Wasserstein Generative Adversarial Networks

Naïl Khelifa
ENSAE Paris
nail.khelifa@ensae.fr

Tom Rossa
ENSAE Paris
tom.rossa@ensae.fr

Abstract

Generative modeling encompasses a class of machine learning techniques designed to capture the underlying distribution of a dataset, enabling the generation of new samples that closely resemble the original data. This constitutes an unsupervised learning problem that fundamentally consists—explicitly or implicitly—in minimizing a divergence metric between the unknown true data distribution and the an estimated distribution to produce realistic samples. Generative Adversarial Networks (GANs) [3], by capitalizing on the powerful representational capabilities of deep learning architectures, have represented one of the most significant machine learning breakthroughs in recent years. Yet, this method comes with drawbacks that justify the present project.

Contributions. Throughout this work, the various practical challenges inherent in training GANs will be empirically explored and theoretically justified. Building on the work of [2], we will demonstrate how these challenges can be mitigated through Wasserstein Generative Adversarial Networks (WGANs), supported by rigorous theoretical justifications. GitHub repository of the project.

1 Background

Notations. Throughout this work, if δ refers to one of the parameter that arise in the following, we will denote by $\mathbb{P}_\delta$ a distribution and by p_δ the probability density associated with it.

1.1 Likelihood-Based Generative Methods

Before the advent of GANs, generative models shared the common goal of estimating the data density through explicit parametric models. Assuming a parametric statistical model $\{\mathbb{P}_\theta; \theta \in \Theta\}$, these methods are based on identifying a distribution $\mathbb{P}_\theta^*$ within this parametric family that maximizes the likelihood of the given dataset,

$$p_\theta^* \in \arg\max_{\theta \in \Theta} \left\{ \frac{1}{N} \sum_{i=1}^{N} \log(p_\theta(x_i)) \right\}.$$

The theoretical foundation of this approach lies in frequentist statistics, which states that asymptotically, the density parameterized by θ^*, the maximum likelihood estimator, should be "close" to the true data distribution $\mathbb{P}_{\text{data}}$. This notion of closeness is formalized using the Kullback-Leibler [8] divergence metric $KL(\mathbb{P}_{\text{data}}||\mathbb{P}_\theta^*)$. Among the most notable approaches,

- *Markov Chain Monte Carlo (MCMC)* methods approximate the data distribution by constructing a Markov chain whose stationary distribution corresponds to $\mathbb{P}_{\text{data}}$. While theoretically robust, MCMC methods are computationally expensive and struggle to scale to high-dimensional data.
- *Variational methods:* Techniques like Variational Inference and Variational Autoencoders (VAEs) [6] optimize a parametric approximation $q_\phi(z|x)$ of the posterior distribution $p(z|x)$,

enabling tractable computation of the likelihood or its lower bound. These approaches rely on explicit assumptions about the latent structure of the data.

- *Kernel Density Estimation* is a non-parametric method that estimates $\mathbb{P}_{\text{data}}$ by averaging over kernel functions centered at the data points. While conceptually simple, it becomes inefficient in high-dimensional spaces.

These earlier approaches laid the foundation for modern generative modeling but often faced significant trade-offs in terms of flexibility, scalability, or computational cost. GANs revolutionized the field by bypassing the need for explicit density estimation, leveraging implicit modeling through adversarial learning. It will later be shown that this breakthrough is closely tied to the choice of divergence metric to be minimized in order to best approximate the true data distribution within the parametric model.

1.2 Generative Adversarial Networks

In their original form, GANs constitute a class of generative algorithms designed to produce samples from a complex distribution by learning a mapping between a simple latent space and the data space. This mechanism can be viewed as an extension of Variational Auto-Encoders (VAEs), which are built on the architecture of two neural networks,

- An *encoder* $f_w : \mathbb{R}^d \mapsto \mathbb{R}^k$ that learns to efficiently encode the information contained in a sample x into a latent representation $z \in \mathbb{R}^k$, with $k < d$, using the parameters of a distribution (often Gaussian) in the latent space.
- A *decoder* $g_\theta : \mathbb{R}^k \mapsto \mathbb{R}^d$ that reconstructs the original sample x from the latent representation z, aiming to minimize the divergence between x and the reconstruction $\hat{x}$.

The objective of VAEs is to approximate the data distribution by maximizing a lower bound on the marginal log-likelihood of the data, known as the Evidence Lower Bound (ELBO). This framework relies on the idea that information is encoded within the parameters of the latent distribution, thereby capturing the uncertainty and structure of the data through a reduced-dimensional space.

For GANs, the mapping between the latent space and the data space is established through a competition between two distinct neural networks: a generator g_θ and a discriminator D. The generator g_θ takes as input a random vector z drawn from a simple distribution $p_{\mathbf{z}}$ and produces a sample $g_\theta(z)$ in the data space. The discriminator D, on the other hand, is trained to distinguish between generated samples $g_\theta(z)$ and real data drawn from the target distribution $\mathbb{P}_{\text{data}}$. This structure is illustrated in Figure 1.

