Université Paris Cité

Deep Learning

Comparative Study of WGAN and DDPM for Generative Image Restoration Tasks

Keïn Sam
Hazim Baroudi

May 2025

Led by
Ilaria Giulini and Arthur Stephanovitch

Contents

1 Introduction **3**

2 Presentation of the Models **4**

2.1 From GAN to WGAN . . . 4

2.1.1 Generative Adversarial Networks (GAN) . . . 4

2.1.2 Wasserstein Generative Adversarial Networks (WGAN) . . . 4

2.2 From SGM to DDPM . . . 5

2.2.1 Score-Based Generative Models (SGM) . . . 5

2.2.2 Denoising Diffusion Probabilistic Models (DDPM) . . . 5

3 Presentation of the Image Restoration Tasks **6**

3.1 Image Inpainting . . . 6

3.1.1 WGAN-based Inpainting . . . 6

3.1.2 DDPM-based Inpainting . . . 7

3.2 Image Outpainting . . . 7

3.2.1 WGAN-based Outpainting . . . 7

3.2.2 DDPM-based Outpainting . . . 7

3.3 Image Super-Resolution . . . 7

3.3.1 WGAN-based Super-Resolution. . . . 7

3.3.2 DDPM-based Super-Resolution . . . 8

4 Implementation of the Models **9**

4.1 Dataset Preparation . . . 9

4.2 Architecture and Model Settings . . . 9

4.3 Basic WGAN (WGAN) . . . 9

4.4 Basic DDPM . . . 9

5 Experimental Results **11**

5.1 Inpainting . . . 11

5.2 Outpainting . . . 11

5.3 Super-Resolution . . . 11

6 Results Analysis **12**

6.1 Comparison of WGAN and DDPM Performance . . . 12

6.2 Advantages and Limitations of Each Model . . . 12

6.3 Discussion of Results by Task . . . 12

7 Conclusion **13**

References **14**

Chapter 1

Introduction

Generative modeling has emerged as a central paradigm in modern deep learning, enabling machines to synthesize high-quality data samples from learned distributions. Among the most prominent frameworks are Generative Adversarial Networks (GANs) and Score-Based Generative Models (SGMs), which represent two fundamentally different approaches to learning generative distributions.

Models studied: This project focuses on a comparative study of two state-of-the-art models from each family: the Wasserstein Generative Adversarial Network (WGAN) and the Denoising Diffusion Probabilistic Model (DDPM). While WGANs leverage an adversarial objective to push the generated distribution closer to the target one via a critic function, DDPMs rely on a gradual denoising process to learn the score function of the data distribution.

Application tasks: We evaluate and compare these models on three image restoration tasks that are inpainting, outpainting, and super-resolution. Each task presents unique challenges in learning a conditional distribution $p(x \mid y)$, where y is an observed degraded or partial version of the target image x.

Project goals:

- Understand the mathematical foundations of WGAN and DDPM, from first principles to modern conditional generation techniques.
- Formulate and implement each of the three generation tasks in a controlled setting using both models.
- Perform a quantitative and qualitative comparison between the two approaches across tasks, highlighting their respective strengths and limitations.

Chapter 2

Presentation of the Models

In this section, we present the mathematical foundations of the generative models studied in this project. We begin with Generative Adversarial Networks (GANs) and their improvement into Wasserstein Generative Adversarial Networks (WGANs). We then cover Score-Based Generative Models (SGMs), leading into Denoising Diffusion Probabilistic Models (DDPMs).

Let $x \in \mathbb{R}^d$ denote a data point drawn from an unknown data distribution p_{data}. Generative models aim to learn a model distribution p_G that approximates p_{data}.

2.1 From GAN to WGAN

2.1.1 Generative Adversarial Networks (GAN)

Let $z \in \mathbb{R}^k$ be a latent variable sampled from a prior distribution p_z, typically $\mathcal{N}(0, I_k)$. The generator $G : \mathbb{R}^k \to \mathbb{R}^d$ maps z to the data space. The discriminator $D : \mathbb{R}^d \to [0,1]$ estimates the probability that a sample is real.

The original min-max optimization problem is:

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

This corresponds to minimizing the Jensen-Shannon divergence $\text{JS}(p_{\text{data}} \| p_G)$, where p_G is the distribution induced by $G(z)$. However, when $\text{supp}(p_{\text{data}}) \cap \text{supp}(p_G) = \emptyset$, gradients vanish, leading to unstable training.

2.1.2 Wasserstein Generative Adversarial Networks (WGAN)

To address this, WGAN replaces the JS divergence with the Wasserstein-1 distance :

$$W(p_{\text{data}}, p_G) = \inf_{\gamma \in \Pi(p_{\text{data}}, p_G)} \mathbb{E}_{(x,y) \sim \gamma}[\|x - y\|]$$

where $\Pi(p_{\text{data}}, p_G)$ is the set of all couplings (joint distributions) with marginals p_{data} and p_G.

