9 and separately in Figure 4.10, Figure 4.11.

The pairwise samples from multivariate normal distributions are selected and plotted in Figure 4.12

FID and Inception Features for 128×128 Images, Version 2

For analysing multivariate normal from covariance matrix of embeddings (features) for 128 by 128 images (generated by larger GAN, Version 2) principal component analysis has been applied to reduce 2048 dimensions to 3 (Figure 4.1), 2 (Figure 4.2) dimensions.

The two distributions' kernel density estimates are plotted side by side in Figure 4.15. The FID score after 11500 epochs of training with latent size 100 is 89.

The FID score after 9000 epochs of training with latent size 128 is 87 . Training with increased latent size has led to decrease in FID score (which is better).

4.3.3 Visual tests

The results have been shown to students, faculty members, professors. From the perspective of non-medical professional viewers the generated images are similar to original ones.

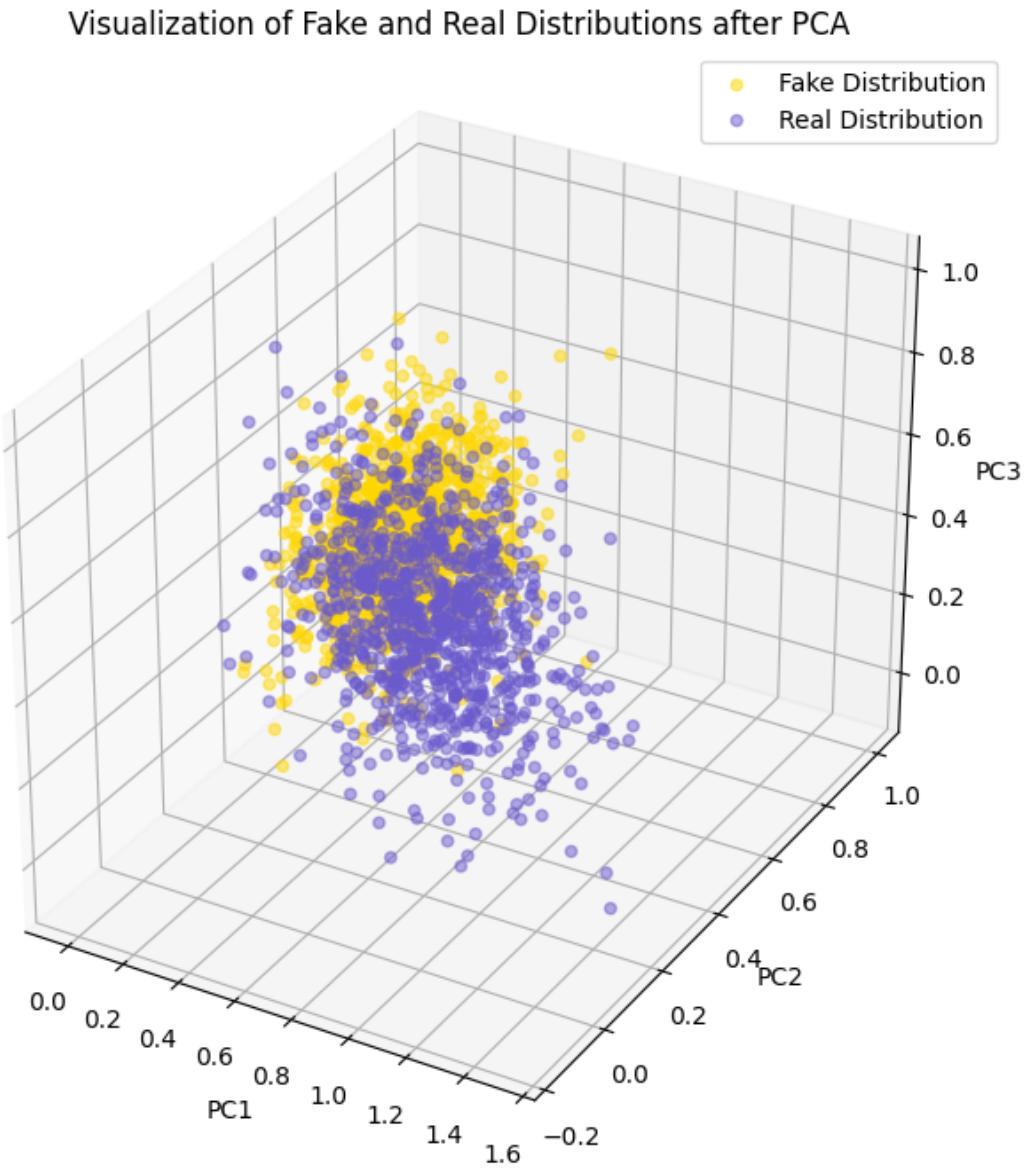

Figure 4.1: Principal component analysis has been applied to reduce 2048 dimensional covariance matrix to three dimensional space for visualization purposes

4.4 Results

4.4.1 Outputs of Larger GAN, Version 1

The Larger GAN's Version 1 has grayscale 128x128 as presented in Figure 4.16.

We leave visual evaluation to the reader (numerical scores have been provided in FID subsection).

4.4.2 Outputs of Larger GAN, Version 2

In Figure 4.17 the results of Larger GAN's Version 2 are stacked in grid.

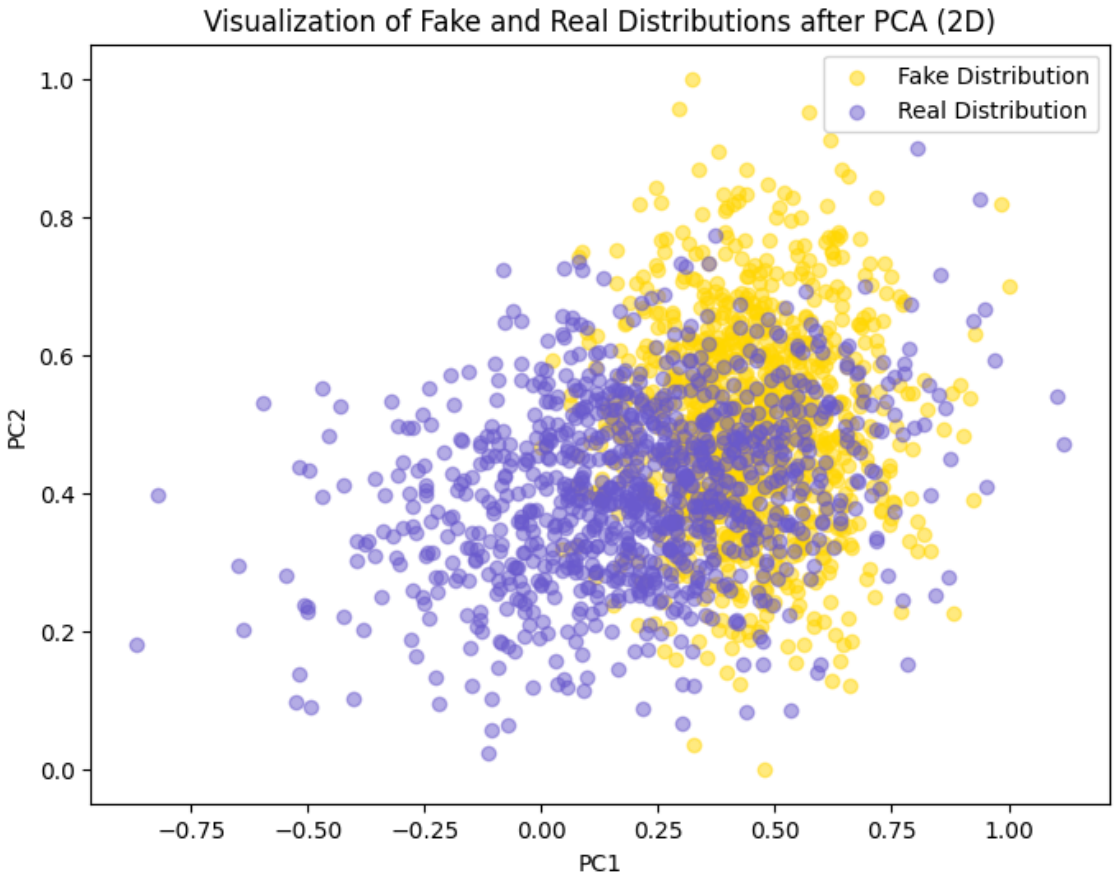

Figure 4.2: Principal component analysis has been applied to reduce 2048 dimensional covariance matrix to two dimensional space for visualization purposes

4.4.3 Outputs of Small GAN

In Figure 4.18 the generated images of small GAN (outputs grayscale images of size 64x64) are stacked in grid.

In Figure 4.19 images seem to be smoother, since the learnable kernel has been convoluted on the linear layer, but those images have higher FID score and numerically are "farther" from real images.

4.5 Comparative Analysis

In following subsections, GANs which output grayscale images of size 64×64 and of size 128×128 are compared in terms of their proximity to real images. For those GANs which have obviously mismatching outputs numerical scores are not provided.

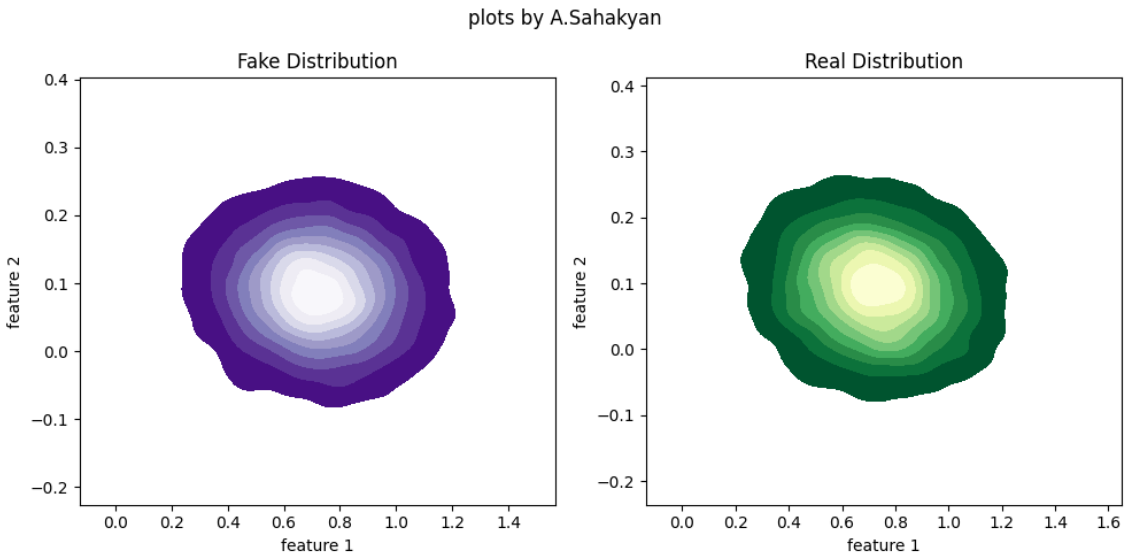

Figure 4.3:

4.5.1 Outputs of DCGAN

The DCGAN with classic implementation (for grayscale outputs of size 64×64) has been trained for 3000 epochs (batch size set to 64), but the issues such as noise artifacts in blank space (background), discontinuous lines persist (Figure 4.20).

DCGAN has been trained with `batch_size`=100 and the results were better with FID score of 149 (Figure 4.21). The discontinuous, redundant lines, noisy pixels (when smooth surfaces are needed) can still be observed and the aim of this thesis is to train with scarce data (improving generated images' fidelity and diversity with smaller batch size).

For 1500 epochs of training DCGAN has FID score of 149 while the proposed small GAN architecture has FID score of 64 .

4.5.2 Outputs of WGAN with Convolutional Layers

The WGAN implementation very similar to DCGAN in previous section, has been trained with Wasserstein loss.

Visually, the results are not similar to training images, so the training with this setup was interrupted (Figure 4.22) after 2500 epochs.

4.5.3 Outputs of WGAN with Linear and Convolutional Layers

As in previous subsection, visually (Figure 4.23), the results are not similar to training images, so the training with this setup was interrupted.

