Optimal Transport : Final Project
Improving GANs Using Optimal Transport

Maxime Berillon and Rémy Deshayes

April 28th 2021

Generative modeling is an unsupervised learning task that aims at discovering and learning the underlying structure of an input data in such a way that the model can be used to generate new examples that plausibly could have been drawn from the input's dataset.
As always with unsupervised learning our main task will amount to learning the probability distribution of that data at hand.

Section 1 introduces one method to learn a probability distribution and the GANs a very famous generative method leveraging this technique. Then, section 2 discusses the concept of distance between probability distributions and presents a very famous distance taken from the optimal transport theory : the Wasserstein distance. As an alternative to the WGAN, section 2 also compares the primal and dual approach of the optimal transport problem introduced. Building on those remarks, section 3, introduces the OT-GAN and its implementation[1].

1 Introduction

1.1 Intuitions on learning a probability distribution

One method - often introduced as an alternative to the maximum likelihood approach - is to define a random variable Z with a fixed distribution $p(z)$ and then use it as an input for a parametric function $g_\theta : \mathcal{Z} \to \mathcal{X}$ that generates samples following a distribution $\mathbb{P}_\theta$.
Varying the parameter θ will allow to change $\mathbb{P}_\theta$ and make it as close as possible to the real distribution.

GANs, that we are studying in our project, leverage this approach. Let us then briefly introduce the GAN model in part 1.2.

1.2 Brief introduction to the GAN model

Definition : GAN

A Generative Adversarial Network is an unsupervised **generative model** where **two networks compete** with each other in a game theory setting.

The first network is called a **generator**, it generates a sample. Its adversary, the other network, is called a **discriminator** and tries to detect if a sample is genuine or if its an output from the generator. The learning procedure can be modeled as a zero sum game - one model's gain is the other's loss.

[1] Notebook can be found on Github here

The generator and the discriminator are multilayer perceptrons meaning they can be any usual deep learning framework, independently.

There is a lot of use-cases for GANs notably in image upsampling, artistic creation and reinforcement learning. Let us use of this use case, namely the art forger, to better understand the GAN's operational principle. A common analogy is that of an art forger - the generator - who tries to forge paintings and an investigator - the discriminator - who tries to detect imitations.

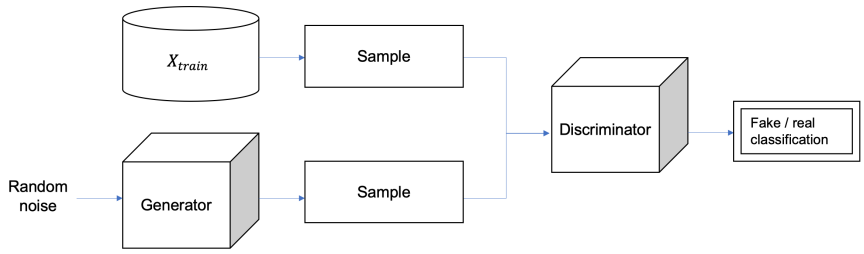


Figure 1: GAN's framework high level view - an interesting takeout from this framework is that the generator has no knowledge of the real data, the only message it will get is the discriminator's output.

That said, training a GAN amounts to a simultaneous 2-players minimax game - the generator (G) tries to fool the discriminator (D) and D tries to avoid being fooled by G.

Let us describe a little bit more this minimax game, this will be useful in later sections and notably in **??**.

As D produces a binary fake vs. true classification, the binary-cross entropy is chosen as a cost function :

$$V = -(y \log(D(x)) + (1-y) \log(1 - (D(x)))$$

where y is a class indicator - 0 for false and 1 for true - and $D(x)$ is the probability to be true output by the discriminator.

Let us denote $\hat{x}$ the false samples, D cost function is as follows :

$$V(D) = -[\log D(x) + \log(1 - D(\hat{x}))]$$

Now, adding G we get:

$$V(D) = -[\log D(x) + \log(1 - D(G(z)))]$$

Eventually, as we are considering multiple observations we end up having :

$$V(D) = -\left[\mathbb{E}_{x \sim p_{\text{data}}\,(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]\right]$$

where $z \sim p_z(z)$, the chosen distribution.

