Learning latent space representations and application to image generation

Group GANERGY
Elhadi Chiter, Othman Hicheur, Fatma-zohra Rezkellah

Generative Adversarial Networks
Data Science Lab - IASD Master

November 2024

Table of contents

Vanilla GAN

Vanilla GAN

- Discriminator Loss:

$$\mathcal{L}_D = -\mathbb{E}_{x \sim p_{\text{data}}(x)} \left[\log D(x)\right] - \mathbb{E}_{z \sim p_z(z)} \left[\log\left(1 - D(G(z))\right)\right]$$

- Generator Loss:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z(z)} \left[\log D(G(z))\right]$$

- Improvement after introducing Batch Normalization layers in the generator architecture.

GAN	FID	Precision	Recall
Vanilla GAN	29.28	0.54	0.23
Vanilla GAN Batch Norm	26.68	0.54	0.19

Table: Comparison of GAN models based on FID, Precision, and Recall.

Figure: Vanilla GAN (Batch Norm) Results

Wasserstein GAN

WGAN

- Motivation: To address instability and mode collapse in vanilla GANs, WGANs use Wasserstein distance, leading to smoother training and improved output diversity (Inspired from [1], [2]).
- Discriminator Loss : Maximizes the difference in the discriminator's output between real and generated samples to approximate the Wasserstein distance

$$\mathcal{L}_D = -\mathbb{E}_{x\sim p_{\text{data}}}[D(x)] + \mathbb{E}_{z\sim p_z}[D(G(z))]$$

- Generator Loss: Minimizes the discriminator's output for generated samples, pushing the generator to create data closer to the real distribution.

$$\mathcal{L}_G = -\mathbb{E}_{z\sim p_z}[D(G(z))]$$

- In practice, we take the mean as the expectation.

WGAN Gradient Penalty

- A variant of WGAN that adds a kind of regularization term to the discriminator loss (a gradient penalty) ([3]).

GAN	FID	Precision	Recall
WGAN	58.41	0.5	0.27
WGAN GP	40.45	0.51	0.32

Table: Comparison of WGAN models based on FID, Precision, and Recall.

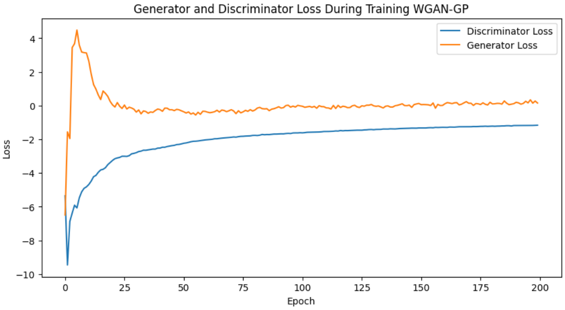

Figure: Loss of Generator and Discriminator of WGAN GP

Figure: Sample images generated by WGAN GP

Self-Attention GAN

Self-Attention GAN

Motivation:

- By taking advice and following some tricks from [4], we can stabilize the training with the use of spectral normalization for the Discriminator and batch normalization for the Generator. Moreover, we use different learning rate as the Discriminator learns faster than the Generator (TTUR).
- We use self-attention on the Discriminator in order to improve the capacity to detect the details that are far in the generated image.
- The self-attention is used after fully connected layers on the Discriminator.
- The loss of the Generator is :

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z}[D(G(z))]$$

- The loss of the Discriminator is :

$$\mathcal{L}_D = \mathbb{E}_{x \sim p_{\text{data}}}[\max(0, 1 - D(x))] + \mathbb{E}_{z \sim p_z}[\max(0, 1 + D(G(z)))]$$

Self-Attention GAN

- We note x the output of the previous layer. We compute the query $Q(x) = W_q x$, key $K(x) = W_k x$, and value $V(x) = W_v x$.
- W_q, W_k, and W_v are learned weight matrices.
- We compute after $S = Q(x) \cdot K(x)$
- We then apply the softmax function to normalize s_{ij} across all positions:

$$\alpha_{ij} = \frac{\exp(S_{ij})}{\sum_k \exp(S_{ik})}$$

- We compute the dot product with value:

$$O = \alpha \cdot V(x)$$

.

- Finally, the output is, with γ learnable parameter set to 0 initially:

$$y = \gamma O + x$$

- After, we flatten y, we use a fully connected layer and we use a sigmoid.

Self-Attention GAN

GAN	FID	Precision	Recall
$\text{SAGAN}_{100 epochs}$	22.11	??	??
$\text{SAGAN}_{200 epochs}$	18.19	0.77	0.58

Table: Comparison of GAN models based on FID, Precision, and Recall.

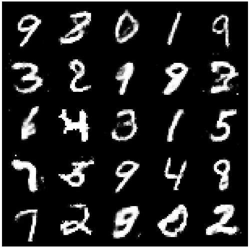

Figure: Self-attention GAN (Batch Norm) Results

Results

Results

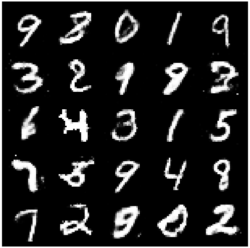

Figure: SAGAN

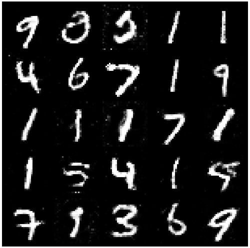

Figure: VGAN

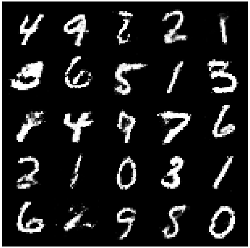

Figure: WGAN GP

Future Work

Future Work

- Hyperparameter Finetuning of SAGAN
- Implementing Unsupervised Static Gaussian Mixture GAN (in the process) [5]
- Implement (Merge) GMGAN & SAGAN
- Implement Supervised Static Gaussian Mixture GAN [5]

Thank you!

References

[1] Priya L. Mehta and Trevor Darrell. *Understanding Deep Learning: From Theory to Practice*. MIT Press, 2023.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. *arXiv preprint arXiv:1701.07875*, 2017.

[3] Ishaan Gulrajani, Gintare Karolina Dziugaite, Nikos Komodakis, and Eric Tzeng. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

[4] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. 2019.

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *CoRR*, abs/1808.10356, 2018.