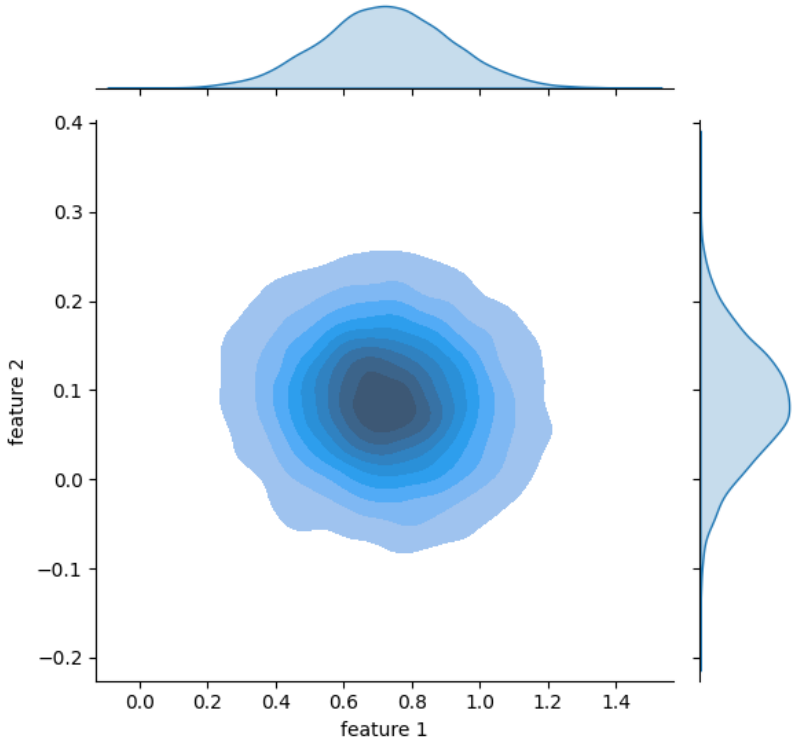

Figure 4.4:

4.5.4 Outputs of Presented Small and Larger GANs

In comparison with the outputs of DCGAN, the proposed methods have lower FID score and don't have issues with edges and lines.

Compared to GANs without skip connections and Gaussian kernel usage, the proposed GANs output images visually similar to training samples.

For the same number of epochs (9000) of training larger GAN Version 1 has FID score of 97 while larger GAN Version 2 has FID score of 90 with `latent_size`=100 and FID score of 87 with `latent_size`=128.

4.5.5 Loss plots

To showcase the training stability of GANs with Wasserstein loss, below visualization of loss functions during the first 1000 epochs of training are provided. During DCGAN training, discriminator's loss function has many spikes and overall the training is not stable (Figure 4.24, Figure 4.25), while GANs with Wasserstein loss have a more 'stable' critic-Figure 4.28, 4.29. But the training stability alone is not a 'guarantee' that the GAN will output close to real results-Figure 4.26.

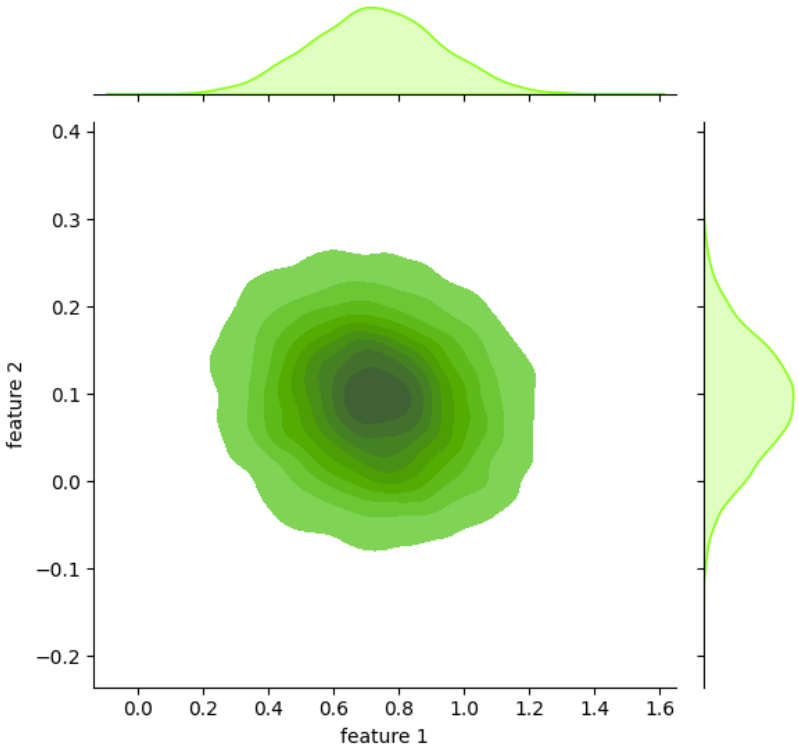

Figure 4.5: Features of samples kernel density estimation (KDE) plot (64×64 images)

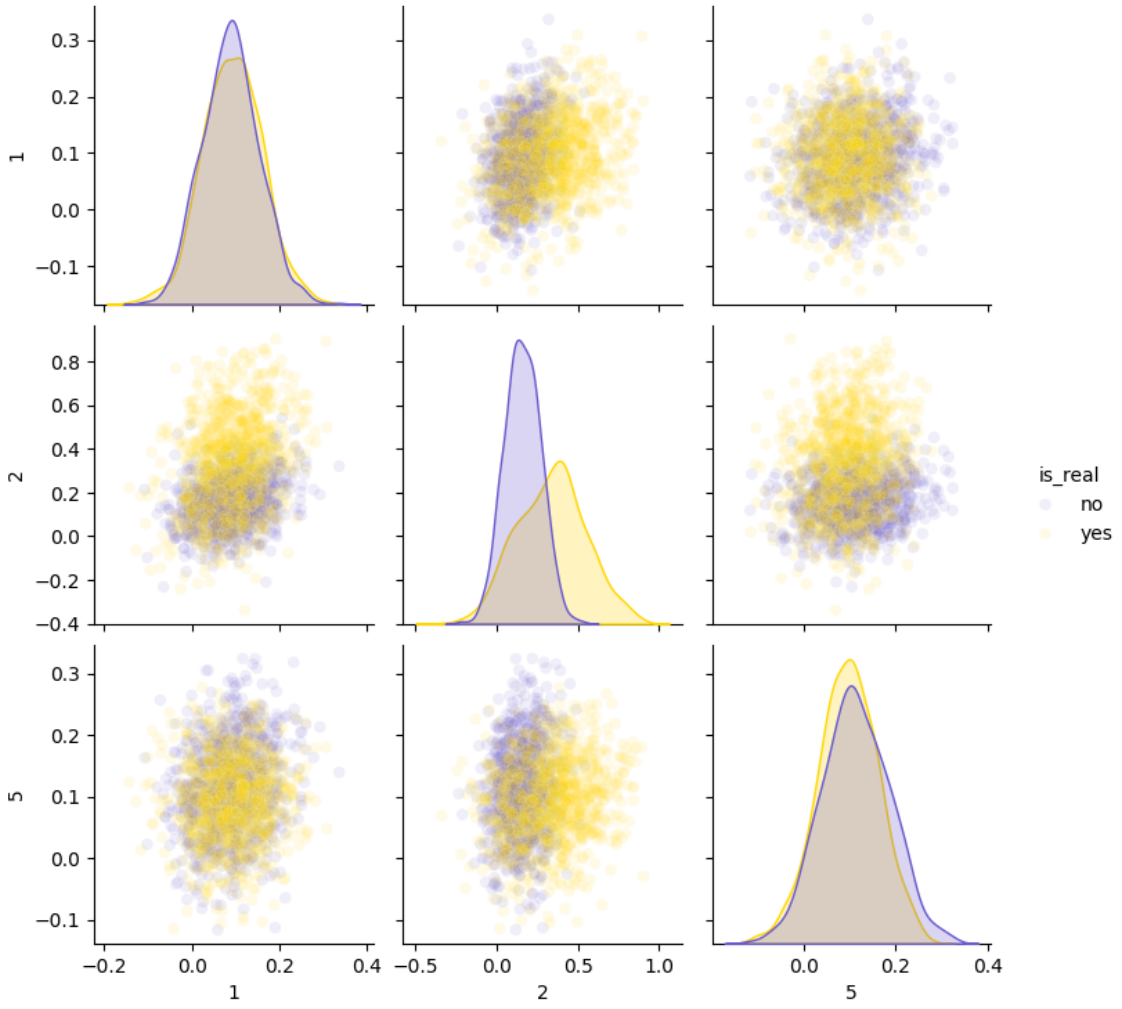

Figure 4.6: Features of samples from the real and the fake distributions density estimation plot by pairs (64×64 images)

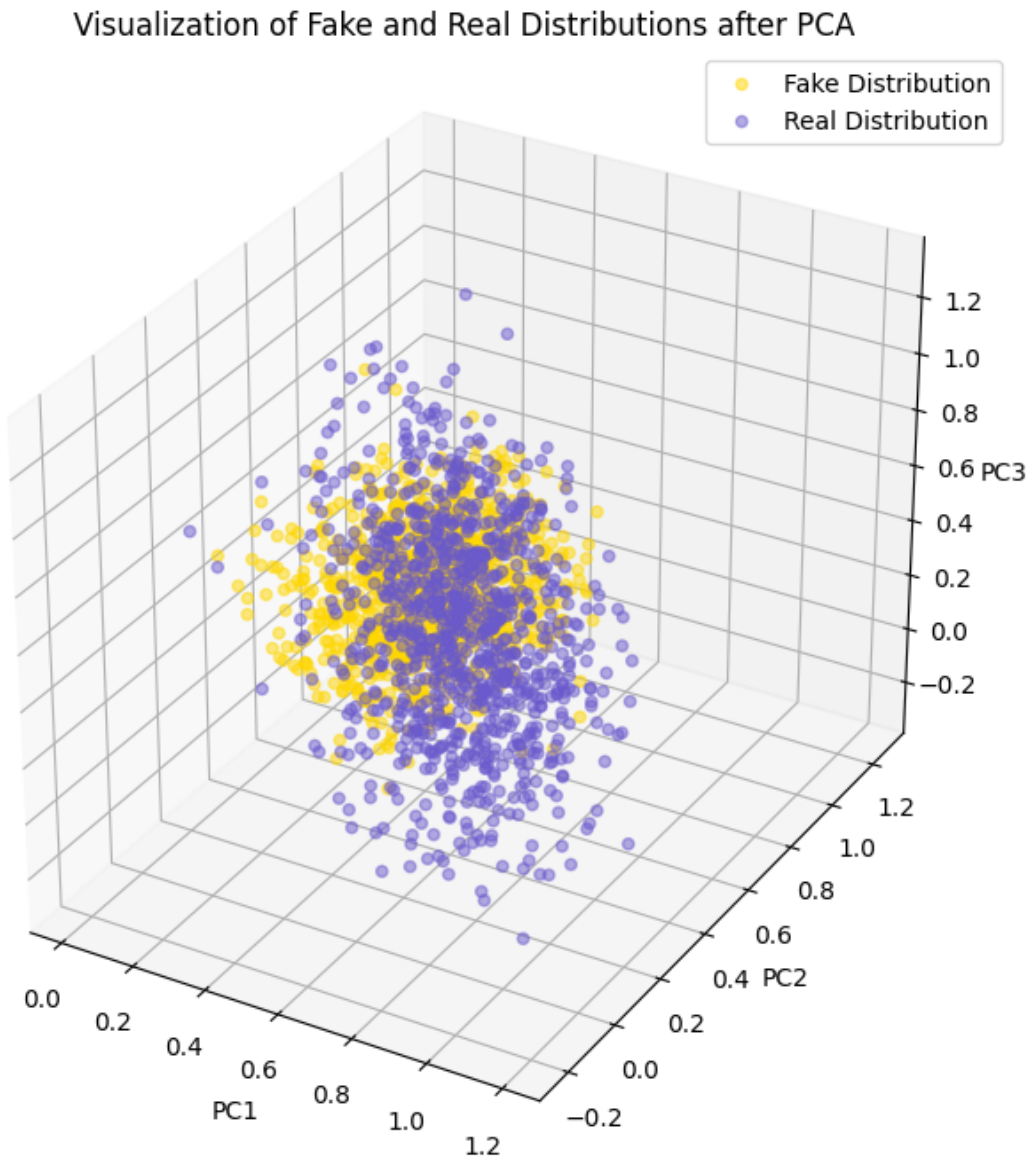

Figure 4.7: Principal component analysis has been applied to reduce 2048 dimensional multivariate normal to three dimensional space for visualization purposes (derived from 128×128 fake (larger GAN V1) and real images)