D optimization program amounts to minimizing its cost function:

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

For its part, G tries to fool D, its cost function is thus the opposite of D's one:

$$V(G) = -V(D)$$

and its optimization program is given by :

$$\min_G V(G) = \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

Hence our 2-players minimax game between D and G:

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

Now that we know a little bit more about the GAN framework, another thing to know is that training GANs can usually suffer from three issues :

- Non-convergence: the model parameters oscillate, destabilize and never converge
- Mode collapse: the generator collapses which produces limited varieties of samples
- Diminished gradient: the discriminator gets too successful that the generator gradient vanishes and learns nothing

Various architectures and alternatives have been proposed as a workaround to these problems. In this context, section 2 introduces one very famous alternative the Wasserstein GAN. The WGAN will be very useful to introduce Optimal Transport concepts in order to ultimately introduce the OT-GAN, focus of this project.

2 Wasserstein GAN - [1]

2.1 The approach in practice

2.1.1 Impact of the distance measure

Recalling the remarks made in 1.1, in order to use this approach and appropriately train g_θ, one needs to define a distance measure between the model and real data distributions which amounts to a crucial discussion on the various ways possible to define a distance $\rho(\mathbb{P}_\theta, \mathbb{P}_r)$.

Indeed, when trying to optimize the θ parameter it is best to have $\mathbb{P}_\theta$ such that $\theta \mapsto \mathbb{P}_\theta$ is continuous, meaning that $\theta_t \to \theta$ yields $\mathbb{P}_{\theta_t} \to \mathbb{P}_\theta$.
However, let us recall the definition of the convergence of a sequence of a probability distributions. $(\mathbb{P}_t)_{t\in\mathbb{N}}$ converges if and only if there is a distribution $\mathbb{P}_\infty$ such that $\rho(\mathbb{P}_t, \mathbb{P}_\infty) \to 0$, meaning that what we defined a few lines above will come down to the definition of ρ.

In their paper, M. Arjovsky, S. Chintala and L. Bottou introduce a measure based on optimal transport theory : the Wasserstein distance. Intuitively, if each distribution is viewed as a unit of mass of earth stacked on a metric space, the measure is the minimum cost of turning one pile into the other. This is supposed to be the product between the mass of earth that needs to be moved and the distance it has to be moved.
This is why the Wasserstein distance is also known as the Earth Mover's distance.

The Wasserstein distance

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma\in\Pi(\mathbb{P}_r,\mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[c(x - y)\right] \tag{W}$$

where c is a cost function - the Euclidian distance in the original paper [1], $\Pi(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joints distributions $\gamma(x, y)$ with marginals $\mathbb{P}_r$ and $\mathbb{P}_g$

2.1.2 Why choose the Wasserstein distance?

The motivation behind the choice of the Wasserstein distance as a cost function has roots in the very desirable properties that (W) has :

1. Under mild assumption over g_θ the Wasserstein distance has guarantees of continuity and differentiability
2. The Wasserstein distance induces a weaker topology than some famous traditional distances such as the Total Variation or the Kullback-Leibler divergence i.e it makes it easier for a sequence of distribution to converge

Those properties are especially useful notably when the spaces at hand are of low dimensions.
This makes the Wasserstein distance a great and sound choice in that setup.

Now that the choice is made, the question is how to optimize the Wasserstein distance given that (W) is intractable?
The authors introduce the dual formulation of the optimal transport problem through the Kantorovich-Rubinstein duality which states that :

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

where the supremum is over all the 1-Lipschitz functions f
Indeed, under the same mild assumptions mentioned earlier over g_θ, (W-KRd) admits a solution - the term becomes a $\max_{||f||_L \leq 1}$ - and :

$$\nabla_\theta W(\mathbb{P}_r, \mathbb{P}_\theta) = -\mathbb{E}_{z\sim p(z)}[\nabla_\theta f(g_\theta(z))]$$

2.2 Wasserstein GAN

2.2.1 The algorithm - Optimal Transport Dual approximation

Now let us recall the GAN architecture introduced in part 1.2, the GAN's field contribution of M. Arjovsky, S. Chintala and L. Bottou is to propose an alternative GAN based on an alternative way to train the generator to ultimately better approximate $\mathbb{P}_r$.
Indeed, recalling the approach introduced in 2.1, training the generator should result in minimizing a distance. That is why in the Wasserstein GAN, generator updates now centers around the use of a *critic* that outputs a quality score of a distance approximation rather than using strong discriminator-made classification.

