Learning to Generate Handwritten Digits with GANs (MNIST)

Anirudh Krishna
University of Maryland

November 03, 2025

Abstract

This report presents the implementation and results of a Generative Adversarial Network (GAN) trained on the MNIST dataset to synthesize handwritten digits. The objective was to enable the generator to produce realistic digit images while the discriminator learned to distinguish between real and generated samples. The project demonstrates adversarial learning, model stability, and improvement in sample quality with increasing epochs.

1. Introduction

Generative Adversarial Networks (GANs) are a class of deep learning models consisting of two networks—a generator and a discriminator—that compete in a minimax game. The generator creates synthetic data samples, and the discriminator attempts to classify them as real or fake. Over time, this adversarial training pushes the generator to produce increasingly realistic samples. The MNIST dataset of handwritten digits provides a simple yet effective benchmark to understand these dynamics.

2. Objective

The main objective was to train a GAN to generate realistic 28×28 grayscale images of handwritten digits and evaluate how the generator improves with successive epochs. This assignment fulfills the following tasks:

- Import necessary libraries and prepare the MNIST dataset.
- Define and initialize the generator and discriminator networks.
- Implement the training loops for both models using adversarial loss.
- Train the GAN for 200 epochs, saving generated images periodically.
- Plot the generator and discriminator loss across epochs.
- Analyze generated samples at epochs 1, 5, 50, 100, 150, and 200.

3. Methodology

3.1 Dataset Preparation

The MNIST dataset contains 60,000 training and 10,000 test images of handwritten digits (0–9). Each image was normalized to $[-1, 1]$ for stable training.

3.2 Generator Network

The generator maps a 100-dimensional latent noise vector z to a 28×28 image using fully connected and transposed convolutional layers. Batch normalization and ReLU activations were used to stabilize gradients, followed by a Tanh activation at the output.

3.3 Discriminator Network

The discriminator takes an image as input and outputs a scalar probability using convolutional layers with LeakyReLU activations and a final Sigmoid activation. It is trained to maximize its ability to distinguish real images from fake ones.

3.4 Training Process

The GAN training alternates between two steps:

1. **Train the Discriminator:** Feed real data and real labels, compute discriminator loss, then feed generated (fake) data with fake labels and compute the fake loss. The total loss is the sum of both.
2. **Train the Generator:** Generate fake data, pass it through the discriminator, and compute the loss using real labels to trick the discriminator.

Optimization was performed using the Adam optimizer ($\beta_1 = 0.5, \beta_2 = 0.999$, learning rate 2×10^{-4}). Each network was updated once per iteration.

4. Results and Analysis

4.1 Training Loss

The discriminator and generator losses across 200 epochs are shown in Figure 1. The discriminator loss initially rises as it learns to distinguish fake images, then stabilizes, while the generator loss decreases as it learns to produce convincing digits.

4.2 Generated Samples

Figure 2 presents generated samples at selected epochs. Early outputs (Epoch 1–5) show noisy or incomplete digits, while later epochs (100–200) exhibit clearer, more structured, and varied handwritten digits.

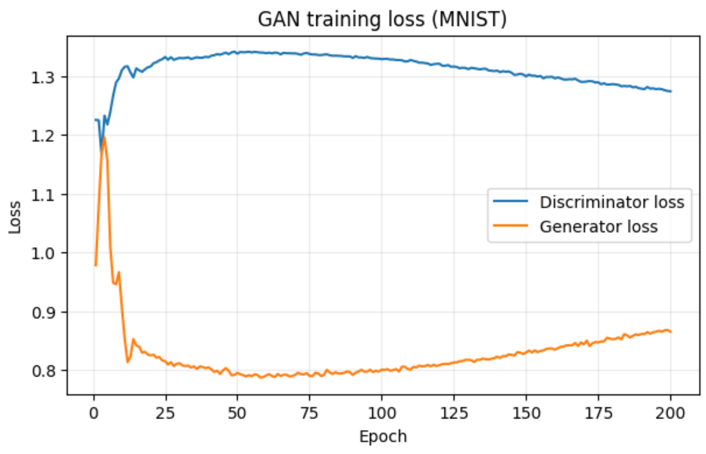

Figure 1: GAN training loss over epochs showing discriminator and generator dynamics.