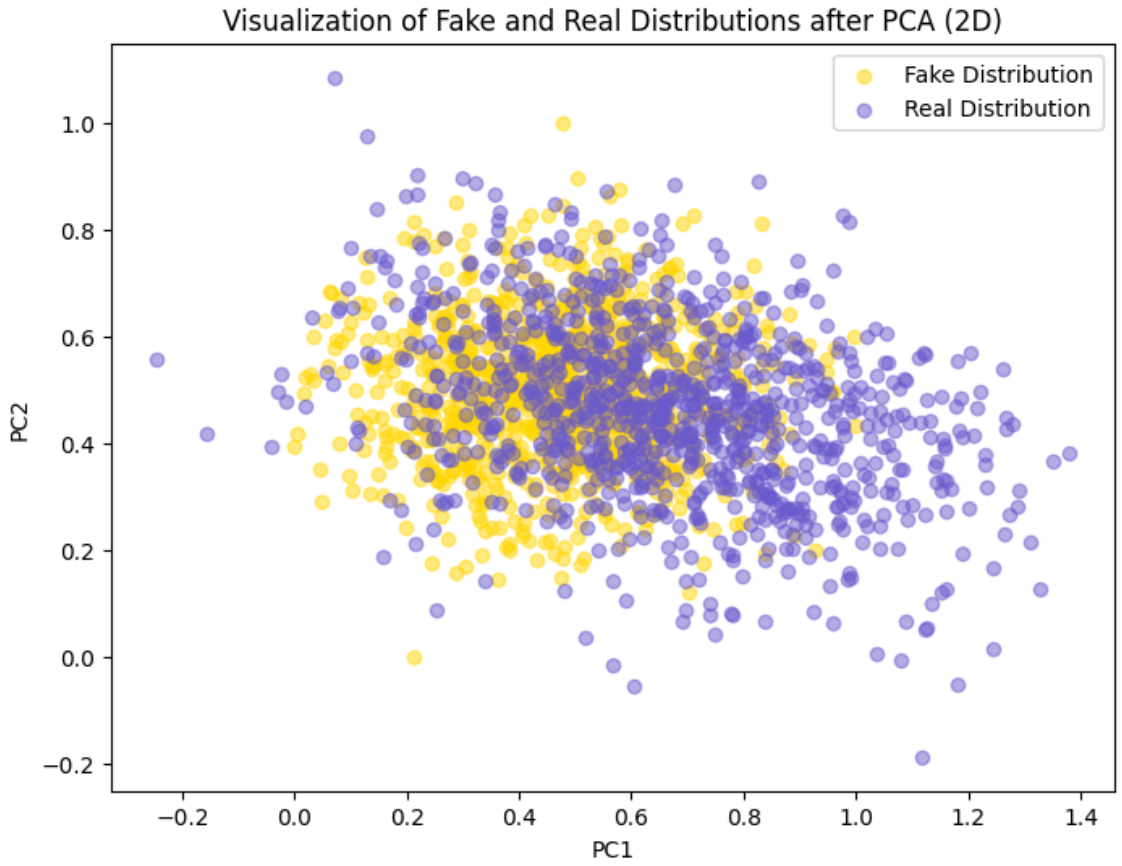

Figure 4.8: Principal component analysis has been applied to reduce 2048 dimensional multivariate normal to two dimensional space for visualization purposes (derived from 128×128 fake (larger GAN V2) and real images)

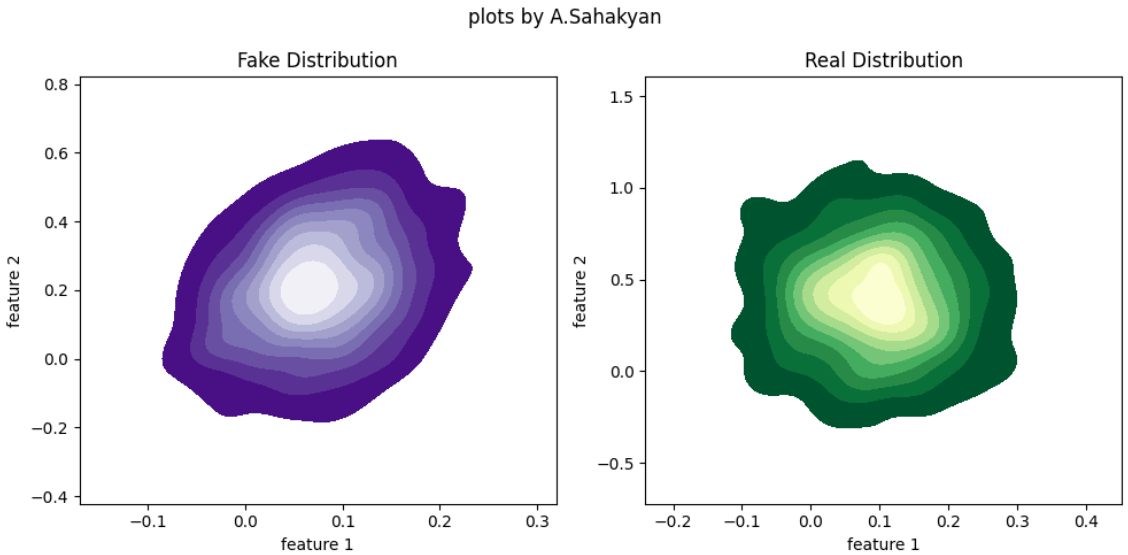

Figure 4.9: Principal component analysis has been applied to reduce 2048 dimensional multivariate normal to three dimensional space for visualization purposes (larger GAN V1, features of samples KDE)

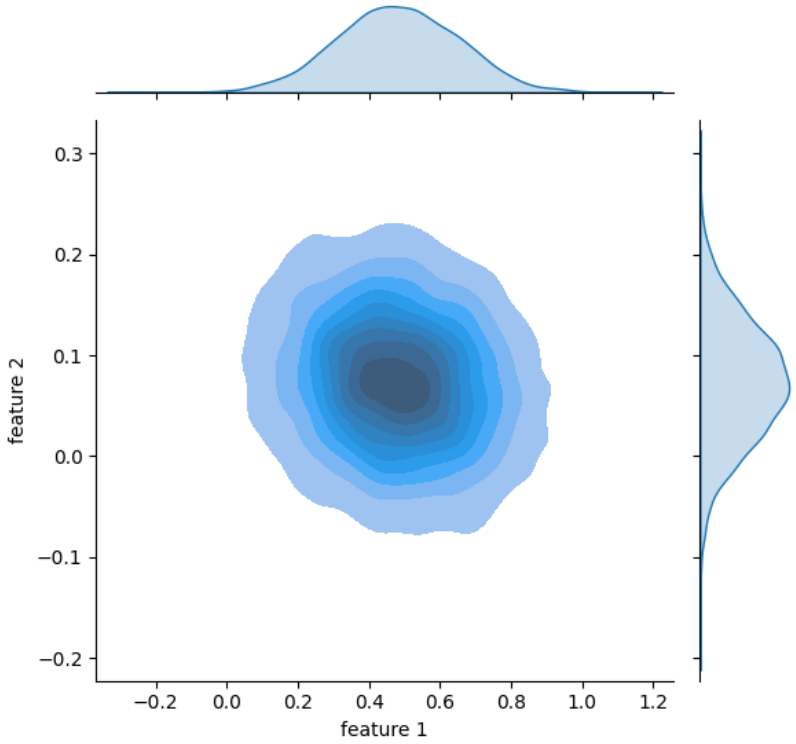

Figure 4.10: Features of samples from the fake multivariate normal KDE plot (128 × 128 images)

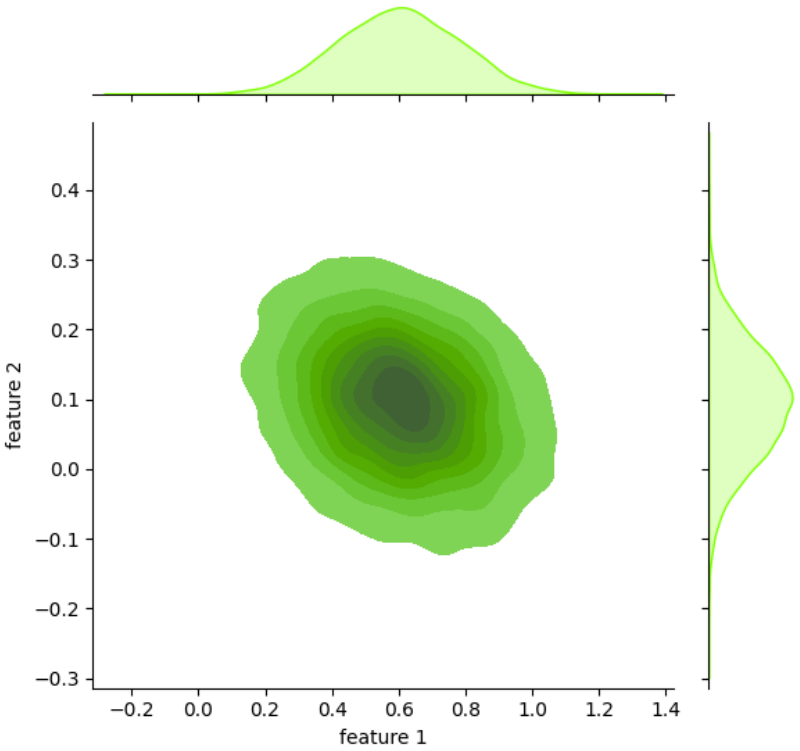

Figure 4.11: Features of samples from the real multivariate normal KDE plot (128 × 128 images)

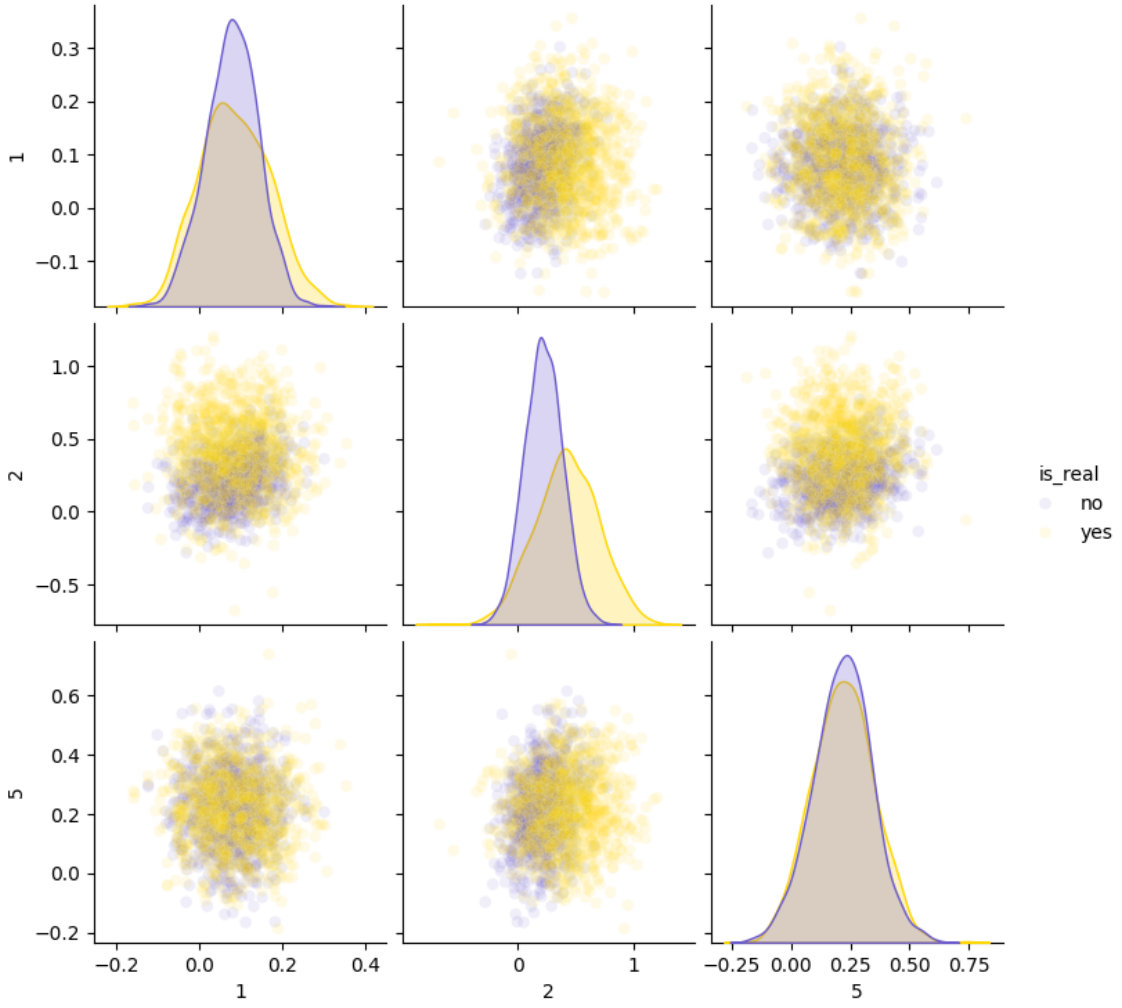

Figure 4.12: Pairwise kernel density estimates from samples of multivariate normal distribution

