Variational Autoencoders aren't Very Varied (As Generators)

Xander Hinrichsen

June 18, 2023

1 Abstract

In recent years, Generative Adversarial Networks (GANs) and Stable Diffusion Models have emerged as prominent and highly favored approaches for generative tasks. These models have garnered substantial attention due to their remarkable capabilities in generating high-quality and diverse samples. In this paper, our objective is to delve into the assessment of what constitutes a "good" generative model. To achieve this, we undertake a comparative analysis between two cutting-edge models: a state-of-the-art Generative Adversarial Network (DeepGAN) and a comparably complex yet less popular generative model, namely the Variational Autoencoder (VAE). By thoroughly examining their respective strengths, limitations, and performance characteristics, we aim to shed light on the distinguishing factors that contribute to their effectiveness in generating synthetic data.

2 Introduction

Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) are two prominent and distinct generative modeling frameworks that have garnered significant attention in recent years. These models offer distinct approaches to capturing and generating complex data distributions, each with its own strengths and limitations.

GANs, introduced by Goodfellow et al. in 2014, are based on an adversarial game between a generator and a discriminator network. The generator aims to synthesize samples that closely resemble real data, while the discriminator's objective is to differentiate between real and generated samples. Through an iterative training process, GANs learn to generate increasingly realistic outputs by constantly improving the generator's ability to deceive the discriminator. GANs are particularly effective at capturing fine-grained details and producing visually appealing, high-fidelity samples.

In contrast, VAEs, proposed by Kingma and Welling in 2013, are based on the principles of autoencoders and probabilistic modeling. VAEs consist of an encoder network that maps input data to a latent space, and a decoder network that reconstructs the data from the latent representation. Unlike GANs, VAEs explicitly model a probabilistic latent space, enabling efficient sampling and interpolation in this space. VAEs aim to learn a compressed and structured representation of the data, allowing for controlled generation and interpolation in the latent space. They prioritize the reconstruction of the input data, making them suitable for tasks that require faithful data reconstruction.

The distinction between GANs and VAEs lies in their underlying objectives and training methodologies. GANs focus on the adversarial interplay between the generator and discriminator networks, striving to achieve a dynamic equilibrium where the generator produces samples that are indistinguishable from real data. In contrast, VAEs aim to learn a latent space representation that facilitates data reconstruction and controlled generation by maximizing the likelihood of the input data given the latent representation.

These different objectives lead to contrasting strengths and limitations. GANs excel at capturing complex data distributions, generating high-quality samples, and producing visually diverse outputs. However, GANs can suffer from mode collapse, where they fail to capture the entire data distribution, resulting in limited diversity. VAEs, on the other hand, prioritize data reconstruction and offer a structured latent space that allows for controlled generation and interpolation. However, VAEs may struggle to capture fine-grained details and can sometimes produce blurry or less visually appealing samples.

3 Method/Architecture

For the BigGan, we have the model architecture as described in the original paper [BDK19]:

$z \in \mathbb{R}^{120} \sim \mathcal{N}(0, I)$ Embed$(y) \in \mathbb{R}^{128}$
Linear $(20 + 128) \to 4 \times 4 \times 16ch$
ResBlock up $16ch \to 16ch$
ResBlock up $16ch \to 8ch$
ResBlock up $8ch \to 4ch$
ResBlock up $4ch \to 2ch$
Non-Local Block (64×64)
ResBlock up $2ch \to ch$
BN, ReLU, 3×3 Conv $ch \to 3$
Tanh

(a) Generator

RGB image $x \in \mathbb{R}^{128 \times 128 \times 3}$
ResBlock down $ch \to 2ch$
Non-Local Block (64×64)
ResBlock down $2ch \to 4ch$
ResBlock down $4ch \to 8ch$
ResBlock down $8ch \to 16ch$
ResBlock down $16ch \to 16ch$
ResBlock $16ch \to 16ch$
ReLU, Global sum pooling
Embed$(y) \cdot \boldsymbol{h}$ + (linear $\to 1$)

(b) Discriminator

We also implemented and trained the BigGan-Deep for comparison:

