Diffusion Models

-Ashwin

Title & Objective

Class-Conditional Diffusion on MNIST

- Goal: Generate MNIST digits (0–9) from noise using a U-Net DDPM you built from scratch.
- Supports: class-conditional sampling, per-epoch progress visuals, step-by-step denoising panel.

What is a Diffusion Model?

Forward (noising): gradually add Gaussian noise to data over T steps.

Reverse (denoising): learn to predict noise ε and iteratively denoise back to an image.

Implementation:

- A **custom scheduler** (linear/cosine betas, T=1000).
- A **custom U-Net** with time + class conditioning.

Training Setup

Loss: MSE on noise prediction (ε-prediction).

Optimizer: AdamW (weight decay 1e-4), grad clip 1.0.

Scheduler: Cosine beta schedule, **T = 1000** steps.

Uses MSE

Key hyperparameters:

- **base_ch = 64**, **time_dim = 128**
- **Learning rate = 2e-4**
- **Batch size = 128**
- **Label dropout = 0.1** (for classifier-free guidance training)
- **Total epoch - 5**

The Training Process

Two-Phase Learning:

Phase 1 - Forward (Corruption):

- Take real MNIST digits
- Add increasing amounts of noise over 1000 timesteps
- $t=0$: Clean image → $t=1000$: Pure noise

Phase 2 - Learning to Denoise:

- Train U-Net to predict noise at each timestep
- Loss function: How well can we predict the added noise?
- Model learns the relationship between noisy images and noise patterns

Dataset & Architecture

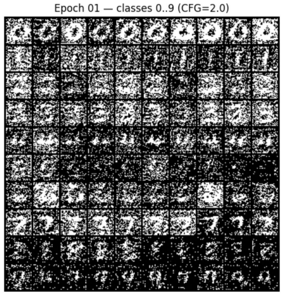
- **Dataset:** MNIST (28×28 grayscale, digits 0–9; 60k train / 10k test).
- **U-Net (from scratch):**
    - ResBlocks + GroupNorm + SiLU
    - Down/Up paths with skip connections (skip-channel fix)
    - **Time embeddings** (sinusoidal + MLP)
    - **Class embeddings** (10 digits + null label for unconditional)

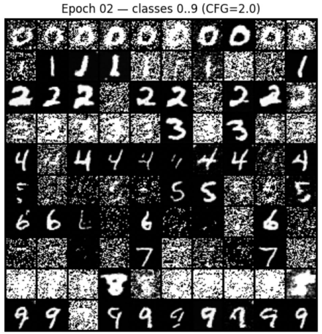
Generation Process

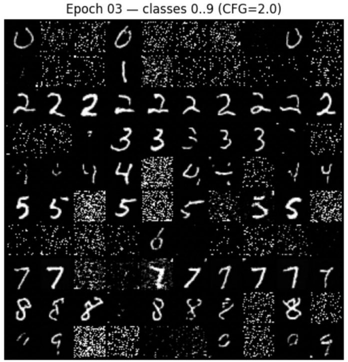
From Noise to Digits:

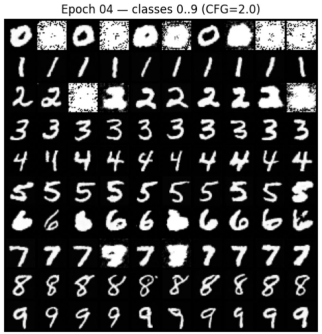
1. Start with random noise (28×28 pixels)
2. For t = 1000 down to 0:
   - Feed noisy image + timestep + (optional class) to U-Net
   - Model predicts what noise to remove
   - Remove predicted noise
   - Add small amount of controlled randomness (except final step)
3. Result: Clean, realistic handwritten digit

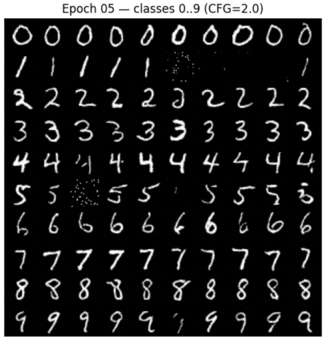
Generated images after each epoch



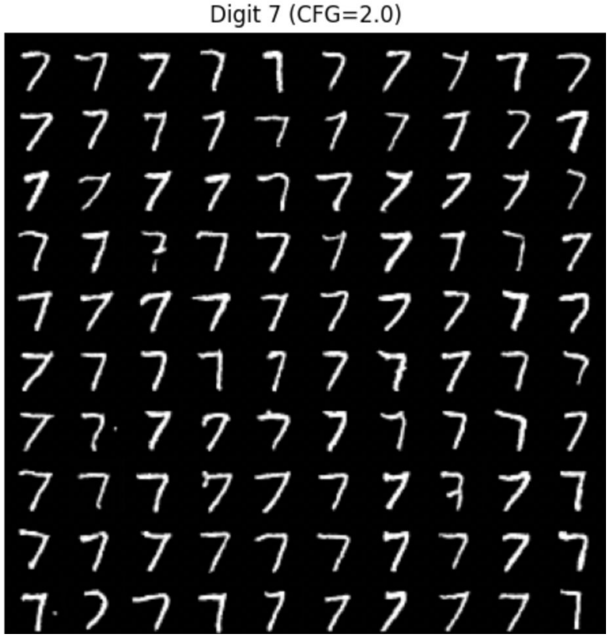


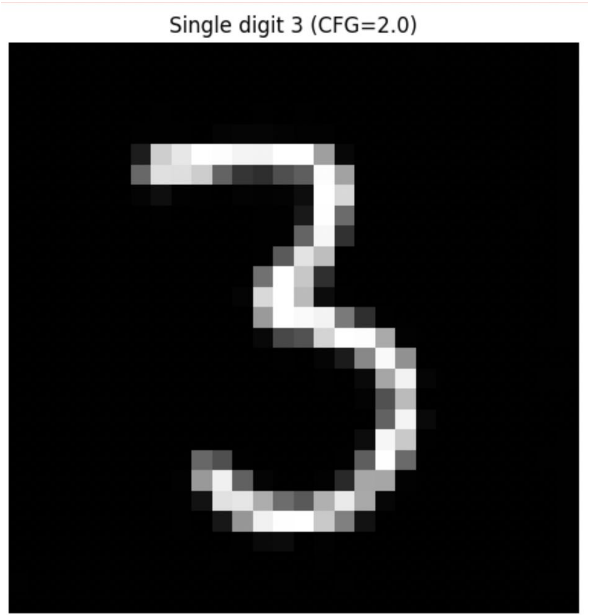






Generating specific digit





Challenges & Future Steps

Challenges Faced

- **Hyperparameter Sensitivity-**Model output quality was highly sensitive to parameters such as learning rate, diffusion steps, and EMA decay.
- Understanding the overall architecture of each of the parts and building from scratch
- **Training Time vs. Quality Trade-off**
  Achieving clearer images required more epochs, but increasing epochs significantly increased training time.

Future Scope

- Parameter Optimization with more epochs
- Longer Training with Better Scheduler Settings
- Generate image from promt