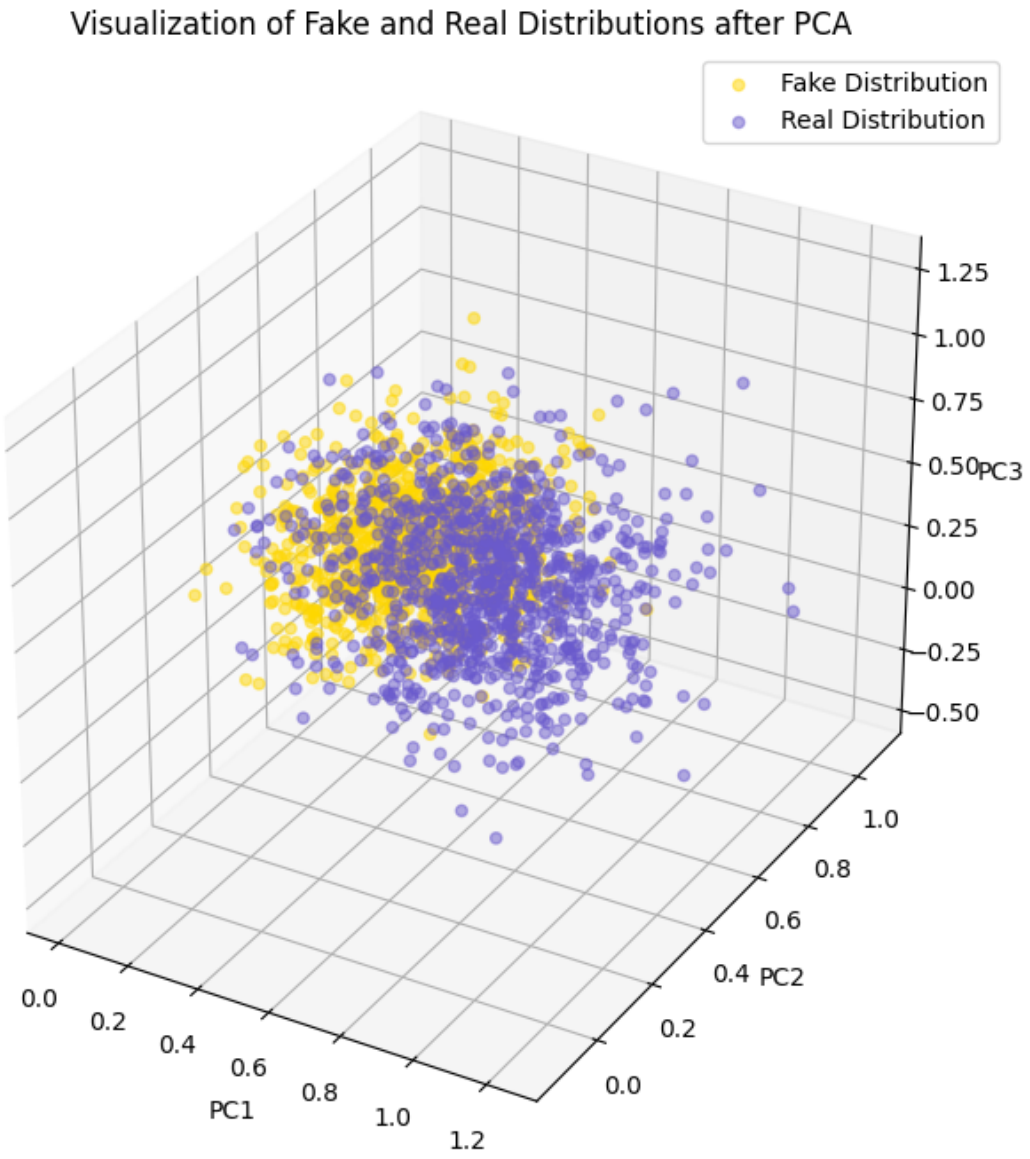

Figure 4.13: Principal component analysis has been applied to reduce 2048 dimensional multivariate normal to three dimensional space for visualization purposes

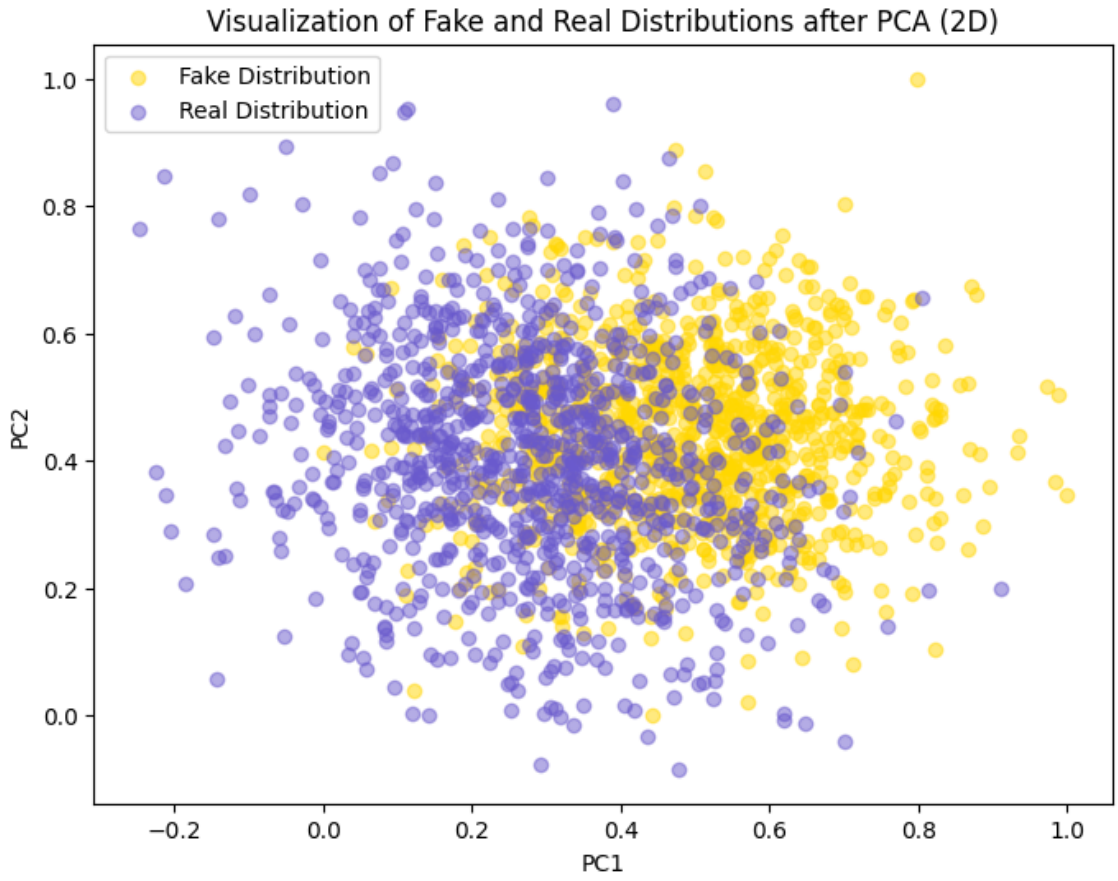

Figure 4.14: Principal component analysis has been applied to reduce 2048 dimensional multivariate normal to two dimensional space for visualization purposes

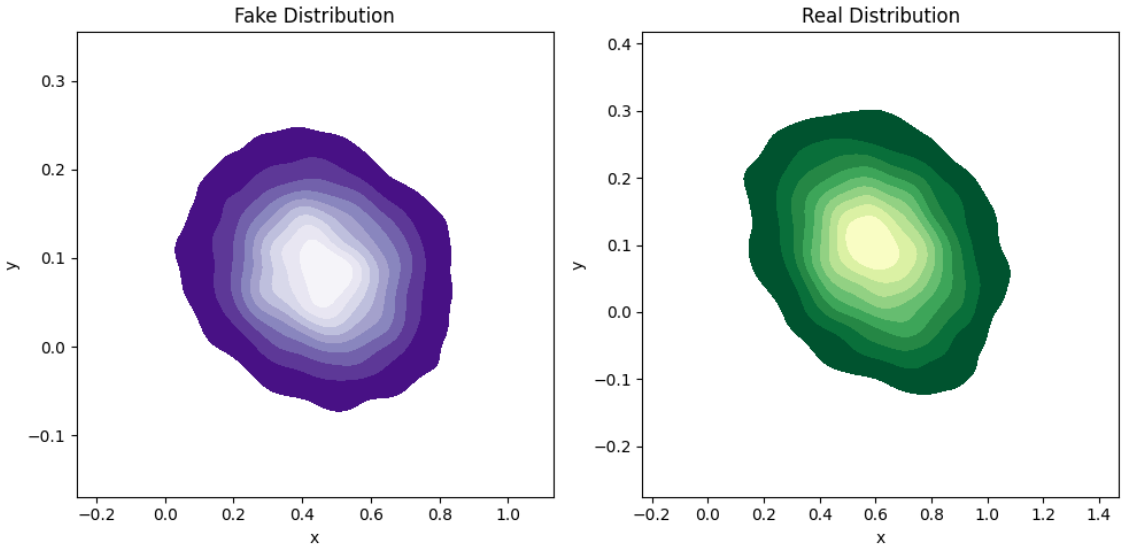

Figure 4.15: Principal component analysis has been applied to reduce 2048 dimensional multivariate normal to three dimensional space for visualization purposes