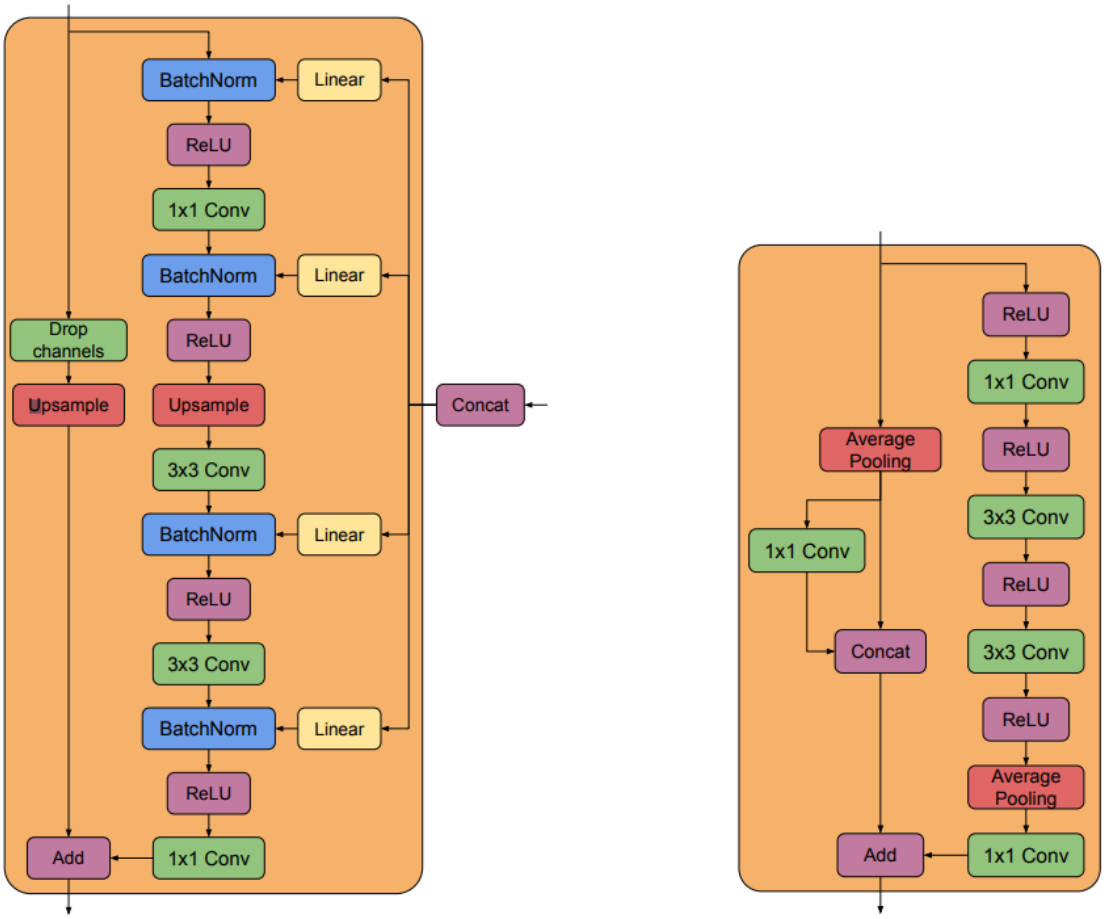
$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$ Embed$(y) \in \mathbb{R}^{128}$
Linear $(128 + 128) \to 4 \times 4 \times 16ch$
ResBlock $16ch \to 16ch$
ResBlock up $16ch \to 16ch$
ResBlock $16ch \to 16ch$
ResBlock up $16ch \to 8ch$
ResBlock $8ch \to 8ch$
ResBlock up $8ch \to 4ch$
ResBlock $4ch \to 4ch$
ResBlock up $4ch \to 2ch$
Non-Local Block (64×64)
ResBlock $2ch \to 2ch$
ResBlock up $2ch \to ch$
BN, ReLU, 3×3 Conv $ch \to 3$
Tanh

(a) Generator

RGB image $x \in \mathbb{R}^{128 \times 128 \times 3}$
3×3 Conv $3 \to ch$
ResBlock down $ch \to 2ch$
ResBlock $2ch \to 2ch$
Non-Local Block (64×64)
ResBlock down $2ch \to 4ch$
ResBlock $4ch \to 4ch$
ResBlock down $4ch \to 8ch$
ResBlock $8ch \to 8ch$
ResBlock down $8ch \to 16ch$
ResBlock $16ch \to 16ch$
ResBlock down $16ch \to 16ch$
ResBlock $16ch \to 16ch$
ReLU, Global sum pooling
Embed$(y) \cdot \boldsymbol{h}$ + (linear $\to 1$)

(b) Discriminator

The residual Blocks are modeled as the following: (b) Is the Generator's Residual Block, and (c) is the Discriminator's Residual Block [BDK19]



In this paper, we train the models on the *Cifar-10* and *Labeled Faces in the Wild* datasets, which are of resolutions 32x32 and 45x45 respectively. Since the architecture as shown in the above chart is for images of resolution 128x128, the model would be incompatible with these datasets.