More precisely, coming back to the remarks made at the end of 2.1.2, the focus is now on finding an approximation for the function f solving (W-KRd).
(W-KRd) is generally still intractable but the authors introduce a neural network which gives a sound approximation. The net has weights w lying in a compact space $\mathcal{W}$ enforced by weights clipping - this is made to ensure that the Lipschitz condition on *critic* functions $(f_w)_{w\in\mathcal{W}}$ is met - and then backprop through $\mathbb{E}_{z\sim p(z)}[\nabla_\theta f_w(g_\theta(z))]$

The Wasserstein GAN received a lot of attention for its stability but also its properties of convergence - the loss of the algorithm is an approximation of the Wasserstein distance up to a constant, meaning that the loss is directly linked to the quality of the generated samples.
However, in paper [4], T. Salimans and H. Zhang, A. Radford and D. Metaxas note that these GANs are not able still to exactly solve this optimal transport problem as the *critic* optimization is only an approximation and the Lipschitz condition roughly enforced.
In this context, alternatives coming back to the optimal transport primal have been introduced. We shall review some of them in part 2.3

2.3 Alternatives - Reverting back to the Optimal Transport Primal

In paper [1], the authors, noting that the optimal transport primal is highly intractable, makes use of neural network approximation of the dual. Here we are reviewing some alternatives leveraging the primal.

2.3.1 Sinkhorn AutoDiff - [2]

A. Genevay, G. Peyre and M. Cuturi in paper [2], come back to (W) and introduce an entropically smoothed generalization of the Wasserstein distance called the Sinkhorn distance.

The Sinkhorn distance

$$D_{\text{Sinkhorn}}\left(\mathbb{P}_r, \mathbb{P}_g\right) = \inf_{\gamma \in \Pi_\beta(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\gamma}\left[c(\mathbf{x},\mathbf{y})\right] \tag{S}$$

where c is a cost function, Π_β is now restricted to distributions with entropy of at least β

The authors approximate this distance by swapping the initial function by an expectation over mini-batches of data $\mathbf{X}$, $\mathbf{Y}$ consisting of K data vectors $\mathbf{x}, \mathbf{y}$.

The cost function c produces a transport cost matrix C of size $K \times K$ where the matrix coefficients $C_{i,j} = c\left(\mathbf{x}_i, \mathbf{y}_j\right)$ indicates the cost of transporting $\mathbf{x}_i$ in mini-batch $\mathbf{X}$ to $\mathbf{y}_j$ in mini-batch $\mathbf{Y}$.
Thus, the joint distribution γ amounts to a matrix M of size $K \times K$ of matchings between the i, j elements. This matrix has all positive coefficients, rows and columns summing to one and has sufficient entropy[2].

Eventually, this yields the mini-batch Sinkhorn distance :

$$\mathcal{W}_c(X, Y) = \inf_{M \in \mathcal{M}} \operatorname{Tr}\left[MC^{\mathrm{T}}\right] \tag{MBS}$$

where c is added to stress the importance of the cost function choice for this distance.

This distance - being tractable - allows a very stable training which is very desirable in a GAN context but working on mini-batch, the gradients of (MBS) are not unbiased estimator of our initial optimal transport program (S) which means that this distance expectation is no longer a valid distance over our probability distributions.

2.3.2 Cramer GAN - [3]

As a workaround to the disadvantage that (MBS) is suffering from, Bellemare et al. introduced the Cramer distance also known as the Energy Distance.

The Cramer distance

$$D_{\text{ED}}(p, g) = \sqrt{2\mathbb{E}[\|\mathbf{x} - \mathbf{y}\|] - \mathbb{E}\left[\|\mathbf{x} - \mathbf{x}'\|\right] - \mathbb{E}\left[\|\mathbf{y} - \mathbf{y}'\|\right]} \tag{ED}$$

where $\mathbf{x}, \mathbf{x}'$ are independent samples from data distribution $\mathbb{P}_r$ and $\mathbf{y}, \mathbf{y}'$ independent samples from the generator dsitribution $\mathbb{P}_\theta$.

This review of two alternatives to the Wasserstein GAN leveraging the primal optimal transport problem allows us to introduce the OT-GAN in section 3, often seen as a synthesis of both the Sinkhorn AutoDiff and the Cramer GAN.

[2] More details on this procedure can be found in paper [2]

3 OT-GAN - [4]

3.1 Introducing the Mini-Batch Energy Distance

As introduced in 2.1, most of what we presented until now is essentially focused on minimizing a distance between two probability distributions. However, the intuition of working with mini batches as in part 2.3.1 was really insightful as working with mini batches is a common practice in Deep Learning and recent work on GANs leveraging the mini batch approach has often yielded better results than a sample-per-sample approach.
This is especially true on computer vision tasks which is our use case of interest throughout the project.

This statement is the starting point of the intuition that T. Salimans, H. Zhang, A. Radford and D. Metaxas had in paper [4]. The idea is to start from the primal form of the optimal transport problem with the Engergy Distance (ED) as in 2.3.2 and use it over mini batches as in 2.3.1.

