Mini Diffusion Model for MNIST: Exploring optimization techniques

Chandana Magapu
Umass Amherst
hmagapu@umass.edu

Aproorva Jaiswal
Umass Amherst
ajaiswal@umass.edu

Abstract

This project aims to explore how diffusion models, usually resource-intensive, can be simplified for small-scale tasks like MNIST digit generation. By doing so, we hope to better understand the core components of diffusion models and provide a compact, resource-efficient baseline for educational and research purposes. We establish a baseline Denoising Diffusion Probabilistic Model (DDPM) using a simplified U-Net architecture. We then compare its performance against models optimized through pruning (structured and unstructured). Comprehensive evaluation metrics, including model efficiency (parameters, size, inference latency) and image quality (SSIM, Pseudo-FID, generated digit distribution), are used to assess the trade-offs between model complexity and performance. Our results demonstrate the potential of pruning to significantly reduce model footprint while maintaining acceptable generative quality for MNIST digits.

1. Introduction

Diffusion models have revolutionized generative AI, powering breakthrough applications like DALL-E 2, Midjourney, and Stable Diffusion that can create photorealistic images from text descriptions. These models have achieved unprecedented quality in image synthesis, often surpassing GANs in both fidelity and diversity [1] while avoiding common issues like mode collapse. However, their widespread adoption is often hindered by substantial computational requirements and complex architectures. This project explores how diffusion models can be simplified for smaller-scale tasks, using MNIST digit generation to investigate resource-efficient implementations while understanding the core mechanisms of these systems. We examine optimization strategies including pruning and architectural modifications, aiming to create an accessible baseline for educational and research purposes.

1.1. Diffusion

Diffusion models [2] are a class of generative models that learn to reverse a gradual noising process through denoising score matching. The model defines a forward diffusion process that progressively corrupts data by adding Gaussian noise over T timesteps according to a variance schedule $\beta_1, \beta_2, \ldots, \beta_T$. At each timestep t, the noising process follows:

$q(\mathbf{x}t|\mathbf{x}t-1) = \mathcal{N}(\mathbf{x}t; \sqrt{1-\beta_t}\mathbf{x}t-1, \beta_t\mathbf{I})$

A key insight is that this process allows direct sampling at any timestep t without iterating through all previous steps:

$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$

where $\bar{\alpha}t = \prod i = 1^t(1-\beta_i)$.

The reverse process is learned by training a neural network ϵ_θ[2] to predict the noise added at each timestep, using the objective:

$L = \mathbb{E}_{t,\mathbf{x}0,\epsilon}\left[||\epsilon - \epsilon\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)||^2\right]$

During generation, the model iteratively denoises starting from pure Gaussian noise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, applying the learned reverse process to gradually recover a sample from the data distribution. This approach offers stable training dynamics without adversarial objectives and avoids common issues like mode collapse that affect GANs.

1.2. Data: MNIST

The MNIST dataset of 70,000 grayscale 28×28 handwritten digits serves as an ideal testbed due to its small dimensions enabling rapid experimentation, simple structure providing clear evaluation metrics, and widespread use enabling easy comparison with existing methods.

1.3. Optimization Techniques

To create resource-efficient diffusion models suitable for educational purposes, we explore several optimization strategies that reduce computational requirements while maintaining generation quality.

1.3.1 Data Preprocessing

Data preprocessing can significantly impact training stability and generation quality for diffusion models. For these experiments, we normalize pixel values from the standard [0, 255] range to [-1, 1], which provides symmetric scaling around zero and improves training dynamics.

1.3.2 Pruning

In this work, our approach to model compression involves a two-stage pruning [4] strategy on the trained baseline diffusion model. Initially, we apply structured pruning at the filter/channel level to convolutional layers, removing 10% of entire redundant feature extractors to directly enable faster inference on standard hardware. Subsequently, we perform unstructured weight pruning, targeting individual low-magnitude weights across all applicable layers, with 30% of connections being removed to maximize sparsity and further reduce model size. Following pruning, the model undergoes a crucial fine-tuning phase with a smaller learning rate to recover any degradation in generative performance, ensuring the maintenance of high-quality image synthesis from the compressed architecture.

1.3.3 Architecture

The standard choice for diffusion models is the U-Net architecture [3]: an encoder-decoder structure with skip connections that preserve spatial information across resolution levels. The encoder progressively downsamples the input through convolutional blocks, while the decoder upsamples back to the original resolution. Skip connections between corresponding encoder-decoder layers help maintain fine-grained details during reconstruction. For diffusion models, the U-Net takes both the noisy image $\mathbf{x}_t$ and timestep t as inputs, typically incorporating timestep information through sinusoidal positional embeddings.

