MNIST Generation with GAN

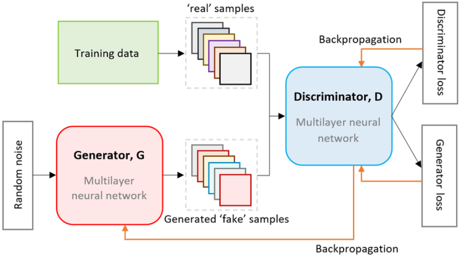
Data Science Lab - Assignment 2

Wenqing Zhang
Joey David
Dang Hoang Khang Nguyen

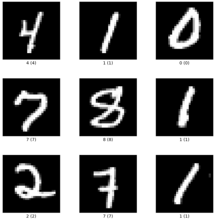
Supervisors: Alexandre Vérine
Constant Bourdrez

Problem

Problem



GAN Architecture



MNIST Dataset

- Train and improve a GAN on the MNIST dataset.
- The Generator architecture is fixed.
- Evaluate with **FID**, **Precision**, and **Recall**.

Training Improvements

Settings

BCE Loss

$$\min_G \max_D \; \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]$$

When $D(G(z)) \longrightarrow 0$,

$$\nabla_G \log(1 - D(G(z))) \longrightarrow 0,$$

so the generator's gradient vanishes.

Hinge Loss

Use hinge loss to avoid vanishing gradients and balance generator–discriminator updates [Miyato et al., 2018]:

$$\mathcal{L}_D = \mathbb{E}_x[\max(0, 1 - D(x))] + \mathbb{E}_z[\max(0, 1 + D(G(z)))]$$

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

Spectral Normalization

Normalizes each weight matrix to enforce a 1-Lipschitz discriminator for traning stability [Miyato et al., 2018]:

$$\bar{W} = \frac{W}{\sigma(W)}, \qquad \sigma(W) = \max_{\|h\|_2=1} \|Wh\|_2$$

where $\sigma(W)$ is the ℓ_2 matrix norm of W

Minibatch Standard Deviation

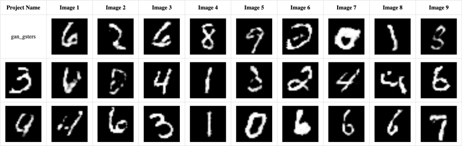
Computes variation across the minibatch and appends it to D [Karras et al., 2017]:

$$\sigma_{c,h,w} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(a_{n,c,h,w} - \bar{a}_{c,h,w}\right)^2}, \quad s = \frac{1}{CHW}\sum_{c,h,w}\sigma_{c,h,w}$$

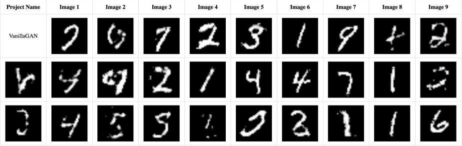
that is, compute the standard deviation for each feature in each spatial location over the minibatch, then average over all features and spatial locations to obtain a single scalar s.

Results

Model & Loss	FID ↓	Precision ↑	Recall ↑
Vanilla GAN with BCE (w.o Spectral Norm)	66.74	0.21	0.18
Vanilla GAN with BCE (w. Spectral Norm)	39.27	0.23	0.19
Vanilla GAN with Hinge loss (w. Spectral Norm)	30.38	0.27	0.20
DCGAN with Hinge loss (w. Spectral Norm)	60.53	0.22	0.17



Generations of Spectral Normalization



Generations of VanillaGAN

- Spectral normalization significantly improves generation quality and stabilizes training
- Hinge loss achieves better FID and convergence than BCE
- Vanilla GAN already performs well, convolutional discriminator adds complexity with limited gain

Wasserstein GAN with Gradient Penalty (WGAN-GP)

WGAN Goal

Improve GAN training stability by approximating the Wasserstein-1 distance between real and generated data.[Arjovsky et al., 2017]

- WGAN Objective:

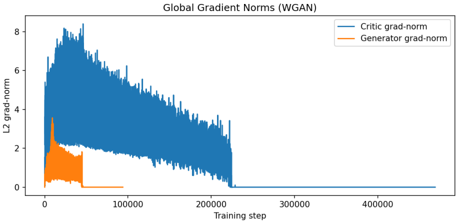
$$\min_{G} \max_{D \in \mathcal{D}} \mathbb{E}_{x \sim \mathbb{P}_r}[D(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[D(G(z))]$$

- The Discriminator (D) $\mathcal{D}$ is the set of 1-Lipschitz functions.
- Original WGAN implementation enforced the 1-Lipschitz constraint using Weight Clipping on the Critic's weights.