Instead of using the Euclidean distance in (ED), the authors propose to use the Sinkhorn distance (MBS) in its mini-batch version - hence, the alleged synthesis between both previous approaches.

The Minibacth Energy Distance

$$D_{\text{MED}}(p, g) = \sqrt{2\mathbb{E}\left[\mathcal{W}_c(\mathbf{X}, \mathbf{Y})\right] - \mathbb{E}\left[\mathcal{W}_c\left(\mathbf{X}, \mathbf{X}'\right)\right] - \mathbb{E}\left[\mathcal{W}_c\left(\mathbf{Y}, \mathbf{Y}'\right)\right]} \tag{M-ED}$$

where $\mathbf{X}, \mathbf{X}'$ are independently sampled mini-batches from distribution $\mathbb{P}_r$ and $\mathbf{Y}, \mathbf{Y}'$ are independent mini-batches from $\mathbb{P}_\theta$

To dig a bit deeper into the quirks and features of the OT-GAN, it is interesting to discuss a notion we deliberately avoided earlier, namely is this expression (M-ED) a metric?

We need to check that the distance's image is in $\mathbb{R}^+$, that the triangle inequality is verified, a symmetry property and the identity of indiscernibles i.e $d(x, y) = 0 \iff x = y$

Those conditions are easily met when the distance used with the generalized (ED) instead of the Euclidean distance is a metric itself.
Thus, focus is now on the mini-batch version of the Sinkhorn we chose to use. As noted in 2.3.1, (MBS) is not a valid metric over probability distributions but this distance is indeed valid between mini-batches.

Hence, unlike the initial (MBS), the (M-ED) is a valid metric by leveraging the Energy Distance in a generalized form.
It takes away the best of both world. Indeed, what is interesting there is that it leverages the statistical consistency of the Energy Distance but add the primal form of the optimal transport problem through the mini-batch version of the Sinkhorn distance.

This will make the OT-GAN very stable and highly competitive on various image generation tasks.
Now that we have better appreciation of the OT-GAN framework let us implementing it.

3.2 OT-GAN implementation

3.2.1 The algorithm

Now, the only thing missing for our implementation is the cost function c.
Authors found that fixed functions gave disappointing results and proposed a scheme where the cost function is learnt adversarially to better discriminate.
For our implementation, we use the cosine distance i.e

$$c_\eta(\mathbf{x}, \mathbf{y}) = 1 - \frac{v_\eta(\mathbf{x}) \cdot v_\eta(\mathbf{y})}{\|v_\eta(\mathbf{x})\|_2 \|v_\eta(\mathbf{y})\|_2}$$

where v_η is the neural net mapping the images into the mini-batch into a learned latent space.

3.2.2 Discriminator

Regarding the discriminator architecture we decided to implement a network with 4 convolutional layers. At each layer it divides the output size by 2. Each convolutional layer is followed by a LeakyReLU activation function. At the end the last output of latten and a L2 normalization layer is applied.

3.2.3 Generator

The generator takes as an input a vector of size 2048 (to match the output of the discriminator). This output goes through 5 convolutional transpose layers that double the output size at each step. The final output is of size 64 x 64 which is precisely the size of our images.

3.2.4 Training

Now that the framework is set up, we went on with the training. We trained the OT-GAN with an Adam optimizer with a learning rate of 10^{-3} and a $\beta_1 = 0.9$, $\beta_2 = 0.999$.
We chose $n_{\text{gen}} = 3$ for the number of iterations of the generator per critic iteration. We drew η_0 initial critic parameters from a uniform distribution, same goes for θ_0, the initial generator parameters.

Here is the pseudo-code we used :

OT-GAN training

for $t = 1$ to N **do**
Sample $\mathbf{X}, \mathbf{X}'$ two independent mini-batches from real data, and $\mathbf{Y}, \mathbf{Y}'$ two independent mini-batches from the generated samples
  $\mathcal{L} = \mathcal{W}_c(\mathbf{X}, \mathbf{Y}) + \mathcal{W}_c(\mathbf{X}, \mathbf{Y}') + \mathcal{W}_c(\mathbf{X}', \mathbf{Y}) + \mathcal{W}_c(\mathbf{X}', \mathbf{Y}') - 2\mathcal{W}_c(\mathbf{X}, \mathbf{X}') - 2\mathcal{W}_c(\mathbf{Y}, \mathbf{Y}')$
  if $t \bmod n_{gen} + 1 = 0$ **then**
  $\eta \leftarrow \eta + \alpha \cdot \nabla_\eta \mathcal{L}$
  else
  $\theta \leftarrow \theta - \alpha \cdot \nabla_\theta \mathcal{L}$
end if
end for

3.2.5 Results

Figure 2 page 9 shows how the OTGAN generator is learning to generate the MNIST images.
We can see that the generator produces some shapes. Even though these are not numbers this is an encouraging result. We are confident that with more time and more computational power we can reach an equilibrium between the generator and the critic that would eventually led to the re-production of MNIST numbers.