By the Kantorovich-Rubinstein duality, this can be rewritten as:

$$W(p_{\text{data}}, p_G) = \sup_{\|C\|_L \leq 1} \mathbb{E}_{x \sim p_{\text{data}}}[C(x)] - \mathbb{E}_{z \sim p_z}[C(G(z))]$$

where $C : \mathbb{R}^d \to \mathbb{R}$ is a 1-Lipschitz function, often called the "critic". In practice, this 1-Lipschitz condition is enforced using weight clipping or gradient penalty (WGAN-GP).

2.2 From SGM to DDPM

2.2.1 Score-Based Generative Models (SGM)

The score function of a distribution $p(x)$ is:

$$s^*(x) := \nabla_x \log p(x)$$

For $\sigma > 0$, perturb the data as $\tilde{x} = x + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, I_d)$. A neural network $s_\theta(x, \sigma)$ is trained to approximate the score, with a denoising objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{i\sim\mathcal{U}, x\sim p_{\text{data}}, \epsilon\sim\mathcal{N}(0,I_d)} \left[\lambda(\sigma_i) \left\| s_\theta(x + \sigma_i\epsilon, \sigma_i) + \frac{\epsilon}{\sigma_i} \right\|^2 \right]$$

Sampling uses Langevin dynamics:

$$x_{t+1} = x_t + \frac{\eta}{2} s_\theta(x_t, \sigma_t) + \sqrt{\eta} \cdot \epsilon_t$$

SGMs simulate a stochastic differential equation (SDE) that transforms noise into data, but their continuous solution can be unstable and difficult to optimize.

2.2.2 Denoising Diffusion Probabilistic Models (DDPM)

DDPMs define a forward diffusion process $\{x_t\}_{t=0}^T$ with Gaussian noise:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I_d)$$

with $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$, so:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) I_d)$$

The generative model learns to reverse this process, predicting noise ϵ and using:

$$\mathcal{L}(\theta) = \mathbb{E}_{t\sim[1,T], x_0\sim p_{\text{data}}, \epsilon\sim\mathcal{N}(0,I_d)} \left[\left\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t) \right\|^2 \right]$$

Sampling iteratively:

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

where $z \sim \mathcal{N}(0, I_d)$.

Thus, DDPMs are discrete-time approximations of SGMs, learning the noise function $\epsilon_\theta(x_t, t)$ by estimating the time-dependent score $\nabla_{x_t} \log p_t(x_t)$ and where the reverse process is derived using the Euler-Maruyama scheme.

Chapter 3

Presentation of the Image Restoration Tasks

This chapter presents the mathematical formulation of the tasks used to compare WGAN and DDPM: inpainting, outpainting, and super-resolution. For each, we describe how WGANs and DDPMs are adapted to conditional generation.

Let $x \in \mathbb{R}^{H \times W \times C}$ be a full image, and y the conditional input (masked version, crop, or low-resolution). The goal is to approximate $p_\theta(x \mid y)$.

3.1 Image Inpainting

Problem. We observe a corrupted image $y = M \odot x$, where $M \in \{0,1\}^{H \times W}$ is a binary mask. The task is to reconstruct x from y.

3.1.1 WGAN-based Inpainting

The generator $G(z, y)$ is conditioned on the masked input y, and the critic $C(x, y)$ evaluates whether an image x is real or generated given the input y. The Wasserstein loss without gradient penalty is:

$$\mathcal{L}_C = \mathbb{E}_{x \sim p_{\text{data}}}[C(x, y)] - \mathbb{E}_{z \sim \mathcal{N}(0,I)}[C(G(z, y), y)]$$

$$\mathcal{L}_G = -\mathbb{E}_{z \sim \mathcal{N}(0,I)}[C(G(z, y), y)]$$

A reconstruction term is added to ensure the inpainting matches the original image:

$$\mathcal{L}_{\text{recon}} = \|M \odot x + (1 - M) \odot G(z, y) - x\|_1$$

Final generator loss:

$$\mathcal{L}_G = -D(G(z, y), y) + \lambda_{\text{rec}} \mathcal{L}_{\text{recon}}$$

3.1.2 DDPM-based Inpainting

The noise predictor $\epsilon_\theta(x_t, t, y)$ is conditioned on y, typically by masking or concatenating it. The training loss is:

$$\mathcal{L}_{\text{DDPM-inpaint}} = \mathbb{E}_{x,\epsilon,t}\left[\|(1-M)\odot(\epsilon - \epsilon_\theta(x_t, t, y))\|^2\right]$$

Only the unobserved regions are predicted.