The training process is formulated as a zero-sum game, where g_θ aims to maximize the probability that D fails to distinguish between real and generated data, while D seeks to minimize its classification error. Formally, this learning process is guided by minimizing a cost function based on the *Jensen-Shannon divergence* between $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_\theta$, the distribution induced by the generator. The objective function is expressed as follows,

$$\min_{g_\theta} \max_{D} \left\{ \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_{\mathbf{z}}(z)}[\log(1 - D(g_\theta(z)))] \right\}.$$

At the discriminator's optimum, the cost function corresponds to a measure of similarity between $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_\theta$, providing an elegant solution to the problem of implicit density estimation. Specifically, when the discriminator is trained to optimality for a fixed generator, it approximates the true probability that a sample originates from the real distribution $\mathbb{P}_{\text{data}}$,

$$D_{\text{opt}}(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_\theta(x)}.$$

In this scenario, the generator's objective function can be reformulated as $-2D_{JS}(\mathbb{P}_{\text{data}} \| \mathbb{P}_\theta) + C$, where C is a constant and D_{JS} represents the Jensen-Shannon divergence, a measure of similarity between two distributions $\mathbb{P}$ and $\mathbb{Q}$,

$$D_{JS}(\mathbb{P} \| \mathbb{Q}) = \frac{1}{2} D_{\text{KL}}(\mathbb{P} \| \mathbb{M}) + \frac{1}{2} D_{\text{KL}}(\mathbb{Q} \| \mathbb{M}),$$

where,

$$\mathbb{M} = \frac{1}{2}(\mathbb{P} + \mathbb{Q}),$$

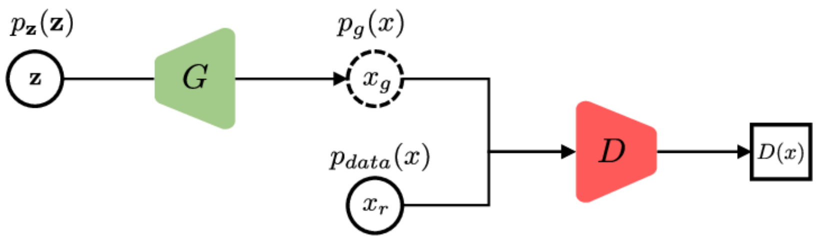


Figure 1: Structure of a generative adversarial network

and $D_{\text{KL}}(\mathbb{P}\|\mathbb{Q})$ denotes the Kullback-Leibler divergence, defined as,

$$D_{\text{KL}}(\mathbb{P}\|\mathbb{Q}) = \int p(x) \log \frac{p(x)}{q(x)}\, dx.$$

In practice, several challenges persist, including the difficulty of training g_θ and D simultaneously, unstable gradients, and the tendency for phenomena such as mode collapse. These limitations have motivated the introduction of WGANs, which aim to improve training stability and ensure better convergence properties.

1.3 Practical Limitations of GANs

The limitations inherent to models based on likelihood maximization or GANs are intrinsically tied to the divergence metric used to approximate the data distribution. In practice, the true distributions of the data that we want to model often reside on low-dimensional manifolds. This is particularly true when generating highly structured data such as images, audio, or text, which exist in high-dimensional spaces. For such data, the support of the distribution from which the dataset is drawn may be concentrated in subspaces of much lower dimensionality. This phenomenon can significantly impact learning, as demonstrated in [1]. When the supports of $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_\theta$ are viewed as two low-dimensional subspaces within a high-dimensional space, their intersection is likely to be negligible. In such cases, methods based on likelihood maximization become ineffective, as the KL divergence between the two distributions will either be undefined or infinite due to its sensitivity to mismatched supports. This explains the common practice of injecting noise into the data to diffuse their support. However, this noise injection degrades the quality of the resulting samples.

Gradient Vanishing. A similar issue arises with GANs. When the discriminator is trained to optimality, it behaves as follows:

$$D_{\text{opt}}(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_\theta(x)} = \begin{cases} \approx 0, & \text{if } x \sim p_\theta, \\ \approx 1, & \text{if } x \sim p_{\text{data}}. \end{cases}$$

This issue leads to the gradient vanishing phenomenon. When D is overly trained, the gradient used to train the generator becomes too weak, potentially stalling or significantly slowing its optimization:

$$\nabla_\theta\, \mathbb{E}_{z\sim p_{\mathbf{z}}(z)}[\log(1 - D_{\text{opt}}(g_\theta(z)))] \approx 0.$$

In other words, while the gradient estimated under an optimal discriminator is reliable, it provides limited information for guiding the generator's training. Practical solutions to mitigate this issue include the $-\log D$ trick introduced by [3], which reformulates the generator's loss as:

$$\mathcal{L}_g = -\mathbb{E}_{z\sim p_{\mathbf{z}}}[\log D(g_\theta(z))],$$

stabilizing training by ensuring meaningful gradients even when $D(g_\theta(z))$ is small. Additionally, the flexibility of GANs allows for replacing the Jensen-Shannon divergence with other f-divergences, as suggested by [9] (see Appendix A). Nonetheless, GAN training remains challenging in practice, requiring a delicate balance between training D and g_θ.