To combat this and stay as loyal to the above described architecture, we simply end the generator early/slightly modify the up-scaling, where the output would be/would be near the desired resolution. In the discriminator, we simply use $log_2(imagewidth)$ number of resblock downs, and always start at 16*ch as the beginning number of channels for the resblock chain. We also trained with the BigGan-deep architecture to compare performance.

To reduce independent variables, and for simplicity, we use the BigGan Discriminator and Generator as the VAE's Encoder and Decoder as well, respectively. The only difference between the VAE and the DeepGan architecture used in this paper is the output size of the fully connected layer at the end of the Discriminator/Encoder, 1 and z, respectively.

Therefore, the only independent Variables in our experiement of comparing the GAN and the VAE are the chosen hyperparameters and training objective.

For the GAN, we use the following vinalla gan Loss function:

$$\text{Loss(D)} = -\frac{1}{n}\sum_{i=1}^{n} lnD(x) + ln(1 - D(G(z))), \text{ Loss(G)} = -\frac{1}{n}\sum_{i=1}^{n} lnD(G(z))$$

Where the objective is to minimize both losses cyclically

For the VAE we use the KL divergence in addition to reconstruction loss as described by the original VAE paper [KW13], but we instead use simple MSE for the reconstruction loss:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \simeq \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \frac{1}{L} \sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)} | \mathbf{z}^{(i,l)})$$

$$\text{where} \quad \mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)} \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$$

4 Experiments

Datasets:

To test the generative power of the models, we train and test on datasets of differing difficulty. *Labeled Faces in the Wild*, which is relatively *(*easy) for the task of generation due to its extremely centered subjects and lack of variation, in comparison to other datasets. In contrast, we also train on *Cifar-10*, which is a relatively difficult dataset for generative models, given asymmetrical scenes, un-centered subjects, and 10 different classes.

Architectures:

Further detailed in the Method section, we train on both datasets for both models, using both the BigGan and BigGan-deep architectures for each combination. We found that the BigGan-deep generally performs better. For both architectures, we also attempt adding more and more residual blocks before the other upscaling/downscaling residual blocks, and unsurprisingly, adding more residual blocks/parameters solely helps performance given enogh training time.

For the VAE, there is the architecture hyperparameter of the dimensionality of the latent space, which is also the output dimension of the VAE encoder. We tested z-dimensions of size 128, 256, and 512, under the *Labeled Faces in the Wild* dataset only. We found that, in regard to the VAE, increasing the dimension of the latent space z increased both validation reconstruction accuracy and Inception Score for the decoder's generative ability.

Optimization:

Adam caused much quicker convergence, for all models, for all datasets.

For the GAN, we attempted different learning rates from the described 2e-4 lr for the Discriminator and 5e-5 for the Generator. Just as the original paper[BDK19] found, increasing the Discriminator learning rate by a factor of 10, to 2e-3 caused the discriminator to outpace the generator and vastly slow convergence. On the other hand, increasing the Generator's lr by a factor of 10 while keeping the D lr constant, caused complete mode collapse, which the model couldn't recover from. Most likely because the generator was able to fool the discriminator because it learns quicker than the discriminator, fooling it with one output, while the discriminator wasn't able to catch up and deter this action.

Activation Functions:

Usually, Leaky ReLU, with a scaling factor of 0 to 0.2 is commonplace in GANs, and as such, we attempted to use this activation function in the BigGan and BigGan-Deep networks, however this unexpectedly hurt performance under all circumstances, so we stuck to vanilla ReLU.

5 Results and Analysis

(1) GAN best results - Cifar-10 Car & Frog, (2) VAE best gen. results - Cifar-10 Horse & Plane
(3) GAN best results - LFW, (4) VAE best gen. results = LFW

Cifar-10 Results:

Model	Architec.	Batch	Ch (mult)	ResBlock	Epochs	IS	FID
BigGan	Normal	512	96	3	314	5.9	25
BigGan	Deep	256	128	6	345	6.2	22
VAE	Normal	512	96	3	110	1.6	39
VAE	Deep	256	128	6	185	1.6	40

Labeled Faces in the Wild Results:

Model	Architec.	Batch	Ch (mult)	ResBlock	Epochs	IS	FID
BigGan	Normal	512	96	3	185	2.7	29
BigGan	Deep	256	128	6	210	2.9	27
VAE	Normal	512	96	3	170	1.3	19
VAE	Deep	256	128	6	172	1.3	15

We observed that both models demonstrated improved performance when the number of residual blocks was increased by two, resulting in a nearly doubled parameter count. Additionally, transitioning from the standard BigGan architecture to the BigGan-Deep architecture also yielded performance enhancements, despite the fact that the parameter count did not change significantly. It is noteworthy to mention that the BigGan-Deep architecture, as indicated in the original BigGan paper[BDK19], actually possesses fewer parameters compared to its counterpart, BigGan. Consequently, we can conclude that the superior performance of BigGan-Deep over BigGan can be attributed solely to its advanced architecture.