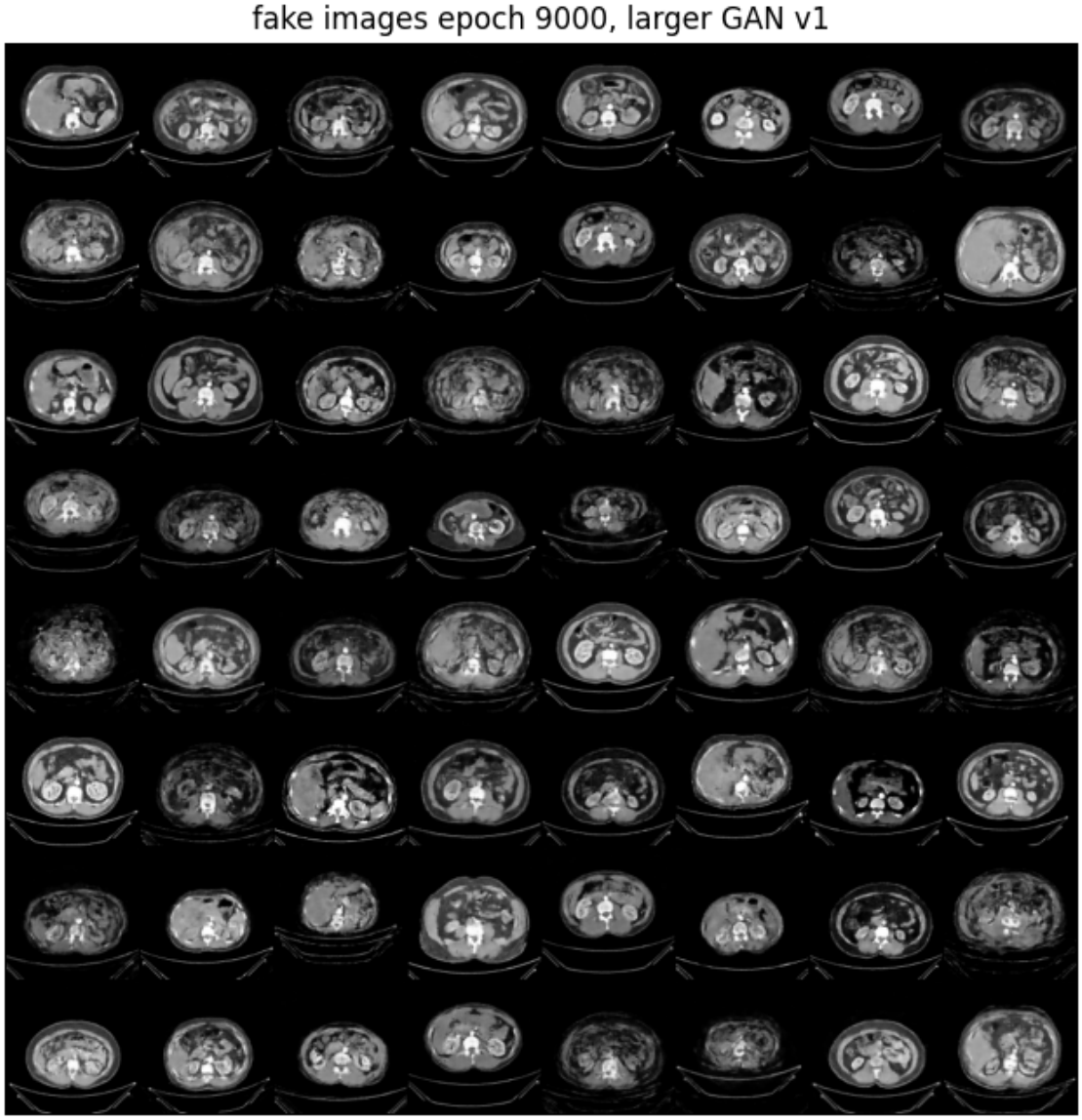

Figure 4.16: Output of larger GAN Version 1 after 9000 epochs of training

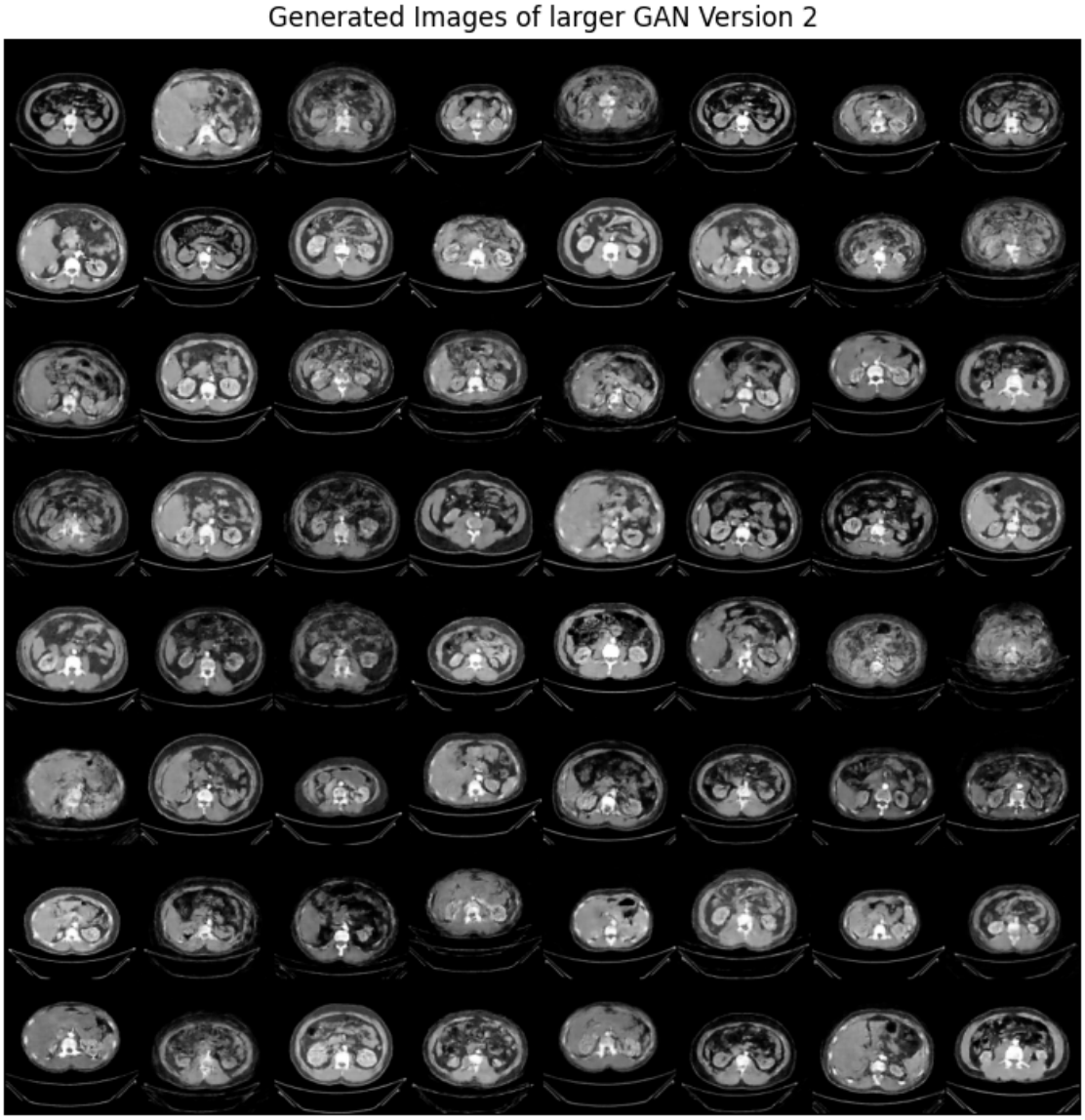

Figure 4.17: Output of larger GAN Version 2 after 11500 epochs of training

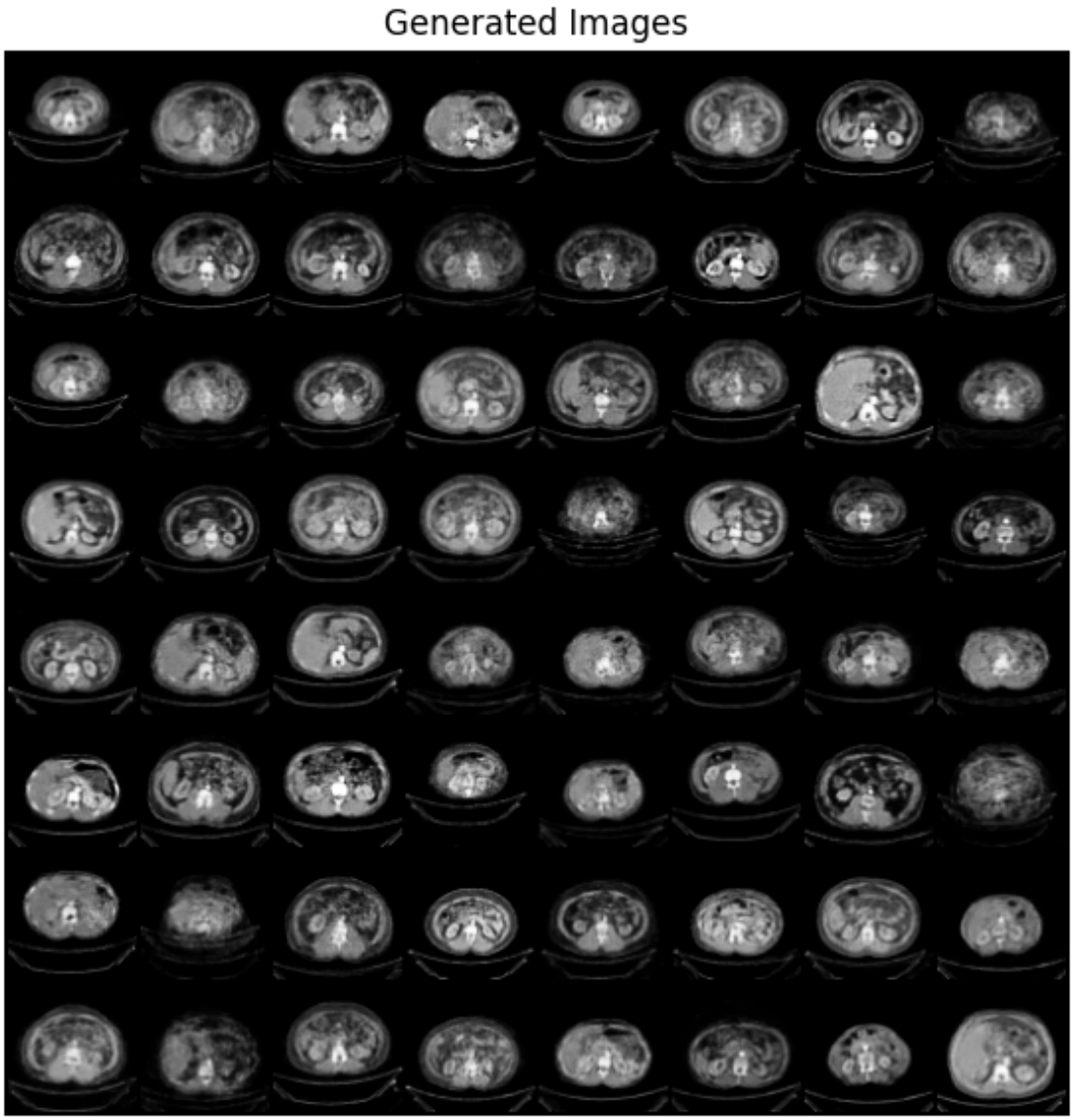

Figure 4.18: Small GAN (without the learnable kernel) outputs has a low FID score of 64 after 7000 epochs of training

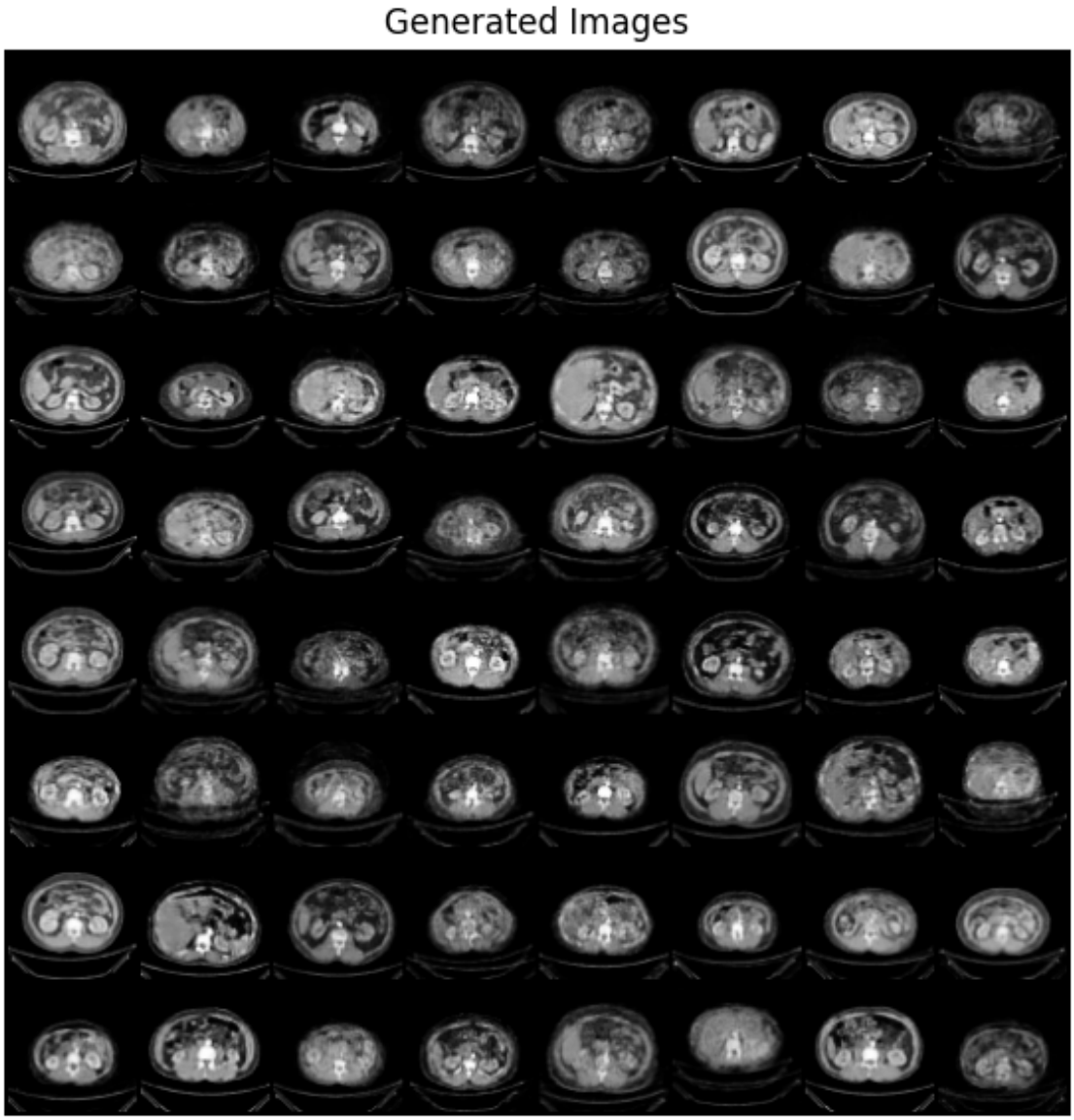

Figure 4.19: For small GAN, the learnable kernel has been applied here, visually it seems normal, but FID score is high (262)