Mode Collapsing. Another well-known limitation of GANs is the mode collapse phenomenon. This occurs during training when the generator g_θ learns to produce only a limited diversity of samples. The distribution learned by the generator degenerates into a sum of Dirac deltas representing the modes for which the discriminator assigns the highest values. This issue often arises due to the competitive dynamics between g_θ and D. If D is overly powerful or g_θ converges too quickly, the generator may focus exclusively on a subset of samples that effectively deceive D, at the expense of exploring the broader sample space. It is illustrated in the experimental setting on synthetic data.

2 Theoretical Foundation of WGANs

Wasserstein Adversarial Networks (WGANs) [2] provide an alternative to classical GANs by ensuring more efficient training, reducing the occurrence of mode collapse, and offering a more reliable method to assess the generator's performance during training.

2.1 A New Divergence Metric

The theoretical innovation introduced by WGANs lies in the adoption of a new metric to replace the Jensen-Shannon divergence traditionally used in the training of GANs. This similarity measure between the true data distribution and the distribution induced by the generator is the Earth Mover's distance (EM), also known as the 1-Wasserstein distance, defined as follows:

$$\mathcal{W}(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta) := \inf_{\gamma \in \Pi(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta)} \mathbb{E}_{(X,Y)\sim\gamma}[||X - Y||_1],$$

where $\Pi(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta)$ denotes the set of joint distributions γ whose marginals are $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_\theta$, respectively. This distance induces the Wasserstein topology on the space of probability distributions. Under this metric, convergence of distributions is equivalent to convergence in the weak-* topology. In other words, if $\mathbb{P}_n \to \mathbb{P}^*$ in the Wasserstein distance, then for every bounded and continuous function f:

$$\int f(x)\, d\mathbb{P}_n(x) \to \int f(x)\, d\mathbb{P}^*(x).$$

This metric takes into account the geometry of the space in which the data reside and induces a topology that is much weaker than that of f-divergences, such as the KL divergence or the Jensen-Shannon divergence. Consequently, it is easier to construct a convergent sequence of distributions under this metric, which is therefore more robust to the absence of overlapping supports.

Thus, the training of a classical GAN can be understood as an optimization procedure of the parameter θ to create a continuous mapping $\theta_t \mapsto \mathbb{P}_{\theta_t}$ that converges to a distribution $\mathbb{P}_{\theta^*}$ minimizing the Jensen-Shannon divergence D_{JS} (or more precisely, a lower bound of it), with $\mathbb{P}_{\text{data}}$. The notion of continuity here is fully defined by the metric ρ under consideration:

$$\theta_t \mapsto \mathbb{P}_{\theta_t} \text{ is continuous } \iff \theta_t \to \theta \implies \rho(\mathbb{P}_{\theta_t}, \mathbb{P}_\theta) \to 0.$$

The authors formalize the relative strength of the topologies induced by these divergences, showing that the Kullback-Leibler (KL) divergence is the strongest, followed by the Jensen-Shannon (JS) divergence and the total variation (TV) distance, with the Earth-Mover (EM) distance being the weakest. Intuitively, it is therefore much easier to construct a mapping $\theta_t \mapsto \mathbb{P}_{\theta_t}$ that is continuous for the Wasserstein distance—since this corresponds to convergence in distribution—than for the JS divergence.

These observations strongly suggest that the EM distance is a more appropriate cost function for the problem under consideration.

2.2 Wasserstein GANs

In the framework of Wasserstein Generative Adversarial Networks (WGANs), the optimization objective is redefined to minimize the Wasserstein distance between the distribution of real data and the distribution of generated samples. The algorithm performs gradient descent on the Earth-Mover (EM) distance, and, as demonstrated, the gradients are well-defined even when the true data distribution resides on a low-dimensional manifold. This is a significant advantage over the Jensen-Shannon (JS) divergence, which lacks continuity in such settings.