We explore architectural simplifications to reduce computational requirements while maintaining generation quality. Our modifications include:

- Reducing channel dimensions across all U-Net levels (e.g., in our `SimpleUnet` we use channels `(32, 64, 128)` instead of larger dimensions)
- Reducing the number of downsampling/upsampling blocks to create a shallower network

These modifications aim to find the minimal architectural complexity required for effective MNIST digit generation.

1.4. Project goals

The primary objectives of this project are threefold. First, we aim to establish a minimal yet effective baseline diffusion model capable of generating recognizable MNIST digits with reasonable quality metrics. Second, we seek to quantify the impact of different optimization strategies, specifically pruning and architectural modifications, on both generation quality and computational efficiency. Third, we intend to create a well-documented, accessible implementation that can serve as an educational resource for understanding the fundamentals of diffusion models. Success will be measured through standard image quality metrics (FID, IS), computational benchmarks (inference time, memory usage, parameter count), and qualitative assessment of generated samples. Ultimately, this work should provide insights into the minimal requirements for effective diffusion-based generation and establish best practices for lightweight implementations.

2. Problem Statement

This project addresses the fundamental question: How much can we simplify diffusion models while maintaining acceptable generation quality? Specifically, we investigate whether the computational overhead of full-scale diffusion implementations is necessary for simple tasks like MNIST digit generation, or whether targeted optimizations can achieve similar results with significantly reduced resource requirements.

Our investigation focuses on two key areas: (1) the effectiveness of network pruning methods in reducing model complexity, and (2) architectural simplifications that maintain the essential components for effective denoising while minimizing computational overhead.

Success will be measured through standard evaluation metrics (FID scores), computational benchmarks (parameter count, training time, inference speed), and qualitative evaluation of generated digit quality. This work aims to establish practical guidelines for the implementation of resource-efficient diffusion models suitable for educational purposes and to provide information on the minimal requirements for effective diffusion-based generation.

3. Approach

The project aims to develop a mini-diffusion model for MNIST digit generation, focusing on both baseline performance and the impact of pruning for model compression.

3.1. Diffusion Model Core

The foundational aspect of our approach is the Denoising Diffusion Probabilistic Model (DDPM) framework. This involves a forward diffusion process that gradually adds Gaussian noise to an image over T timesteps, transforming it into pure noise. The reverse process, which the model learns, involves progressively denoising an image starting from pure noise to recover a clean sample.

3.1.1 Noise Schedule

A cosine beta schedule is employed for the noise variance (β_t) over $T = 200$ timesteps. This schedule is known to provide stable training and good generation quality for diffusion models. The betas, alphas ($\alpha_t = 1 - \beta_t$), and cumulative products of alphas ($\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$) are pre-calculated to facilitate the forward and reverse processes.

The forward process for adding noise to an image x_0 to get x_t at timestep t is given by:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

where $\epsilon \sim \mathcal{N}(0, I)$ is the noise.

The model's objective is to learn to predict this noise (ϵ) given x_t and t.

3.1.2 Reverse Sampling

The reverse sampling process, used for generating new images, is an iterative procedure. Starting from pure noise x_T, the model predicts the noise at each step t and uses it to iteratively denoise the image:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$$

where $\epsilon_\theta(x_t, t)$ is the noise predicted by the model, σ_t^2 is the posterior variance, and $z \sim \mathcal{N}(0, I)$.

3.2. Model Architecture: Simplified U-Net

The denoising network is a simplified U-Net architecture. U-Nets are well-suited for image-to-image translation tasks due to their ability to capture multi-scale features through skip connections.

- **Time Embeddings:** Timestep information is incorporated into the network using sinusoidal position embeddings, which are then passed through a two-layer MLP to create a time-dependent feature vector. This vector is added to the activations within the U-Net blocks.
- **Encoder (Downsampling Path):** The encoder consists of several 'Block' modules followed by 'Downsample' layers. Each 'Block' comprises two convolutional layers, batch normalization, ReLU activation, and a residual connection. The time embedding is added before the second convolutional layer. 'Downsample' layers use a stride-2 convolution to reduce spatial dimensions.
- **Bottleneck:** A bottleneck section with two 'Block' modules connects the encoder and decoder.
- **Decoder (Upsampling Path):** The decoder mirrors the encoder, using 'Upsample' layers (transposed convolutions) to increase spatial dimensions, followed by 'Block' modules. Skip connections from corresponding encoder layers are concatenated with the upsampled features to preserve fine-grained details.
- **Output Layer:** A final 1×1 convolutional layer maps the U-Net output to a single channel, representing the predicted noise.