3.2 Image Outpainting

Problem. Given a central crop $y = \mathcal{C}(x)$, predict the full image x.

3.2.1 WGAN-based Outpainting

The generator $G(z, y)$ generates full images conditioned on y, and the discriminator evaluates realism. The loss functions are:

$$\mathcal{L}_D = D(x, y) - D(G(z, y), y) + \text{GP}$$

$$\mathcal{L}_G = -D(G(z, y), y) + \lambda_{\text{rec}}\|\mathcal{C}(G(z, y)) - y\|_1$$

3.2.2 DDPM-based Outpainting

The crop y is encoded and injected into x_t at each timestep, either via concatenation, masking, or cross-attention. The training loss is:

$$\mathcal{L}_{\text{DDPM-outpaint}} = \mathbb{E}_{x,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(x_t, t, y)\|^2\right]$$

3.3 Image Super-Resolution

Problem. Given a low-resolution image $y = D(x)$, reconstruct the high-resolution image x.

3.3.1 WGAN-based Super-Resolution.

The generator $G(y)$ upsamples the image, and the discriminator $D(x, y)$ evaluates realism. The adversarial loss is:

$$\mathcal{L}_G = -D(G(y), y)$$

A content loss (pixel-wise or perceptual) is added:

$$\mathcal{L}_{\text{SR}} = \lambda_1\|x - G(y)\|_1 + \lambda_2\|\phi(x) - \phi(G(y))\|_2^2$$

Final generator loss:

$$\mathcal{L}_G = -D(G(y), y) + \mathcal{L}_{\text{SR}}$$

3.3.2 DDPM-based Super-Resolution

The low-resolution image y is embedded and injected into ϵ_θ via cross-attention or concatenation. The training loss is:

$$\mathcal{L}_{\text{DDPM-SR}} = \mathbb{E}_{x,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(x_t, t, y)\|^2\right]$$

Classifier-free guidance can be used for more control during inference.

Chapter 4

Implementation of the Models

4.1 Dataset Preparation

We used the CIFAR10 dataset (32x32 pixels images) for training. The dataset is adapted for each tasks.

4.2 Architecture and Model Settings

This section presents the PyTorch-based architectures developed for both the WGAN and thre DDPM.

4.3 Basic WGAN (WGAN)

Critic Architecture

The critic is implemented as a CNN and is composed of the following:

- Three downsampling blocks with `Conv2d`, `BatchNorm2d` and `LeakyReLU` activations.
- A final `Conv2d` layer mapping to a single scalar output.

Weight clipping is applied during training to enforce the 1-Lipschitz constraint.

Generator Architecture

The generator tries to mirror the critic and starts from a latent vector, then is composed of:

- Three upsampling blocks with `BatchNorm2d` and `LeakyReLU` activations.
- A final `Conv2d` layer with a `Tanh` activation to scale pixel values to $[-1, 1]$

This design follows the DCGAN architecture and has shown stable performance when paired with the WGAN critic.

4.4 Basic DDPM

Noise Schedule

We implemented a linear noise schedule for the β_t coefficients between fixed bounds $\beta_{\min}$ and $\beta_{\max}$, precomputing all necessary coefficients (α_t, $\bar{\alpha}_t$) for efficient sampling and training.

U-Net Architecture for ϵ_θ

The denoising model is a U-Net with skip connections and time conditioning:

- **Input:** Noisy image x_t and timestep t
- **Time Embedding:** t is passed through a sinusoidal positional encoding and an MLP
- **Encoder:** Three convolutional downsampling blocks
- **Bottleneck:** Combines spatial features with the time embedding
- **Decoder:** Three upsampling blocks with skip connections from the encoder
- **Output:** A noise tensor of the same shape as x_t

Sampling Procedure

We use a sampling loop using the precomputed noise schedule. Starting from $x_T \sim \mathcal{N}(0, I)$, the model is called repeatedly to estimate $\epsilon_\theta(x_t, t)$ and compute x_{t-1} using the reverse diffusion formula.

Chapter 5

Experimental Results

5.1 Inpainting

5.2 Outpainting

5.3 Super-Resolution

Chapter 6

Results Analysis

6.1 Comparison of WGAN and DDPM Performance

6.2 Advantages and Limitations of Each Model

6.3 Discussion of Results by Task

Chapter 7

Conclusion

Bibliography

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. 2017.

[2] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. 2021.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. 2014.

[4] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. 2017.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. 2020.

[6] Alex Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. 2021.

[7] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. 2021.

[8] Lilian Weng. From gan to wgan. https://lilianweng.github.io/posts/2017-08-20-gan/, 2017.

[9] Lilian Weng. What are diffusion models? https://lilianweng.github.io/posts/2021-07-11-diffusion-models/, 2021.