The objective function of WGANs is derived from the Kantorovich-Rubinstein duality, which expresses the Wasserstein distance between the real data distribution $\mathbb{P}_{\text{data}}$ and the generated data distribution $\mathbb{P}_\theta$ as:

$$\mathcal{W}(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_{\text{data}}}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x)], \tag{1}$$

where the supremum is taken over all 1-Lipschitz functions f. In practice, this constraint is enforced by approximating f with a neural network f_w, often called the critic, and regularizing its weights to approximate the Lipschitz condition. If the supremum of program (1) is achieved for a certain set of critic network parameters, solving the problem is equivalent to:

$$\max_{w\in\mathbf{W}} \mathbb{E}_{x\sim\mathbb{P}_{\text{data}}}[f_w(x)] - \mathbb{E}_{z\sim p(z)}[f_w(g_\theta(z))], \tag{2}$$

subject to the following conditions:

- $G : (z, \theta) \mapsto g_\theta(z)$ is continuous in θ,
- G is locally Lipschitz,
- $\mathbb{E}[||Z||] < \infty$.

Under these conditions, the mapping $\theta \mapsto \mathcal{W}(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta)$ is also continuous and differentiable almost everywhere. This is particularly true for all feedforward neural networks. This inherent property of the Wasserstein distance allows the formulation of a gradient descent procedure consisting of successive optimizations of the critic network f_w and the generator g_θ:

1. **Critic Update**: The critic network is trained to maximize the difference between the expectations of $f_w(x)$ over real and generated samples. This step estimates the Wasserstein distance by learning a function f_w that best separates real and generated data. The critic's optimization program is:

$$\max_{w\in\mathbf{W}} \mathbb{E}_{x\sim\mathbb{P}_{\text{data}}}[f_w(x)] - \mathbb{E}_{z\sim p(z)}[f_w(g_\theta(z))] \approx \mathcal{W}(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta). \tag{3}$$

2. **Generator Update**: The generator is trained to minimize the estimated Wasserstein distance by producing samples that reduce the critic's ability to distinguish between real and generated data. The generator's minimization program is:

$$\min_{\theta\in\Theta} -\mathbb{E}_{z\sim p(z)}[f_w(g_\theta(z))]. \tag{4}$$

The generator's program is solved using gradient descent. Under the aforementioned conditions, and if the critic network's parameter space $\mathbf{W}$ is compact, the following holds:

$$\nabla_\theta \mathcal{W}(\mathbb{P}_{\text{data}}, \mathbb{P}_\theta) = -\mathbb{E}_{z\sim p(z)}[\nabla_\theta f_w(g_\theta(z))]. \tag{5}$$

Weight Clipping. The compactness of $\mathbf{W}$ is enforced through a clipping method, ensuring the Lipschitz continuity of the critic network f_w. Clipping involves constraining each weight $w \in \mathbf{W}$ to remain within a fixed interval $[-\tau, \tau]$, where $\tau > 0$ is a hyperparameter. After each weight update, w is replaced by $\min(\max(w, -\tau), \tau)$ whenever necessary. This method is the one presented in the paper, but it introduces several drawbacks. In particular, clipping can lead to capacity underuse, as the critic network may fail to learn effectively due to the limited representational power imposed by the constraint.

Gradient Penalty. To address these limitations, an alternative approach known as gradient penalty was introduced [4]. Instead of directly constraining the weights, this method penalizes the norm of the gradient of the critic's output with respect to its input. Specifically, the penalty term $\lambda(\|\nabla_x f_w(x)\|_2 - 1)^2$ is added to the critic's loss function, where $\lambda > 0$ is a regularization coefficient and the gradient is computed on a linearly interpolated path between real and generated samples. This penalty ensures that the critic approximates a 1-Lipschitz function without limiting its capacity. As a result, WGANs with gradient penalty exhibit more stable training dynamics and improved performance, particularly in high-dimensional or complex data settings.

Algorithm 1 Training Algorithm for Wasserstein GANs

```
1: Input: Real data distribution $\mathbb{P}_{\text{data}}$, generator $g_\theta$, critic $f_w$, learning rates $\alpha_\theta$, $\alpha_w$, clipping
   threshold $\tau$, number of critic updates per generator update $n_{\text{critic}}$
2: while not converged do
3:     for $t = 1$ to $n_{\text{critic}}$ do
4:         Sample a minibatch of real data $\{x_i\}_{i=1}^m \sim \mathbb{P}_{\text{data}}$
5:         Sample a minibatch of noise $\{z_i\}_{i=1}^m \sim p_{\mathbf{z}}$ where $p_{\mathbf{z}}$ is the noise prior
6:         Generate samples $\{g_\theta(z_i)\}_{i=1}^m$ using the generator
7:         Compute critic loss:
               $$\mathcal{L}_{\text{critic}} = -\frac{1}{m}\sum_{i=1}^m f_w(x_i) + \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z_i))$$
8:         Update critic parameters: $w \leftarrow w - \alpha_w \nabla_w \mathcal{L}_{\text{critic}}$
9:         Clip critic weights: $w \leftarrow \text{clip}(w, -\tau, \tau)$
10:    end for
11:    Sample a minibatch of noise $\{z_i\}_{i=1}^m \sim p_{\mathbf{z}}$
12:    Generate samples $\{g_\theta(z_i)\}_{i=1}^m$
13:    Compute generator loss:
               $$\mathcal{L}_{\text{generator}} = -\frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z_i))$$
14:    Update generator parameters: $\theta \leftarrow \theta - \alpha_\theta \nabla_\theta \mathcal{L}_{\text{generator}}$
15: end while
16: Output: Trained generator $g_\theta$
```