Wasserstein GAN with Gradient Penalty (WGAN-GP)

Drawbacks of Clipping

It can lead to exploding or vanishing gradients.



Grad norms during training process(n_critic=5)

The resluts of WGAN

Wasserstein GAN with Gradient Penalty (WGAN-GP)

Improveme of WGAN-GP

Replace weight clipping with a Gradient Penalty to enforce the 1-Lipschitz constraint. [Gulrajani et al., 2017]

$$\mathcal{L}_D = \underbrace{\mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] - \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})]}_{\text{Wasserstein Distance Approximation}} + \underbrace{\lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]}_{\text{Gradient Penalty Term}}$$

- $\hat{x}$ is a random sample interpolated between real x and fake $\tilde{x}$ data ($\hat{x} = \epsilon x + (1-\epsilon)\tilde{x}$, where $\epsilon \sim U(0,1)$).

Model & Loss	**FID ↓**	**Precision ↑**	**Recall ↑**
WGAN-GP	41.43	0.50	0.24

Differentiable Augmentations

Intuition

- Apply the **same stochastic augmentations** to both real and generated samples.
- Forces D to focus on *content* instead of superficial cues; reduces overfitting on limited data.

add real vs. fake batches sharing identical spatial jitter and color jitter.

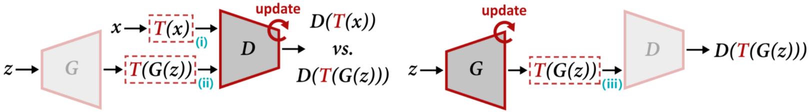
DiffAug Mechanics

Equivariance Constraint

$$\mathcal{L}_D = \mathbb{E}_{x \sim p_{\text{data}}}\left[f\Big(D\left(T\left(x\right)\right)\Big)\right] + \mathbb{E}_{z \sim p_z}\left[g\Big(D\left(T\left(G\left(z\right)\right)\right)\Big)\right]$$

$$T = T_{\text{blur}} \circ T_{\text{translation}} \circ T_{\text{cutout}}, \qquad T_{\text{real}} = T_{\text{fake}} \text{ each step}$$

- Differentiable operations keep gradients flowing: e.g. noise, blur, *translation via grid-sample*.

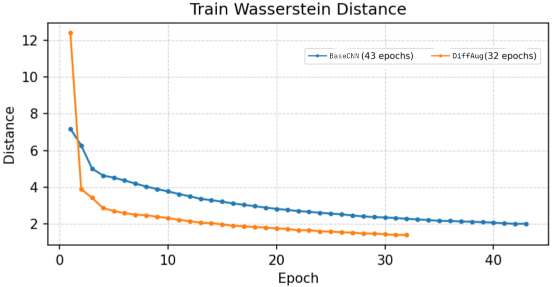


Baseline: MLP Discriminator

- Setup: D, vanilla generator, DiffAug (color + translation).
- Outcome: Complete performance breakdown, barely better than noise! (FID > 100).
- Bottleneck: MLP D misses spatial correlation; augmentations look like noise to dense layers.

Jut to be sure... Illegal CNN Upgrade

- To test whether it at least *could* work, temporarily swapped in lightweight CNN G/D (stride convs, spectral norm).
- Result: Despite some of our augmentations being still too heavy, we end up getting better recall vs. base CNN (0.82 vs 0.80).
- Convolutions don't suffer from the fixed-place inconvenients of dense layers MLP - the issue was the translations.



Integrating with Our Base Discriminator

- Use what we've learnt back into legal land: keep non-convolutional D but borrow DiffAug components that preserve alignment.
- Safe ops: Blur, add small amounts of noise, cutout with fixed grid; No translations or rotations.

Results: TBD...

References

References

Karras, T., Aila, T., Laine, S., and Lehtinen, J.
Progressive Growing of GANs for Improved Quality, Stability, and Variation.
arXiv preprint arXiv:1710.10196 (2017).

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y.
Spectral Normalization for Generative Adversarial Networks.
arXiv preprint arXiv:1802.05957 (2018).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.
Generative Adversarial Nets.
In *Advances in Neural Information Processing Systems*, Vol. 27 (2014).

Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, Song Han.
Differentiable Augmentation for Data-Efficient GAN Training.
In *arXiv preprint arXiv:2006.10738 (2020).*

References 2

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C.
Improved training of Wasserstein GANs.
Advances in Neural Information Processing Systems, 30 (2017).

Arjovsky, M., Chintala, S., and Bottou, L.
Wasserstein GAN.
arXiv preprint arXiv:1701.07875 (2017).

Thanks for your attention!