Turning our attention to the comparison between the VAE and GAN, the recorded Inception and FID scores provide intriguing insights. Specifically, when evaluated on the complex Cifar-10 dataset, the GAN outperformed the VAE comprehensively in all aspects, with a notable emphasis on significantly higher IS scores. This disparity is also evident in the visualized results, where the VAE's output diversity is limited compared to the GAN. However, in the case of the Labeled Faces in the Wild dataset, although the VAE achieved better FID scores across all instances, its IS scores remained considerably lower compared to the GAN, highlighting the VAE's limitations in generating outputs with high variety. This discrepancy can be seen in the results of the generated faces as well. While the GAN did produce faces that had glasses, mustaches, and different ethnicities, the VAE did none of the above.

Since the model architectures for the VAE and the GAN are nearly identical, we can infer that the difference in model performance is mainly due to the different training objectives

A GAN is able to learn to create representations of the dataset through self play, between the generator and the discriminator. Intuitively, it should make sense that, ideally, the GAN scores a much higher Inception Score, due to this interplay. I.E. If the generator were to produce only one image, it doesn't matter how realistic the image is, the discriminator should be able to learn that the image is fake, and penalize the gan for producing such similar images.

The VAE has no such fail safe for penalizing similar generated output. In fact, the VAE's only objectives are to minimize the squared difference of the input versus the output, and to move the latent space closer to the Gaussian distribution. Since the VAE's decoder, the generative part of the model, is learning to recreate images through reconstruction of an original image, given only a lossy compressed version of the original. Therefore, I believe that the model's most probabilistic approach is to recreate the image based on maximizing the probability that the given image is from the dataset. Intuitively, then, the VAE will default to producing images that represent the

majority features of the dataset, to minimize loss. Therefore, we can hold the VAE's training objective accountable for the low IS.

6 Conclusion

In this research paper, we conducted a comparative analysis of two prominent generative modeling frameworks: Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs). Our objective was to assess what constitutes a "good" generative model and shed light on the factors that contribute to their effectiveness in generating synthetic data.

Through our experiments and analysis, we observed that GANs excel in capturing complex data distributions, producing high-quality samples, and generating visually diverse outputs. Their adversarial training process allows them to learn fine-grained details and produce visually appealing results. However, GANs can suffer from mode collapse, limiting the diversity of their generated samples.

On the other hand, VAEs prioritize faithful data reconstruction and offer a structured latent space that enables controlled generation and interpolation. They learn a compressed representation of the data and perform well in tasks that require accurate reconstruction. However, VAEs may struggle to capture fine-grained details and can sometimes produce blurry or less visually appealing samples.

The discrepancy in performance between GANs and VAEs can be attributed to their different training objectives. GANs focus on achieving a dynamic equilibrium between the generator and discriminator networks, while VAEs aim to maximize the likelihood of the input data given the latent representation.

Our experiments also showed that architectural variations, such as increasing the number of residual blocks or transitioning to advanced architectures like BigGan-Deep, can improve the performance of both GANs and VAEs.

Overall, GANs and VAEs offer distinct approaches to generative modeling, each with its own strengths and limitations. The choice of which model to use depends on the specific requirements of the task at hand. GANs are suitable for tasks that prioritize capturing complex data distributions and producing visually diverse outputs, while VAEs are more appropriate for tasks that emphasize faithful data reconstruction and controlled generation.

Future research could explore hybrid approaches that combine the strengths of GANs and VAEs or investigate novel generative modeling frameworks to further enhance the quality and diversity of generated samples.

In conclusion, understanding the characteristics and trade-offs of different generative models, such as GANs and VAEs, contributes to advancing the field of generative machine learning and enables researchers and practitioners to choose the most suitable model for their specific needs.

References

[KW13] Diederik P. Kingma, and Max Welling (2013).*Auto-Encoding Variational Bayes* arXiv preprint arXiv:1312.6114, 2013.

[BDK19] Andrew Brock, Jeff Donahue, and Karen Simonyan. *Large Scale GAN Training for High Fidelity Natural Image Synthesis.* arXiv preprint arXiv:1809.11096., 2019.

7 Links

All of the code, except for preparing the LFW dataset, was mine, completely from scratch:
GitHub for VAE: https://github.com/Xander-Hinrichsen/Variational-Autoencoder-VAE-.git
GitHub for BigGan: https://github.com/Xander-Hinrichsen/biggan.git
Weights and Biases logging: https://wandb.ai/xhinrichsen/vae-faces?workspace=user-xhinrichsen