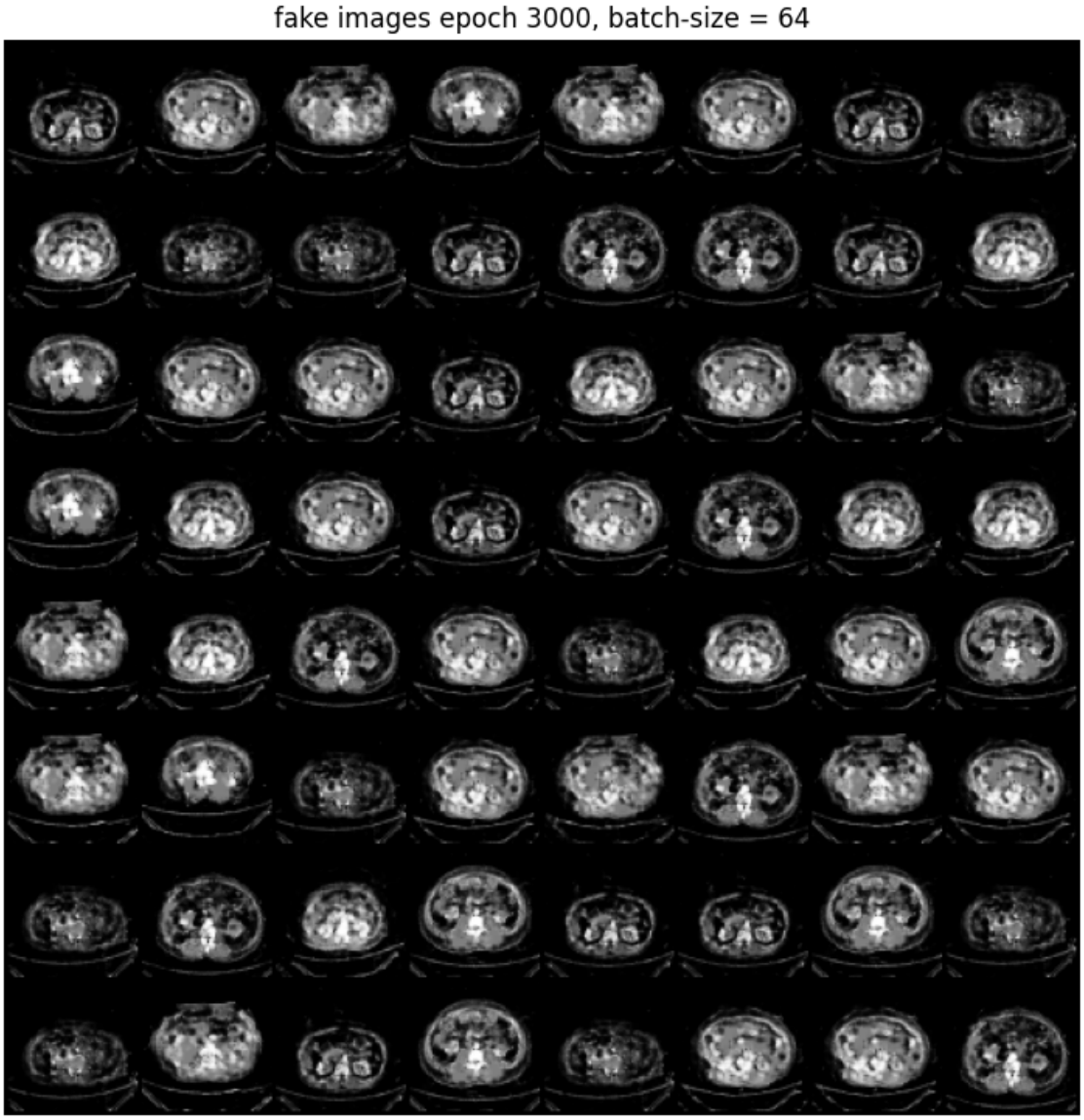

Figure 4.20: DCGAN with batch size 64 has been trained for 3000 epochs (for grayscale $64x64$ outputs)

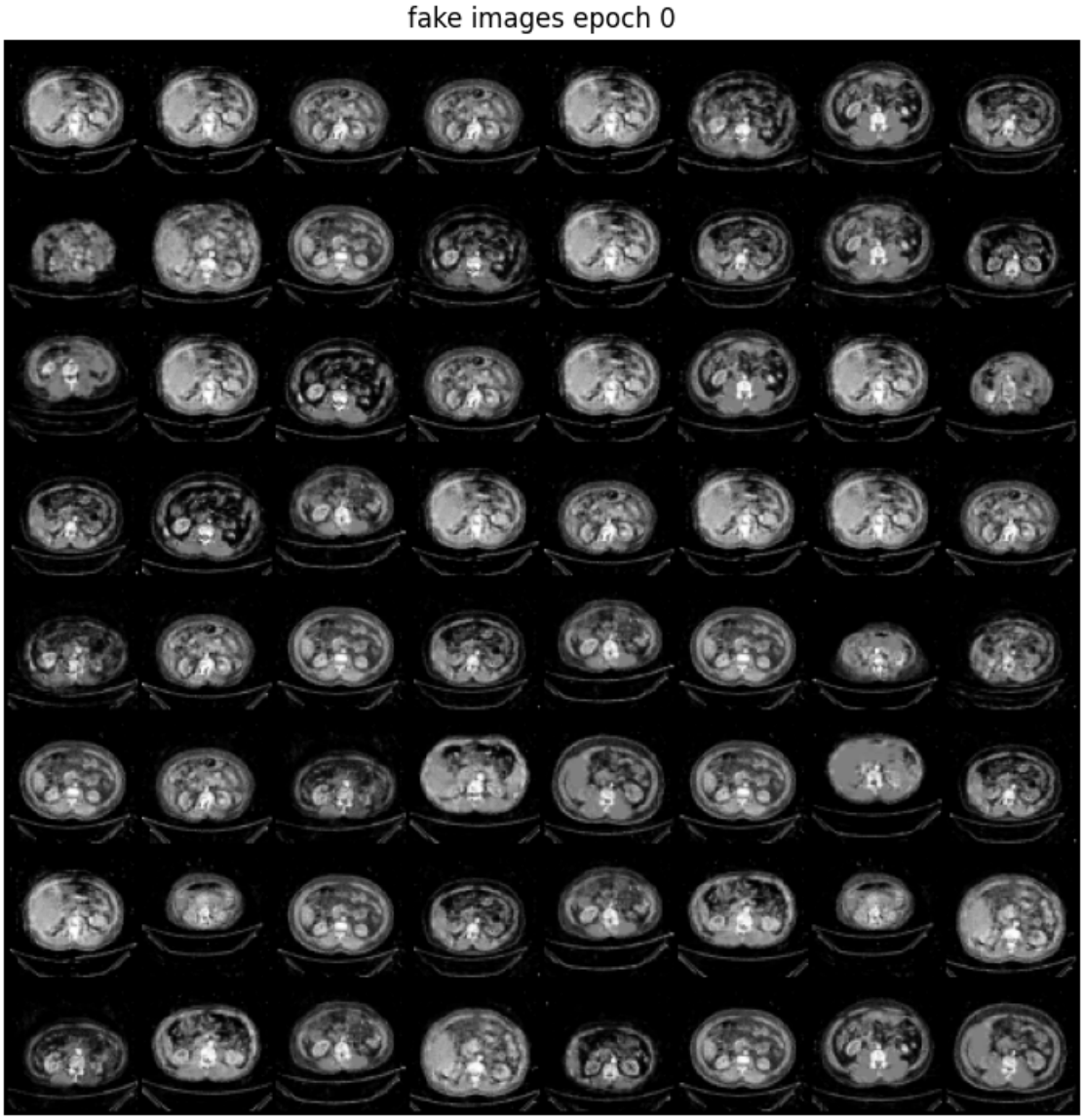

Figure 4.21: DCGAN with batch size 100 has been trained for 1500 epochs (for grayscale $64x64$ outputs) and produces better results, with lower FID score (149)

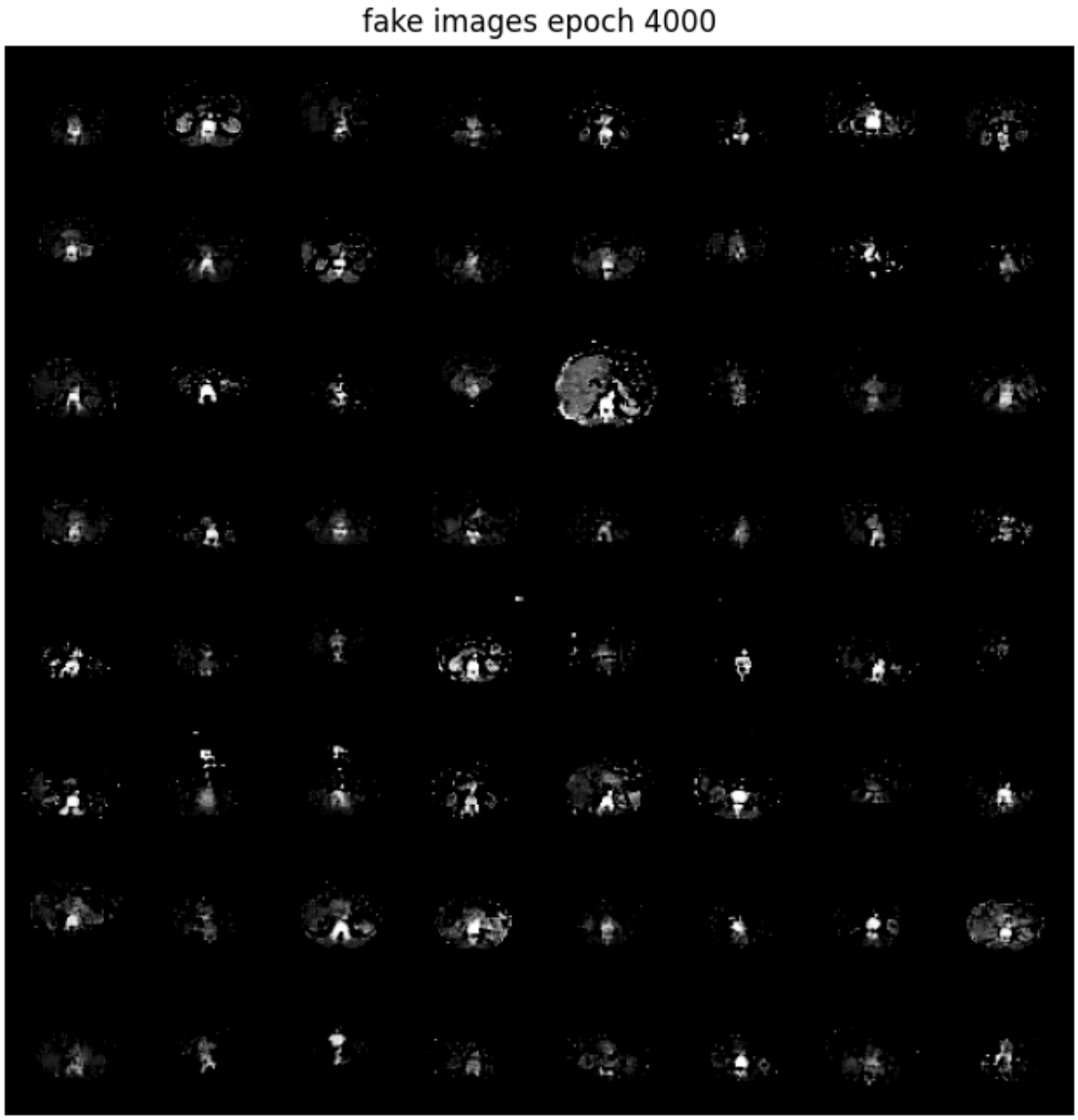

Figure 4.22: WGAN with convolutional layers, without any modifications, failed to produce ‘normal’ images

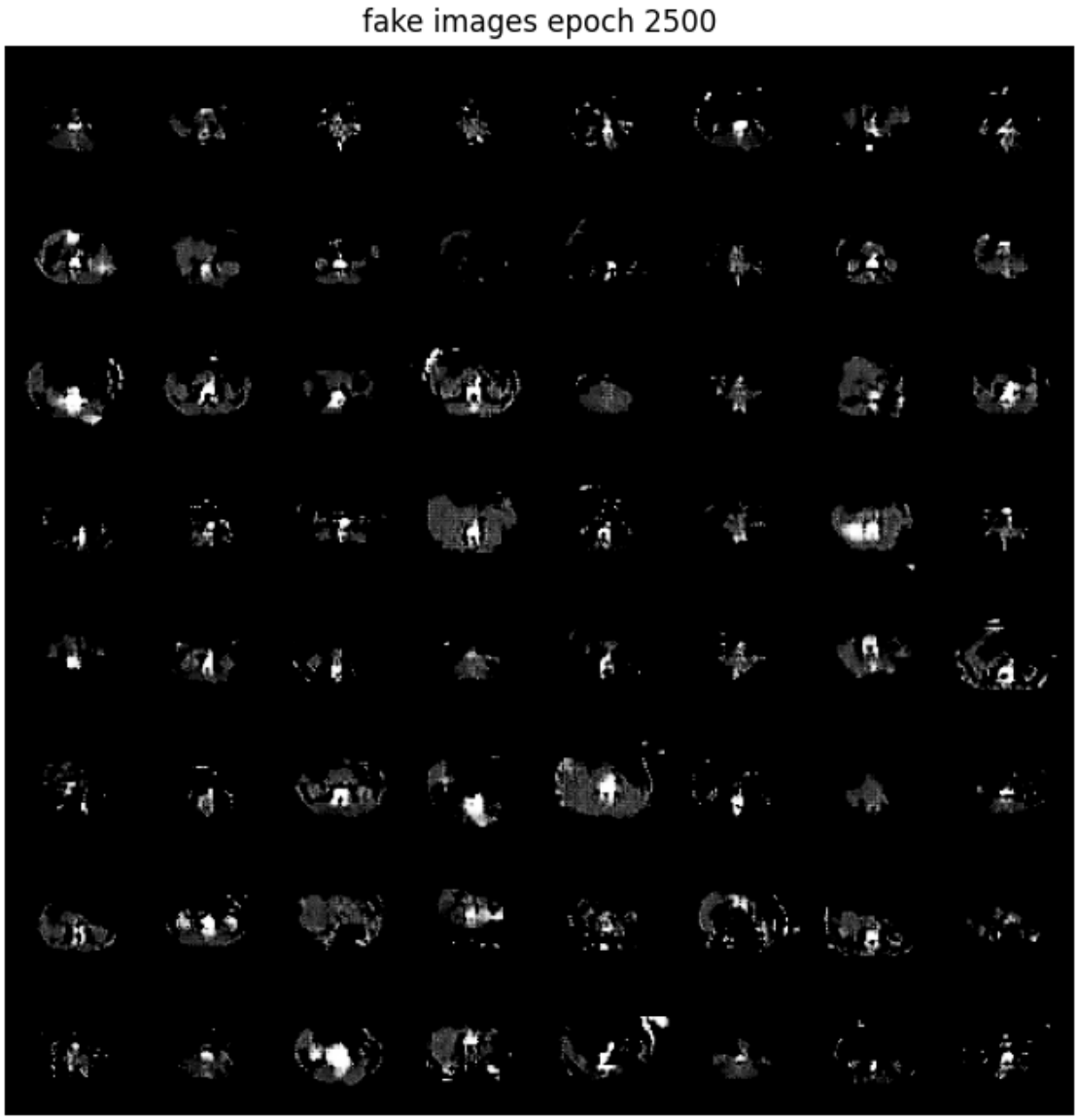

Figure 4.23: WGAN with linear and convolutional layers, without any modifications, failed to produce 'normal' images