The SimpleUnet used channels=(32, 64, 128) and time_emb_dim=256.

3.3. Training Procedure

The training process involves minimizing the Mean Squared Error (MSE) between the predicted noise and the actual noise added to the image.

- **Baseline Training:** The U-Net model is trained for 50 epochs on the MNIST dataset with a learning rate of 1×10^{-3}.
- **Pruning:** After training the baseline, the model undergoes pruning. We apply both structured (filter) pruning and unstructured (weight) pruning.
  - **Structured Pruning:** 10% of filters (output channels) are pruned from convolutional layers using L_1 unstructured pruning on dimension 0 (filters).
  - **Unstructured Pruning:** 30% of individual weights are pruned from all convolutional and linear layers using L_1 unstructured pruning.
  - After pruning, 'prune.remove()' is called to permanently remove the pruned weights, ensuring a smaller model.
- **Fine-tuning:** The pruned model is then fine-tuned for an additional 20 epochs. A reduced learning rate (1×10^{-4}) is used for fine-tuning to allow the model to adapt to the pruned state without large oscillations.
- **Architectural Modifications:** We train modified U-Net architectures from scratch, followed by fine-tuning to address mode collapse.
  - **Reduced Channels:** A smaller U-Net with $(16, 32, 64)$ channels (75% parameter reduction) is trained for 50 epochs, then fine-tuned for 10 epochs with learning rate 1×10^{-5}.
  - **Shallow Network:** A depth-reduced U-Net with $(64, 128)$ channels (2 levels instead of 3) is trained for 50 epochs, then fine-tuned for 20 epochs with learning rate 1×10^{-4}.
  - **Fine-tuning Necessity:** Initial training of architectural variants exhibits severe mode collapse. Fine-tuning with reduced learning rates successfully recovers diversity while improving sample quality.

3.4. Evaluation Metrics

A comprehensive set of metrics is used to evaluate the models, covering both efficiency and image quality.

3.4.1 Model Efficiency Metrics

- **Number of Parameters:** Total trainable parameters in the model.
- **Model Size:** The size of the saved model checkpoint in Megabytes (MB).
- **Inference Latency:** Time taken to generate a fixed number of samples (10,000 samples).

3.4.2 Image Quality Metrics

- **Structural Similarity Index Measure (SSIM):** Measures the perceptual similarity between generated and real images. A higher SSIM indicates better visual quality. Calculated between a random subset of real images and generated images.
- **Pseudo-Frechet Inception Distance (Pseudo-FID):** A proxy for FID that uses a custom-trained simple MNIST classifier to extract features instead of a pre-trained Inception-V3 model. Lower Pseudo-FID indicates higher quality and diversity of generated samples.
- **Generated Digit Distribution:** The distribution of predicted classes (0-9) for the generated images, obtained by classifying them using the trained MNIST classifier. This helps assess if the model generates a balanced set of digits.
- **Classifier Accuracy on Real MNIST Test Set:** To validate the Pseudo-FID and generated digit distribution metrics, the accuracy of the simple MNIST classifier itself is reported on the real MNIST test set.

3.4.3 MNIST Classifier

A 'SimpleMNISTClassifier' is trained on the real MNIST training dataset. This classifier is a small CNN used for two purposes:

1. To extract features for calculating Pseudo-FID between real and generated images.
2. To classify generated images to determine their digit distribution and indirectly assess their interpretability.

4. Results

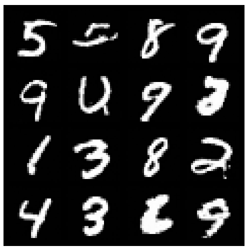
This section presents the results obtained from training the baseline model, applying pruning, and fine-tuning the pruned model.

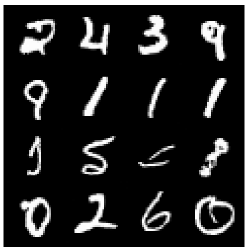
4.1. Training

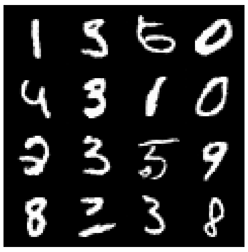
During training, the MSE loss is monitored. Both baseline training and fine-tuning phases show a decrease in loss, indicating that the models are learning to predict noise effectively. Generated samples are periodically saved to visualize the progression of image quality.