2.3 Practical Advantages of WGANs

Meaningful Loss Function. Wasserstein GANs are based on the same adversarial framework as classical GANs, wherein two networks compete: a discriminator (or critic) and a generator. However, the objective function optimized during training is derived from the Earth-Mover (EM) distance, which is continuous and differentiable almost everywhere for most standard generator architectures. This property significantly stabilizes the training process. Unlike the discriminator in a classical GAN, the critic in a WGAN benefits from being trained to optimality. When the critic reaches optimality, the loss function it maximizes (3) becomes a reliable approximation of the EM distance between the data distribution and the generator-induced distribution. In this way, the decrease in the generator's loss over iterations serves as a reliable indicator of the quality of the samples produced by the generator.

Improve Training Stability. Furthermore, the EM distance aligns well with the geometry of the data space. Unlike the discriminator in classical GANs, the WGAN critic does not suffer from saturation, even when the distributions lie on low-dimensional manifolds. This effectively addresses the gradient vanishing problem observed in classical GANs. In WGANs, gradients consistently provide meaningful information to the generator, guiding its updates and improving training stability.

All these empirical advantages of WGANs will be demonstrated in the experimental section.

Robust to Mode Collapsing. Empirical observations reveal that the generator in WGANs learns to capture the low-dimensional structure of the data distribution first, followed by finer details. This behavior explains the absence of mode collapse, a phenomenon often encountered in classical GANs. Mode collapse, in classical GANs, can be interpreted as a shortcut taken by the generator to deceive the discriminator by learning only a subset of the data distribution's modes. In contrast, WGANs encourage the generator to learn the entire data distribution more faithfully.

3 Experiments

For a comprehensive review of the experimental setup, the reader is referred to the GitHub repository of this project. In this section, the training process of our models (datasets, architectures, key

Figure 2: Examples of horses pictures in the CIFAR-10 dataset.

parameters) is first recalled. Two performances are showcased: one on a large dataset with real-world data, and another on synthetic data. Additionally, we take this opportunity to highlight the limitations of GANs, particularly the phenomena of *gradient vanishing* and *mode collapsing* observed both on synthetic and real data.

3.1 Entraînement sur données réelles

3.1.1 Dataset

The first dataset used for this experiment is the well-known CIFAR-10 dataset [7], a collection of 60,000 color images (32x32 resolution) divided into 10 equally-sized classes, corresponding to various object categories (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck).

We initially attempted to train the generator and discriminator on the entire CIFAR-10 dataset. However, we realized that the underlying distribution varied significantly across different classes, making it challenging to train the generator effectively. As a result, we decided to focus on a single class to obtain the most "homogeneous" distribution across all images. To select such a class, we intuitively chose the one whose pixel distribution seemed easiest to learn. After several trials, we settled on the *horse* class (see Figure 2), which corresponds to class index 7 in the CIFAR-10 dataset. The resulting dataset contains approximately 6,000 images exclusively from the horse class, which are used as training data.

3.1.2 Architectures

The model implemented for this experiment is a Deep Convolutional GAN (DCGAN) [10], which can be considered as an adaptation of GANs in the context image generation, as the only difference lies in the fact that the generator and the discriminator are parametrized by deep convolutionnal networks. In our setting we have:

Discriminator Network The discriminator, $D(x)$, is a convolutional neural network that outputs the scalar probability of input x coming from the real dataset rather than the generator. The network structure consists in four convolutional layers with LeakyReLU activation functions, a Sigmoid activation in the final convolutional layer, and a fully connected layer mapping extracted features to a probability score.

Generator Network The generator, $G(z)$, maps a latent vector $z \sim \mathcal{N}(0, 1)$ to the data space. It is implemented as a deconvolutional neural network that upscales the latent space to generate realistic images. The network structure consists in a linear layer that maps the latent vector to a feature map, four transposed convolutional layers upscale the feature maps, batch normalization layers and ReLU activation ensure stability during training. A Sigmoid activation in the final layer produces pixel values in $[0, 1]$.

3.1.3 Training and key parameters

The training loop alternates between maximizing the probability of correctly classifying real and fake samples (*discriminator training*) and minimizing the discriminator's ability to distinguish fake samples from real ones (*generator training*). Gradient norms for both networks are tracked to monitor vanishing gradients.