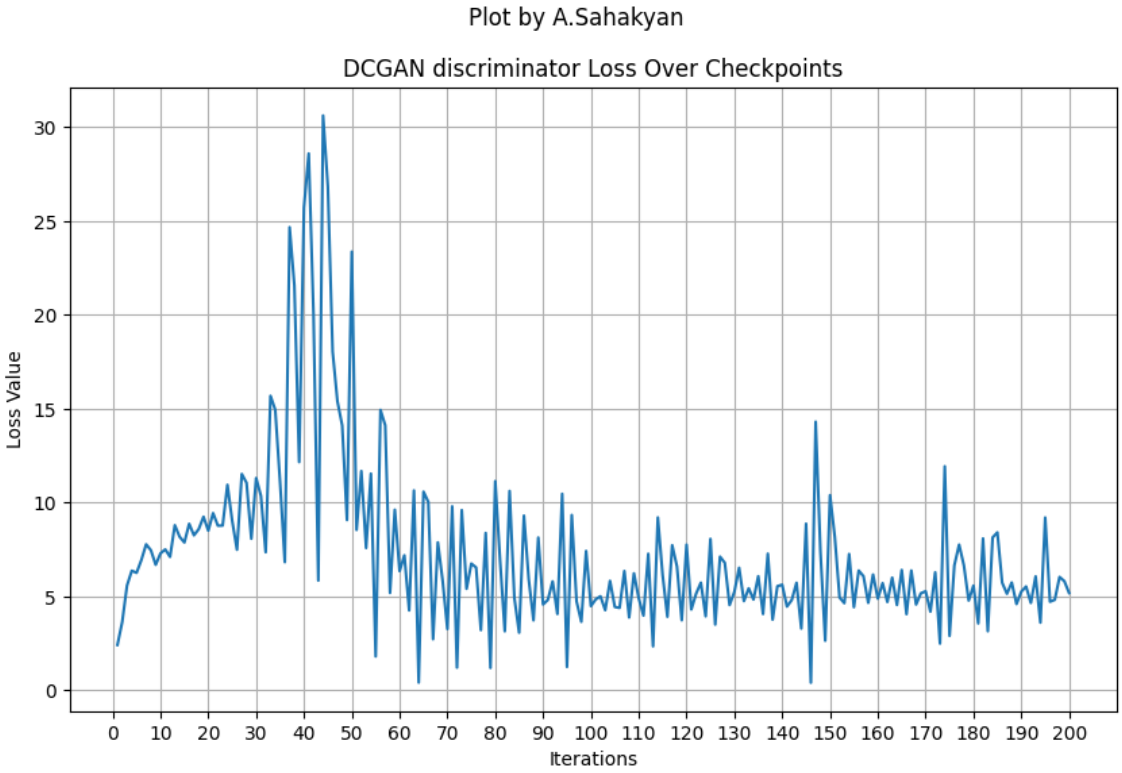

Figure 4.24: DCGAN discriminator losses are not stable

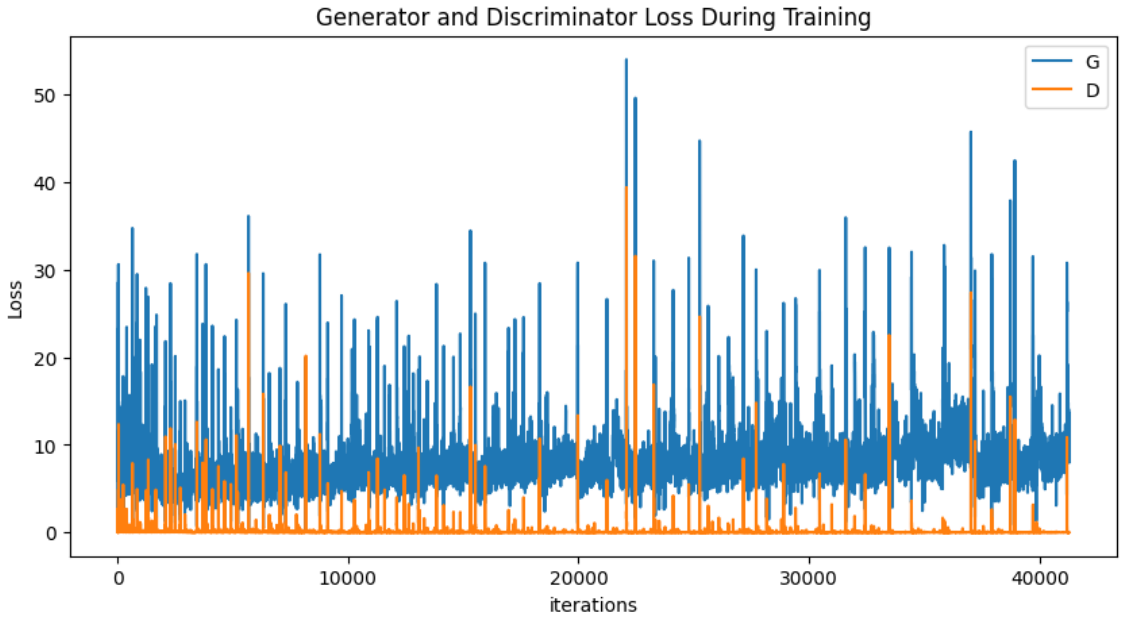

Figure 4.25: DCGAN discriminator losses are not stable

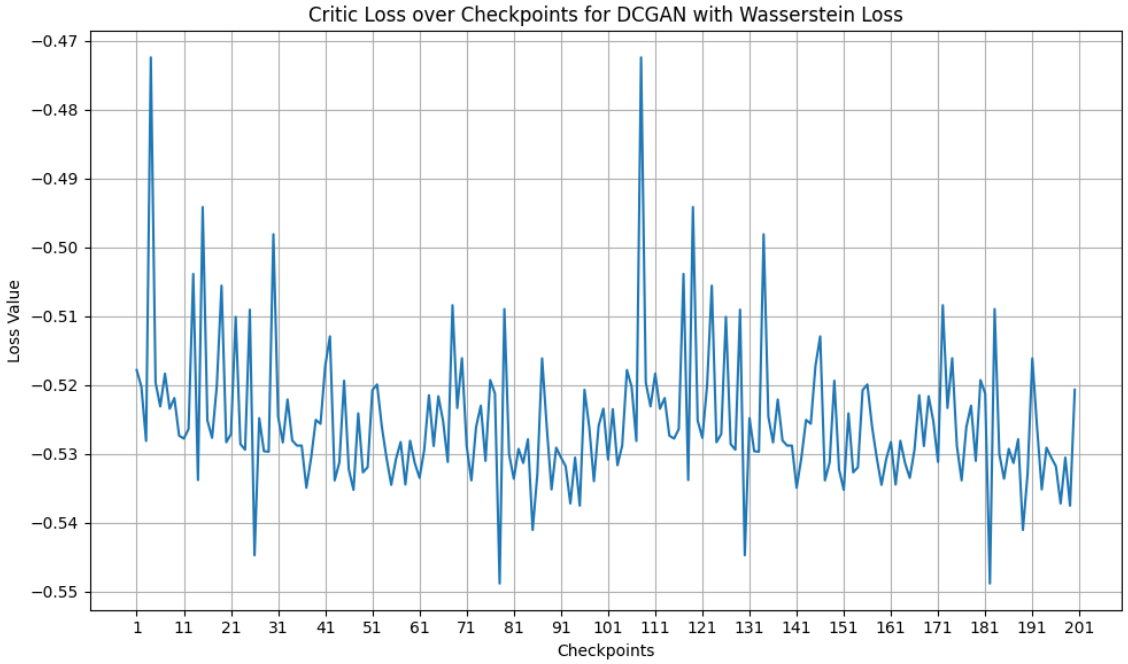

Figure 4.26: The training stability alone is not a sufficient condition for generating close to real images