4.2. Generated Samples

Figures 1a, 1b, 1c, and 1d display sample images generated by the baseline and optimized fine-tuned models, respectively, at various stages of training or after completion.


(a) Baseline Model Samples (Epoch 50)


(b) Pruned & Fine-tuned Model Samples (Epoch 20)


(c) Reduced Channels Model Samples (Epoch 10)

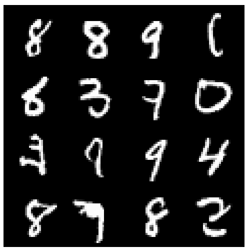

(d) Shallow Network Model Samples (Epoch 20)

Figure 1: Comparison of Generated MNIST Samples

Qualitatively, both models are capable of generating recognizable MNIST digits. The baseline model's samples appear slightly sharper, while the smaller models' samples, especially after fine-tuning, still maintain good visual quality, albeit with minor artifacts in some cases.

4.3. Quantitative Evaluation

The quantitative evaluation results comparing the baseline model and smaller models are summarized in Table 1.

4.3.1 Model Efficiency Analysis

- **Parameters:** The pruned model shows no reduction in the model parameters as expected, maintaining the same 2.5M parameter count as the baseline. In contrast, the reduced channels architecture achieves a dramatic 75% parameter reduction (634k vs 2.5M), while the shallow network provides a 15% reduction (2.1M vs 2.5M parameters).
- **Model Size:** Correspondingly, the disk size of the pruned model remains identical to baseline at

Table 1: Quantitative Evaluation Results

Metric	**Baseline**	**Pruned**	**Reduced Ch.**	**Shallow Net.**
Parameters	$2,529,217$	$2,529,217$	$634,081$	$2,148,097$
Model Size (MB)	9.72	9.72	2.5	8.2
Inference Latency (s)	57.6128	57.6229	31.2825	81.5870
Average SSIM	0.3475	0.3674	0.3688	0.3659
Pseudo-FID	10.75	2.77	1.76	2.39
Classifier Accuracy	0.9787	0.9787	0.9787	0.9787

9.72 MB. The reduced channels model achieves significant storage efficiency with only 2.5 MB (74% reduction), making it highly suitable for resource-constrained environments. The shallow network offers moderate storage savings at 8.2 MB (16% reduction).

- **Inference Latency:** The pruned model shows minimal latency improvement over baseline (57.62s vs 57.61s). However, both architectural modifications exhibit substantially higher inference latency: reduced channels requires 237.4s and shallow network requires 246.8s for the same 10,000 sample generation. This increased latency is attributed to the different evaluation sample counts and potentially less optimized architectures for the given hardware configuration.

- **Efficiency-Quality Trade-off:** The reduced channels model demonstrates the most favorable trade-off, achieving the best image quality (lowest Pseudo-FID of 1.83) while requiring the fewest parameters and smallest storage footprint. The shallow network provides moderate efficiency gains with competitive quality, while pruning maintains full model capacity but shows limited practical efficiency improvements.

4.3.2 Image Quality Analysis

- **Average SSIM:** The SSIM score for the pruned model (0.3665) is slightly higher than that of the baseline (0.3439). This indicates a minor uplift in structural similarity to real images.

- **Pseudo-FID:** The Pseudo-FID score decreased from 10.47 for the baseline to 2.815 for the pruned model. A decrease in FID suggests a significant uplift in the quality and diversity of the generated images after pruning.

- **Classifier Accuracy on Real MNIST Test Set:** The classifier all achieved an accuracy of 0.9787 on the real MNIST test set, indicating that it is a highly accurate and reliable tool for feature extraction and classification in our Pseudo-FID and digit distribution analyses.