For the loss function, we considered the Binary Cross Entropy Loss (`BCELoss`) both for the generator and discriminator. For the optimizer, we considered Adam optimizers [5] with the traditional values $\beta_1 = 0.5$ and $\beta_2 = 0.99$. The experiment uses the following parameters are describe in Table 1. The choice of these parameters was suggested by those made in the seminal papers [2], [10] and [4].

Parameter	**Value**
Image size	$(3, 32, 32)$ (CIFAR-10 images)
Latent dimension	100
Batch size	16
Number of epochs	1400
Learning rate	1×10^{-5}

Table 1: GAN Hyperparameters

3.1.4 Results

Images We examined the performance of the generator over the course of training, both for GANs and WGANs. With the parameters fixed, images were generated by the GAN and WGAN generators every 100 epochs (controlled by the parameter `img_plot_periodicity`). The results for the first 900 epochs are presented in Figures 3 and 4.

It was observed that DCGANs can be more difficult to train in practice. Notably, the quality of the generated samples did not significantly improve as the generator's loss decreased, as illustrated in Figure 3a. In contrast, as detailed in Section 3.3, the decrease in the WGAN generator's loss was accompanied by a marked improvement in the quality of the generated samples (see Figure 3b). This difference arises because the WGAN loss function provides a reliable estimate of the Earth Mover's (EM) distance between the distribution induced by the generator and the real data distribution. Consequently, it serves as a reliable indicator of the generator's progress.

Gradient Vanishing To understand the differential performance of the generators observed between Figures 4 and 3, we noted that the only difference between these two experiments was the loss function, while the network architectures remained fixed and similar. We revisited the main limitations of GANs, as discussed in the theoretical framework of this work. Specifically, we observed that the gradients of the generator became vanishingly small when the discriminator was nearly optimally trained. To investigate this further, we tracked the evolution of the gradient norms for both networks throughout training (in parallel with the previously conducted training runs). The results are presented in Figure 6.

It is evident that when the discriminator becomes too effective too quickly, as observed in this case, it efficiently classifies real data from simulated data. When the support of the data distribution and the generator-induced distribution are nearly disjoint, the discriminator's output is close to 0 or 1 for most inputs. This behavior prevents the generator from extracting useful information to improve itself; in other words, the gradient of its optimization problem is effectively pushed towards zero:

$$\nabla_\theta \, \mathbb{E}_{z \sim p_{\mathbf{z}}(z)}[\log(1 - D_{\text{opt}}(g_\theta(z)))] \approx 0.$$

3.2 Training on Synthetic Data

The aim of this experiment is to study the phenomenon of *mode collapse* in a GAN. This phenomenon occurs when the generator produces similar or identical samples, failing to capture the full diversity of the target distribution.

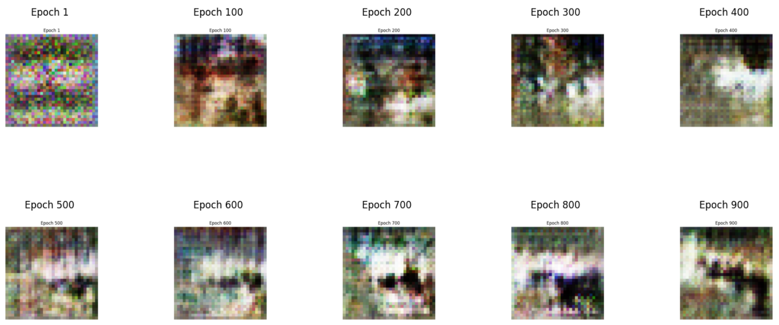


Figure 3: (a) DCGAN

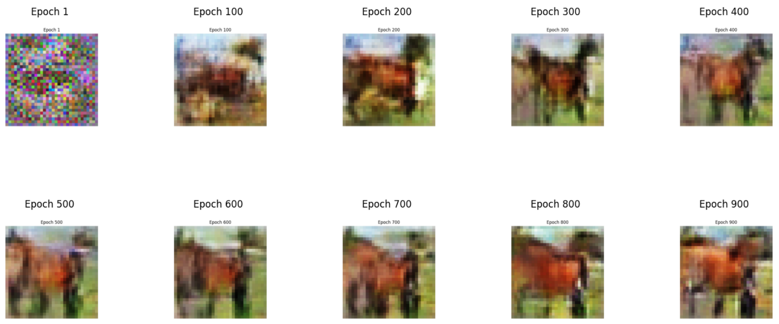


Figure 4: (b) WGAN

Figure 5: Images generated by the generator every 1000 epochs with a fixed random latent vector (`fixed_latent_vector = torch.randn((1, self.generator.latent_dim))`). The first image corresponds to the performance at epoch 0. The two images depict results from DCGAN (a) and WGAN-GP (b).