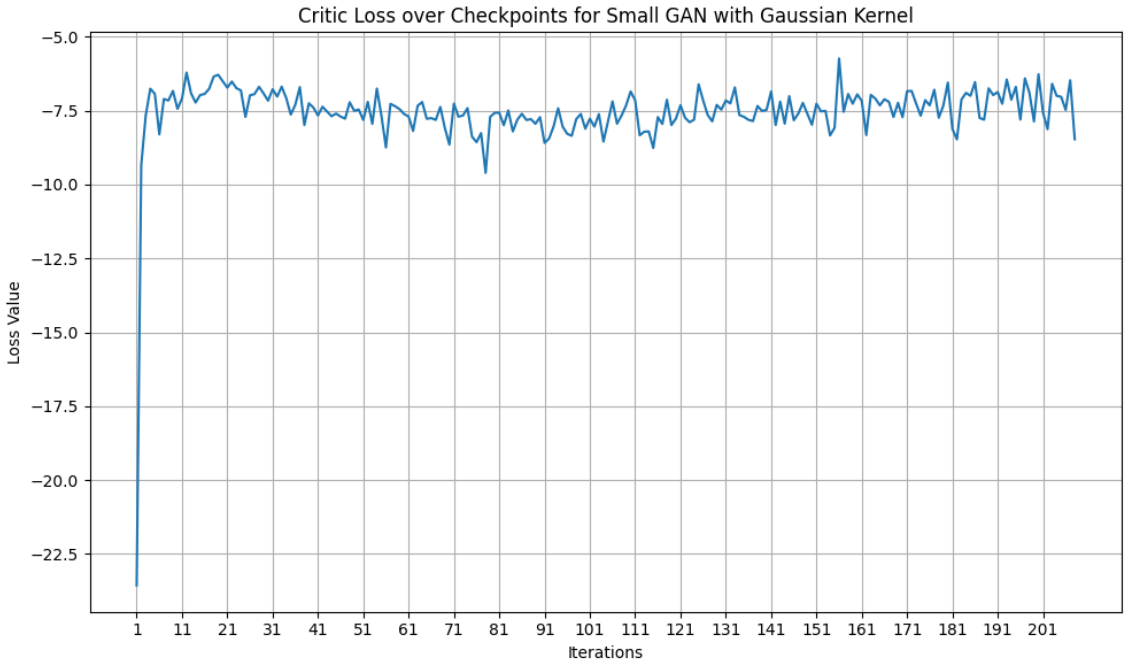

Figure 4.27: Small GAN (with proposed additions) had a stable training

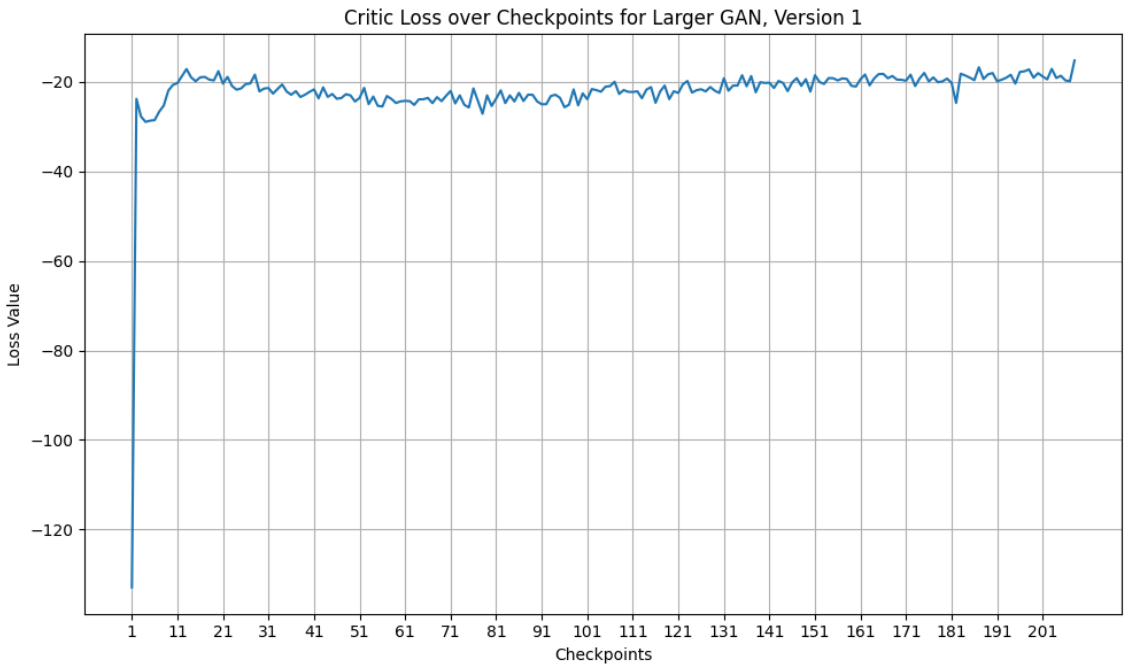

Figure 4.28: Larger GAN, Version 1 had a stable training

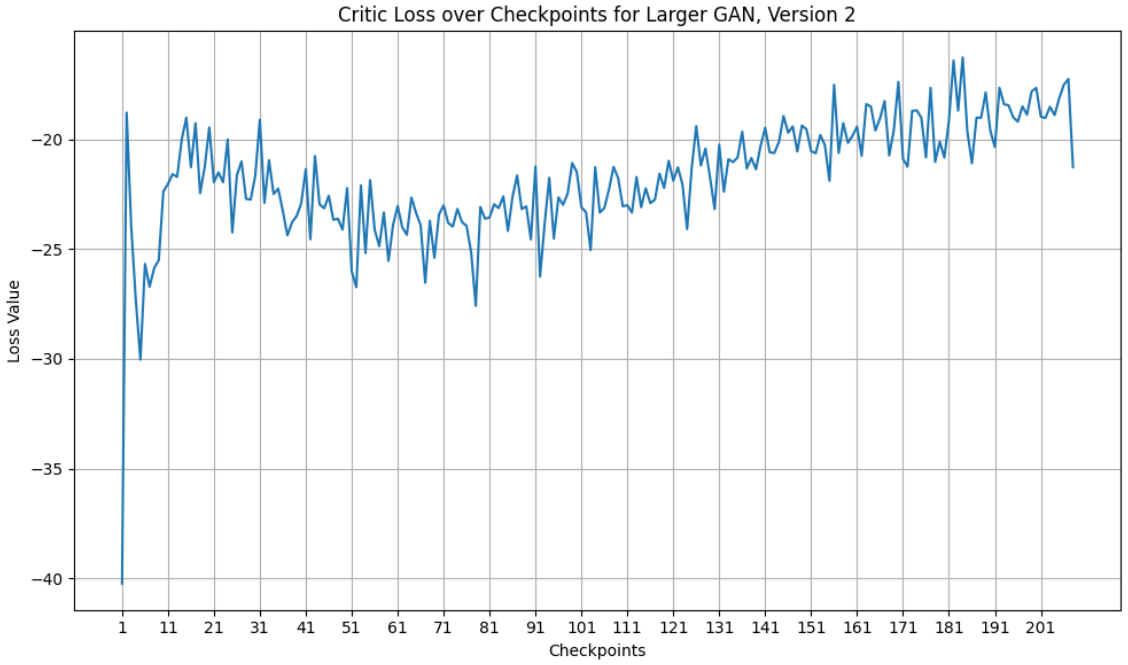

Figure 4.29: Larger GAN, Version 2 had a stable training

Chapter 5

Conclusion and Future Work

5.1 Limitations

As noted in the Hardware Characteristics subsection 4.2.4, there have been technical difficulties in terms of memory capacity when working on the local computer. The online services have larger memory space, but enforce limits unless one pays.

5.2 Future Work

The concepts discussed, techniques introduced in this report have been tested through experiments and it is assumed that the work will be continued for further increase in fidelity, diversity (in modalities), resolution of generated medical images.

Generative modeling remains an active, very broad research topic. Its particular application for image generation with the help of GANs has still room for improvement. Additionally, the medical imaging presents challenges of its own and the medical field has evolving requirements of its own. Over time, as research expands for finding tumors cure, cancer cure, and cure for other specific diseases, synthesized images will have more usage in neural networks trained to detect those diseases or to do other forms of analyses. Therefore, it would be beneficial for future authors interested in AI and medical research to think upon methods of generative models, which can help medical professionals.

In addition to generative AI methods, further applications should also be developed. Efficient algorithms for denoising, edge detection, segmentation still present research opportunities.

5.3 Conclusion

To summarize, this report proposes methods which have potential of improving generative adversarial networks and apply to unconditional image synthesis. The

methods are applicable on "larger" GANs as well for generating grayscale medical images. The experiments conducted show an improvement in produced images.

Appendix A

Supplementary Information

Borel Set σ-algebra over S is a family of subsets of S closed under complement, countable union and countable intersection. The Borel algebra over $\mathbb{R}$ is the smallest σ-algebra containing the open sets of $\mathbb{R}$.

BCE Loss

$$L_{bce} = -\frac{1}{m}\sum_{i=0}^{m} y_i \log h(x_i, \theta) + (1 - y_i) \log(1 - h(x_i, \theta)) \tag{A.1}$$

where y_i stand for labels, h is a distribution function on x_i and probability θ.

Adaptive Instance Normalization

Adaptive Instance Normalization is a normalization method that aligns the mean and variance of the content features with those of the style features.

$$AdaIN(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y) \tag{A.2}$$

where x is content input and y is style input. We align the channel-wise mean and variance of x to match those of y.

Distances discussed in WGAN paper [GAA+17]

In the coming formulas, let's denote real and generated distributions as $P_r, P_g \in Prob(\mathbb{X})$, where $\mathbb{X}$ is a compact metric set(such as the space of images $[0, 1]^d$).

- Total Variation (TV) distance

$$\delta(P_r, P_\theta) = \sup_{A \in \sigma} |P_r(x) - P_g(x)| \tag{A.3}$$

  let Σ denote the set of all Borel (explained in 2.1.1) subsets of $\mathbb{X}$

  Disadvantage: Challenging to use in stochastic gradient descent optimization, since it may not have gradient (in case there is no supremum)

- Kullback-Leibler (KL) distance

$$KL(P_r||P_g) = \int \log\left(\frac{P_r(x)}{P_g(x)}\right) P_r(x) d\mu(x) \tag{A.4}$$

  $\mu(x)$ is a measure defined on $\mathbb{X}^2$

  Disadvantage: The KL divergence is asymmetric and tends to infinity in case $P_g(x) = 0$, $P_r(x) > 0$

- Jensen-Shannon (JS) distance

$$JS(\mathbb{P}_r, \mathbb{P}_g) = KL(P_r||P_m) + KL(P_g||P_m) \tag{A.5}$$

  where $P_m = (P_g + P_r)/2$

  Disadvantage: JS divergence is unbounded, which can lead to training instability

Mathematical formulation for Random Noise 'Truncation' Mathematically formulated, let x be a random variable defined as

$$x = \Phi^{-1}(\Phi(\alpha) + \mathcal{U} \cdot (\Phi(\mathcal{B}) - \Phi(\alpha)))\sigma + \mu \tag{A.6}$$

where Φ is the cumulative distribution function of normal distribution, Φ^{-1} is its inverse, $\mathcal{U}$ a uniform random number on $(0, 1)$, following the distribution truncated to the range (a, b)

Bibliography

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[AMB+94] Brett M Averick, Jorge J Moré, Christian H Bischof, Alan Carle, and Andreas Griewank. Computing large sparse jacobian matrices using automatic differentiation. *SIAM Journal on Scientific Computing*, 15(2):285–294, 1994.

[BB22] Wilhelm Burger and Mark J Burge. *Digital image processing: An algorithmic introduction*. Springer Nature, 2022.

[BB23] Christopher Michael Bishop and Hugh Bishop. *Deep Learning - Foundations and Concepts*. 1 edition, 2023.

[BDS18] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[BS00] Robert Gardner Bartle and Donald R Sherbert. *Introduction to real analysis*, volume 2. Wiley New York, 2000.

[CWD+18] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[DN21] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[DYK+19] Salman UH Dar, Mahmut Yurt, Levent Karacan, Aykut Erdem, Erkut Erdem, and Tolga Cukur. Image synthesis in multi-contrast mri with conditional generative adversarial networks. *IEEE transactions on medical imaging*, 38(10):2375–2388, 2019.

[FAKA+18] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using gan for improved

liver lesion classification. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 289–293. IEEE, 2018.

[GAA+17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[GLW+19] Jiaqi Gu, Zeju Li, Yuanyuan Wang, Haowei Yang, Zhongwei Qiao, and Jinhua Yu. Deep generative adversarial networks for thin-section infant mr image reconstruction. *IEEE Access*, 7:68290–68304, 2019.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[GPAM+20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[IHH+22] Mohammad Nazrul Islam, Md Hasan, Md Hossain, Md Alam, Golam Rabiul, Md Zia Uddin, and Ahmet Soylu. Ct kidney dataset: Normal-cyst-tumor and stone, 2022.

[IZZE17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[JTA+22] Jiwoong J Jeong, Amara Tariq, Tobiloba Adejumo, Hari Trivedi, Judy W Gichoya, and Imon Banerjee. Systematic review of generative adversarial networks (gans) for medical image classification and segmentation. *Journal of Digital Imaging*, 35(2):137–152, 2022.

[KALL17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[KLA19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[MHL+24] J. Ma, Y. He, F. Li, et al. Segment anything in medical images. *Nature Communications*, 15:654, 2024.

[MO14] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[NTL+17] Dong Nie, Roger Trullo, Jun Lian, Caroline Petitjean, Su Ruan, Qian Wang, and Dinggang Shen. Medical image synthesis with context-aware generative adversarial networks. In *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III 20*, pages 417–425. Springer, 2017.

[NTL+18] Dong Nie, Roger Trullo, Jun Lian, Li Wang, Caroline Petitjean, Su Ruan, Qian Wang, and Dinggang Shen. Medical image synthesis with deep convolutional adversarial networks. *IEEE Transactions on Biomedical Engineering*, 65(12):2720–2730, 2018.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[S+21] K Suganthi et al. Review of medical image synthesis using gan techniques. In *ITM Web of Conferences*, volume 37, page 01005. EDP Sciences, 2021.

[STR+18] Hoo-Chang Shin, Neil A Tenenholtz, Jameson K Rogers, Christopher G Schwarz, Matthew L Senjem, Jeffrey L Gunter, Katherine P Andriole, and Mark Michalski. Medical image synthesis for data augmentation and anonymization using generative adversarial networks. In *Simulation and Synthesis in Medical Imaging: Third International Workshop, SASHIMI 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Proceedings 3*, pages 1–11. Springer, 2018.

[SWH+20] Liyan Sun, Jiexiang Wang, Yue Huang, Xinghao Ding, Hayit Greenspan, and John Paisley. An adversarial learning approach to medical image synthesis for lesion detection. *IEEE journal of biomedical and health informatics*, 24(8):2303–2314, 2020.

[YZD21] Yu Yu, Weibin Zhang, and Yun Deng. Frechet inception distance (fid) for evaluating gans. *China University of Mining Technology Beijing Graduate School*, 2021.

[YZW+19] Biting Yu, Luping Zhou, Lei Wang, Yinghuan Shi, Jurgen Fripp, and Pierrick Bourgeat. Ea-gans: edge-aware generative adversarial networks for cross-modality mr image synthesis. *IEEE transactions on medical imaging*, 38(7):1750–1762, 2019.

[ZGF17] Le Zhang, Ali Gooya, and Alejandro F Frangi. Semi-supervised assessment of incomplete lv coverage in cardiac mri using generative adversarial nets. In *Simulation and Synthesis in Medical Imaging: Second International Workshop, SASHIMI 2017, Held in Conjunction with MICCAI 2017, Québec City, QC, Canada, September 10, 2017, Proceedings 2*, pages 61–68. Springer, 2017.