GAN Implementation

Matthew Evans

May 1, 2025

1 Implementation

We implemented a Generative Adversarial Network (GAN) in PyTorch to generate handwritten digits from the MNIST dataset. Both the generator G and the discriminator D are designed as multilayer perceptrons (MLPs). Training uses binary cross-entropy as the loss function, and all input images are normalized with a mean of 0.5 and a standard deviation of 0.5.

1.1 Network Architecture

- Discriminator
    1. 28×28 image input $\rightarrow$ 1024 units, Leaky ReLU (negative slope 0.2)
    2. Dropout (p=0.3)
    3. 1024 $\rightarrow$ 512 units, Leaky ReLU (0.2)
    4. Dropout (p=0.3)
    5. 512 $\rightarrow$ 256 units, Leaky ReLU (0.2)
    6. Dropout (p=0.3)
    7. 256 $\rightarrow$ 1 unit, Sigmoid
- Generator
    1. 256 input $\rightarrow$ 512 units, Leaky ReLU (0.2)
    2. 512 $\rightarrow$ 1024 units, Leaky ReLU (0.2)
    3. 1024 $\rightarrow$ 28×28 output, Tanh

1.2 Hyperparameters

- Dropout: 0.3
- Leaky ReLU negative slope: 0.2
- Size of Z: 100
- Batch size: 100
- Learning rate: 0.0002
- Epochs: 200

2 Results

An inversion of the loss covers occurs around epoch 40. The D loss approaches $\log(4)$ and G's loss approaches $-\log(\frac{1}{2})$.

This implementation uses BCE loss, which computes

$$-\mathbb{E}[y \log p + (1-y)\log(1-p)].$$

At equilibrium, the optimal discriminator is $D^*(x) = \frac{1}{2}$, so its minimax value is

$$V(D^*, G^*) = \log(\frac{1}{2}) + \log(\frac{1}{2}) = -\log(4) \approx -1.386.$$

At $D(G(z)) = \frac{1}{2}$, we have

$$-\log(1 - \frac{1}{2}) = -log(\frac{1}{2}) \approx 0.693.$$

Comfortingly, our results approach these theoretical results, as shown in table 1 and figure 1.

Table 1: Training loss by epoch and batch

Epoch	Batch	D Loss	G Loss
10	400	0.4	2.41
20	400	0.78	2.04
30	400	1.04	1.33
40	400	1.05	1.21
50	400	1.18	1.31
60	400	0.96	1.37
70	400	1.17	0.9
80	400	1.41	0.99
90	400	1.38	0.87
100	400	1.31	1.02
110	400	1.45	0.87
120	400	1.29	1.05
130	400	1.24	0.94
140	400	1.31	0.82
150	400	1.23	0.92
160	400	1.39	0.87
170	400	1.22	1.05
180	400	1.18	0.89
190	400	1.44	0.76
200	400	1.28	0.86

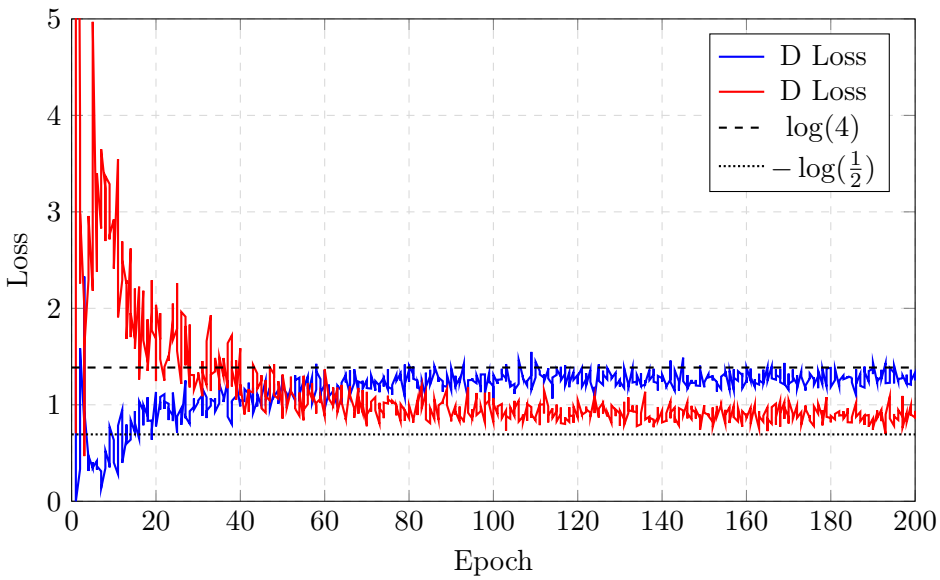
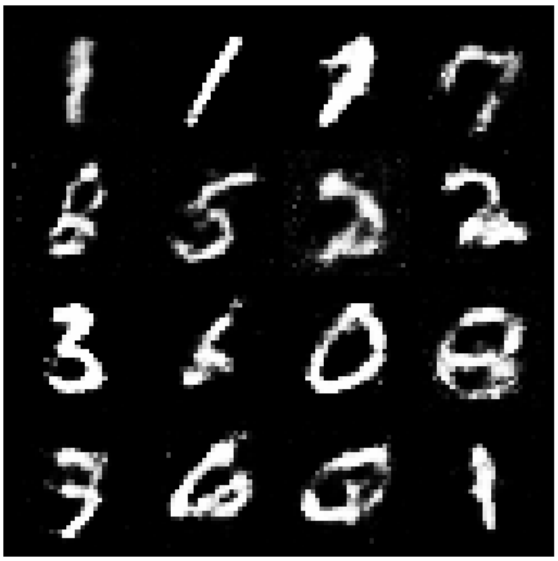


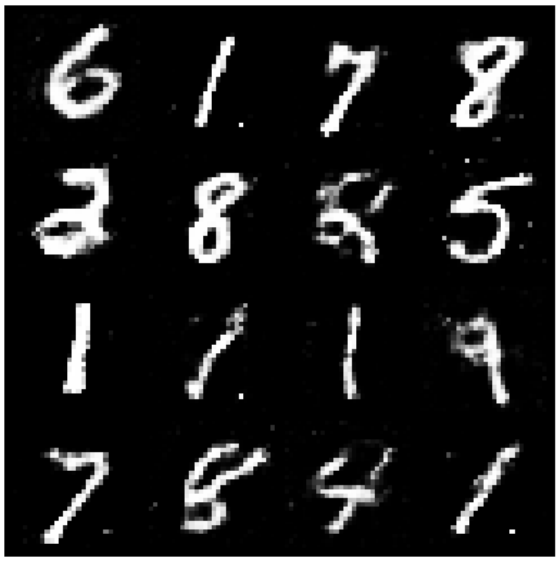
Figure 1: Training and validation loss curves over epochs.

(a) Epoch 25

(b) Epoch 50



(c) Epoch 100



(d) Epoch 200

Figure 2: Generated samples at different epochs in the learning process.