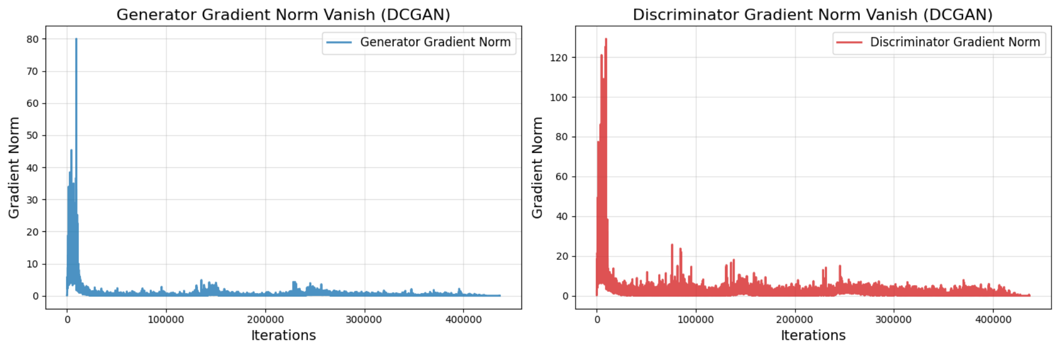


Figure 6: $\|\nabla G(\cdot)\|$ and $\|\nabla D(\cdot)\|$ depending on the iterations.

3.2.1 Dataset

The data used in this experiment is generated in the form of a 2D *Gaussian ring*. Specifically, the data is sampled from a Gaussian mixture model:

$$X_{\text{sample}} \sim \sum_{i=1}^{8} \frac{1}{8} \mathcal{N}(\mu_i, \Sigma)$$

where the means μ_i are uniformly distributed on a circle and the variance $\Sigma = 0.05 \cdot I_2$ is constant.

3.2.2 Architectures

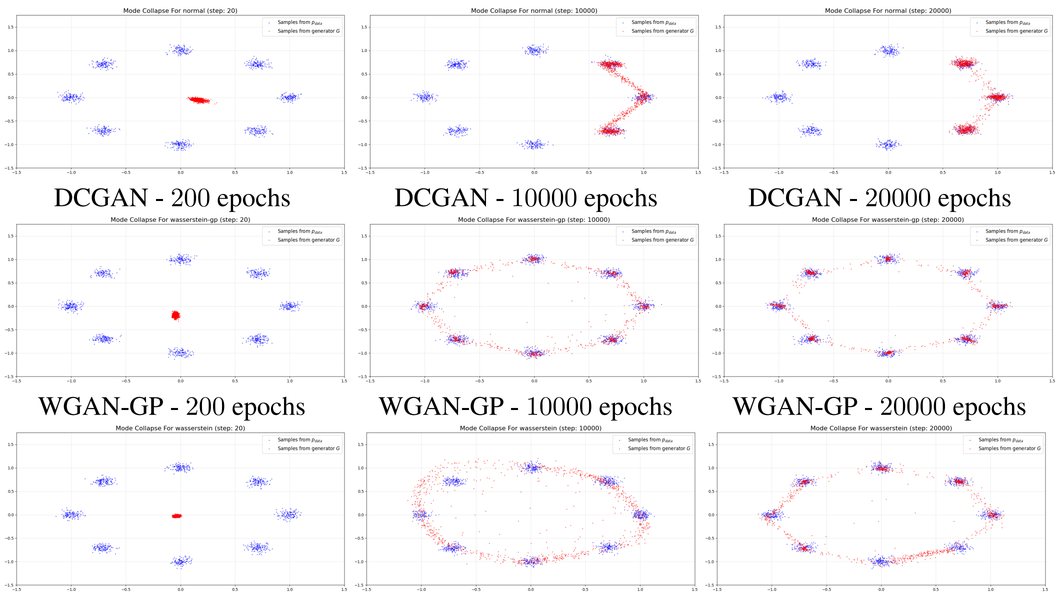
Generator The generator is a neural network composed of several fully connected layers, activated by *LeakyReLU* and *Tanh*. It takes a latent vector as input and generates 2D samples.

Discriminator The discriminator is also a multi-layer neural network, using *LeakyReLU*, which produces a probability indicating whether the sample is real or generated.

3.2.3 Results - Mode Collapsing

For the dataset and architectures defined above, we compare the performance of a standard GAN, a WGAN with weight clipping on the critic's network, and a WGAN with gradient penalty. As discussed in Section 3.2, gradient penalty is an alternative method to enforce the Lipschitz continuity of the functions f_w corresponding to the critic. This approach allows for greater modeling flexibility in the neural network without imposing direct constraints on the weights.