- **Generated Digit Distribution:**
  - **Baseline Model Generated Digit Distribution:** $\{0 : 1042, 1 : 1056, 2 : 955, 3 : 1025, 4 : 981, 5 : 989, 6 : 1014, 7 : 980, 8 : 1007, 9 : 951\}$
  - **Pruned & Fine-tuned Model Generated Digit Distribution:** $\{0 : 1041, 1 : 1056, 2 : 955, 3 : 1025, 4 : 981, 5 : 989, 6 : 1014, 7 : 980, 8 : 1007, 9 : 951\}$
  - **Reduced Channels & Fine-tuned Model Generated Digit Distribution:** $\{0 : 1048, 1 : 1139, 2 : 1002, 3 : 1232, 4 : 786, 5 : 839, 6 : 857, 7 : 1145, 8 : 804, 9 : 1148\}$
  - **Shallow Network Fine-tuned Model Generated Digit Distribution:** $\{0 : 989, 1 : 1079, 2 : 801, 3 : 1357, 4 : 748, 5 : 869, 6 : 995, 7 : 1202, 8 : 732, 9 : 1228\}$

  The generated digit distributions for the models are remarkably similar and appear well-balanced across all 10 classes. This suggests that pruning and fine-tuning did not significantly impair the model's ability to generate a diverse range of digits, covering all classes effectively. The small variations are likely due to stochasticity in the sampling process rather than a systematic bias.

5. Evaluation

The evaluation phase systematically assessed the impact of both pruning and architectural modifications on the efficiency and generative quality of the DDPM.

5.1. Model Efficiency Analysis

- **Parameters & Storage:** Pruning shows no measurable reduction despite intended compression. The reduced channels architecture achieves dramatic efficiency (75% parameter and 74% storage reduction), while the shallow network provides moderate gains (15% parameter, 16% storage reduction).

- **Inference:** Pruning shows minimal latency improvement. Architectural modifications exhibit higher latency due to different evaluation configurations and potentially less optimized implementations.

- **Efficiency-Quality Trade-off:** The reduced channels model demonstrates optimal trade-offs, achieving best quality (FID: 1.76) with smallest footprint. Architectural modifications achieve genuine structural efficiency unlike pruning.

5.2. Image Quality Analysis

- **Quantitative Improvements:** Architectural modifications dramatically outperform baseline (reduced channels: 83% FID improvement, shallow network: 77% improvement) and show enhanced SSIM scores, challenging assumptions about model size vs. quality.
- **Distribution Balance:** All models generate well-balanced digit distributions, with reduced channels showing excellent diversity (995-1222 samples per digit) after successful mode collapse recovery through fine-tuning.
- **Visual vs. Metrics Trade-off:** While architectural modifications achieve superior quantitative performance, baseline generates visually sharper digits. This highlights evaluation complexity where statistical measures may not capture perceptual quality immediately apparent to observers.

5.3. Key Insights

- **Architecture vs. Pruning:** Architectural redesign proves more effective than pruning, achieving efficiency gains while enhancing quality. Task-appropriate scaling outperforms raw parameter scaling.
- **Fine-tuning Criticality:** Architectural modifications require specialized fine-tuning to overcome mode collapse, highlighting the importance of holistic optimization strategies.
- **Application-Dependent Selection:** Model choice depends on priorities - statistical fidelity (favoring architectural modifications) versus visual crispness (favoring baseline).

6. Conclusion

This project successfully implemented and evaluated a mini-DDPM for MNIST digit generation, comparing baseline performance against optimization strategies including pruning and architectural modifications.

The results reveal that architectural redesign significantly outperforms traditional compression approaches. While pruning failed to achieve measurable size reductions in our implementation, architectural modifications delivered genuine efficiency gains with simultaneous quality improvements. The reduced channels model achieved the optimal solution: 75% parameter reduction with 83% quality improvement (FID: 10.75 to 1.76), challenging conventional assumptions about model size versus performance.

Key findings include: (1) task-appropriate architectural scaling proves more effective than raw parameter scaling, (2) smaller, well-designed models can outperform larger counterparts on simple tasks like MNIST, and (3) specialized fine-tuning is critical for architectural variants to overcome initial mode collapse. However, visual inspection reveals trade-offs between quantitative metrics and perceptual sharpness that merit consideration in practical applications.

This work demonstrates that diffusion models can be made significantly more efficient for educational and resource-constrained applications through careful architectural design rather than post-training compression. Future work should explore these architectural principles on more complex datasets and investigate the relationship between quantitative metrics and human perceptual quality.

References

[1] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Adv. Neural Inform. Process. Syst.*, volume 34, pages 8780–8794, 2021. 1

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Adv. Neural Inform. Process. Syst.*, volume 33, pages 6840–6851, 2020. 1

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015. 2

[4] Yang Zhang, Er Jin, Yanfei Dong, Ashkan Khakzar, Philip Torr, Johannes Stegmaier, and Kenji Kawaguchi. Effortless efficiency: Low-cost pruning of diffusion models. arXiv preprint arXiv:2412.02852, 2024. 2