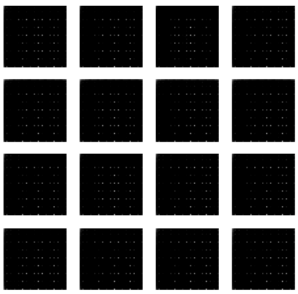
References

[1] Martin Arjovsky, Soumith Chintala and Léon Bottou. Wasserstein GAN. In *International conference on machine learning*, 2017.

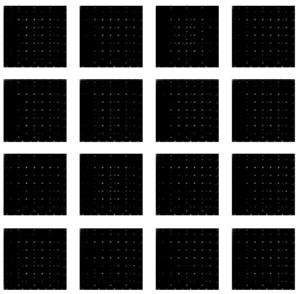
[2] Aude Genevay, Gabriel Peyre and Marco Cuturi. Learning Generative Models with Sinkhorn Divergences. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, 2018

[3] Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer and Remi Munos. The cramer distance as a solution to biased wasserstein gradients. In *arXiv 1705.10743*, 2017

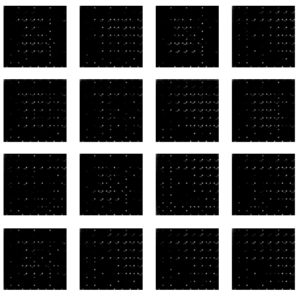
[4] Tim Salimans, Han Zhang, Alec Radford and Dimitris Metaxas. Improving GANs Using Optimal Transport. In *International Conference on Learning Representations*, 2018



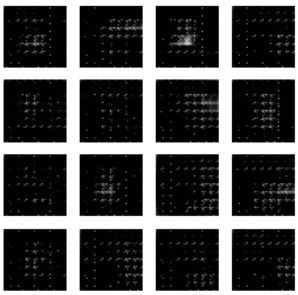
(a) Epoch 1



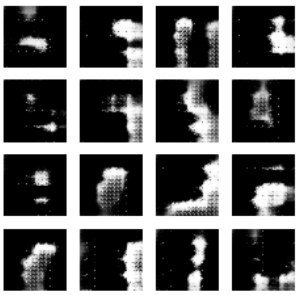
(b) Epoch 1



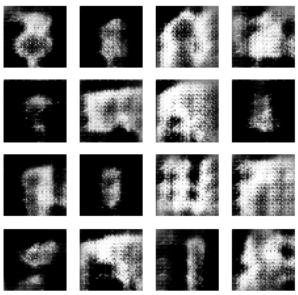
(c) Epoch 2



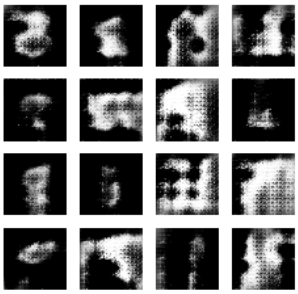
(d) Epoch 2



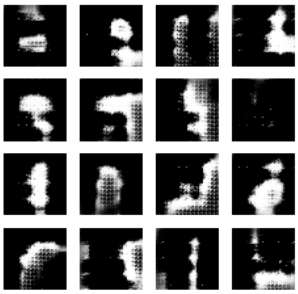
(e) Epoch 3



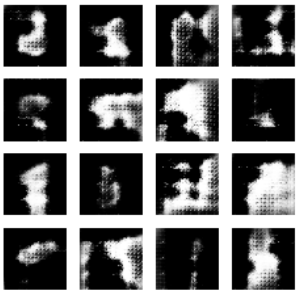
(f) Epoch 3



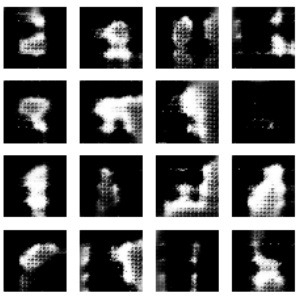
(g) Epoch 3



(h) Epoch 4



(i) Epoch 4



(j) Epoch 5

Figure 2: Evolution of the OTGAN through training