Similar to the CIFAR dataset, samples are generated progressively during training. The goal is to observe the extent to which the generator manages to cover the full range of modes in the target distribution. It can indeed be observed that DCGAN successfully generates high-quality samples,



DCGAN - 200 epochs DCGAN - 10000 epochs DCGAN - 20000 epochs

WGAN-GP - 200 epochs WGAN-GP - 10000 epochs WGAN-GP - 20000 epochs

WGAN-CP - 200 epochs WGAN-CP - 10000 epochs WGAN-CP - 20000 epochs

Figure 7: Comparison of DCGAN, WGAN, and WGAN with CP at various epochs.

in the sense that they are concentrated within certain modes of the target distribution. This aligns with the observation made in [3], where GANs tend to focus on learning only specific modes of the distribution—those that are most effective in deceiving the discriminator. As demonstrated here, the exploration capacity of GANs is limited, and even at the end of training, the samples only occupy 3 of the 8 modes in the mixture. In contrast, WGANs appear to be less prone to the phenomenon of mode collapsing. This behavior can be attributed to the flexibility provided by the Earth Mover's (EM) objective function, which strongly penalizes samples that deviate significantly from the support of the true distribution, thereby encouraging a more effective exploration of the data space. It is also observed that GANs with clipping initially seem to capture the underlying structure of the distribution, such as the circle in this case, and then focus on the modes. This phenomenon can be interpreted as the effective use of the space geometry for the Wasserstein distance, whereas normal GANs are more constrained in sampling high-density points.

4 Conclusion

Throughout this project, the focus was placed on the innovation represented by Wasserstein GANs within the field of generative models. The study aimed to understand the various limitations inherent in training classical GANs, both from a theoretical standpoint and based on practical observations. The project sought to justify how WGANs, by modifying the objective function for both generators and discriminators, could address the limitations of traditional GANs. Through a series of experiments conducted on both synthetic and real-world data, these innovations were empirically illustrated. The phenomenon of *gradient vanishing* during the training of GANs was observed, highlighting the challenges associated with their training process. Subsequently, through a toy example, it was demonstrated that WGANs, whether employing clipping or gradient penalty, are less prone to *mode collapsing*.

Looking ahead, several avenues could be explored to extend this work, especially if more time and technical resources were available. One direction would be to conduct experiments on more complex datasets, such as high-resolution images or videos, to assess the scalability and robustness of WGANs in high-dimensional settings. Additionally, exploring these models on larger datasets and incorporating more advanced computational resources could provide deeper insights into their performance and generalization capabilities. Another promising approach is to use the Wasserstein distance, estimated by the WGAN objective, as a quantitative metric for evaluating generative models. This would provide a more rigorous, data-driven way to compare models, moving beyond qualitative assessments. Additionally, exploring variations of WGANs, such as those incorporating Wasserstein distance regularization or alternative optimization techniques, could improve training stability and sample quality. Analyzing the interplay between clipping and gradient penalty techniques would also be valuable for understanding their impact on model performance. Finally, combining WGANs with advanced techniques like self-supervised or semi-supervised learning could enhance the model's ability to learn from limited labeled data, expanding its applicability to real-world scenarios.

References

[1] Martin Arjovsky and Léon Bottou. “Towards principled methods for training generative adversarial networks”. In: *International Conference on Learning Representations*. 2017.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein generative adversarial networks”. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML’17. Sydney, NSW, Australia: JMLR.org, 2017, pp. 214–223.

[3] Ian J. Goodfellow et al. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems 27 (NeurIPS)*. Curran Associates, Inc., 2014, pp. 2672–2680.

[4] Ishaan Gulrajani et al. “Improved Training of Wasserstein GANs”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017, pp. 5767–5777.

[5] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR* (2014).

[6] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *CoRR* (2013).

[7] Alex Krizhevsky, Vinod Nair, Geoffrey Hinton, et al. “The CIFAR-10 dataset”. In: *online: http://www. cs. toronto. edu/kriz/cifar. html* 55.5 (2014), p. 2.

[8] Solomon Kullback and Richard A. Leibler. “On Information and Sufficiency”. In: *Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86. DOI: 10.1214/aoms/1177729694.

[9] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. “f-GAN: Training Generative Neural Samplers Using Variational Divergence Minimization”. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016, pp. 271–279.

[10] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. In: *CoRR* (2015).

5 Appendix

5.1 Appendix A

The f-divergences constitute a general family of measures designed to quantify the difference between two distributions P and Q. They are defined as:

$$D_f(P\|Q) = \int f\left(\frac{p(x)}{q(x)}\right) q(x)\, dx,$$

where $f : \mathbb{R}_+ \to \mathbb{R}$ is a convex function satisfying $f(1) = 0$. This definition encompasses well-known divergences such as the Kullback-Leibler divergence ($f(t) = t \log t$) and the Jensen-Shannon divergence.

In the context of Generative Adversarial Networks (GANs), the original objective of minimizing the Jensen-Shannon divergence can be generalized using an f-divergence. This approach involves adjusting the cost function to minimize a specific divergence between the real data distribution p_{data} and the generated distribution p_θ. This generalization, introduced in f-GANs ([9]), enables the selection of the function f to influence the convergence properties and training behavior, making GANs adaptable to various practical scenarios.