5. Discussion

As training progressed, the generator learned to capture digit structure and shape diversity. Early epochs displayed blurred or indistinguishable patterns, whereas after 100 epochs, digits resembled actual MNIST samples. The loss plot indicates a balanced adversarial relationship, with both models improving in tandem.

6. Conclusion

This project successfully demonstrates adversarial training on MNIST. After 200 epochs, the GAN produces realistic handwritten digits, validating its learning capability. Future improvements could include conditional GANs or Wasserstein loss for more stable convergence.

Acknowledgment and Academic Integrity

This report and code were prepared by Anirudh Krishna as part of an academic assignment for the University of Maryland. AI tools were used solely for grammar and structure improvement. All code execution, experimentation, and analysis were independently performed.

References

- Goodfellow, I. et al., “Generative Adversarial Nets,” *NeurIPS*, 2014.
- Radford, A., Metz, L., & Chintala, S., “Unsupervised Representation Learning with DC-GANs,” *ICLR*, 2016.

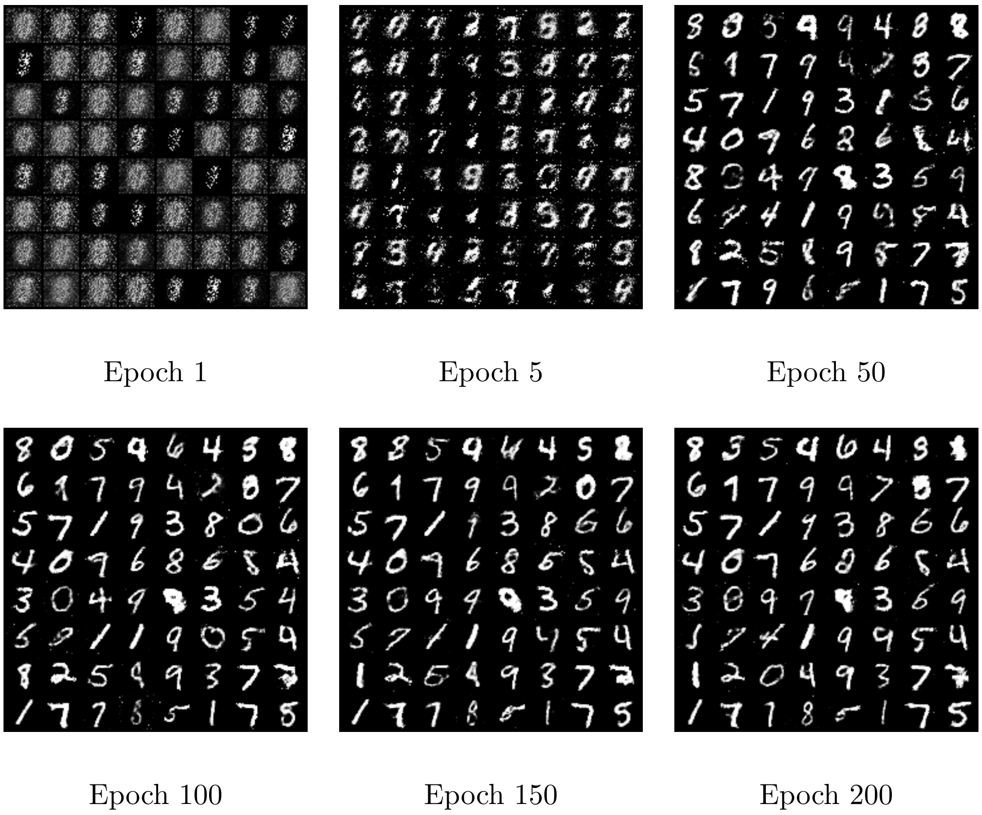

Figure 2: Generated digit samples across different epochs during training.