Improving GANs using Optimal Transport
Optimal Transport Project Report

Amine TAZI amine.tazi@ensae.fr
Hamdi BEL HADJ HASSINE hamdi.belhadjhassine@ensae.fr

May 4, 2022

1 Introduction

Thanks to the recent advances in Deep Learning, image generation has gained significant traction in the last decade and has become a booming area of research. One of the milestones of image generation was the advent of GANs, proposed by Y. Goodfellow et al. in 2014 [1]. GANs are described by Yann LeCun as "the most interesting idea in the last 10 years in Machine Learning", and the accessibility of massive amounts of data and computing power, especially on GPUs, has fueled the development of many variants of GANs achieving new state-of-the-art performance every year.

Although the predominant use of GANs is image generation, they can be generally used for most data generation tasks including text and sound, and they have many applications such as data augmentation, image compression and representation learning.

One drawback of GANs is that their training is notoriously hard and prone to mode collapse (generating identical or similar samples instead of diverse samples) and vanishing gradients (the discriminator becoming too strong and not giving feedback to the generator). To overcome those issues, many solutions have been proposed and in this work we ill focus on the solution provided by Salimans et al. [2] which applies Optimal Transport to define a new metric measuring the distance between the generator distribution and the data distribution. This metric, called mini-batch energy distance, effectively captures the distance between the two distributions and allows the authors to achieve state-of-the-art results on popular benchmarks for image generation.

2 Optimal Transport for Generative modeling with GANs

2.1 Previous works

In their basic form, GANs consist in two neural networks with a symmetric architecture : the generator and the discriminator. The generator is trained to trick the discriminator by generating data having a distribution as close as possible to that of real data, while the discriminator is trained to distinguish the synthetic outputs of the generator and real data.

The generator takes as input random noise vectors z and using a neural network transforms them into images $y = g(z)$, while the discriminator takes an image drawn from either the generated data (we denote such images y) or the original data distribution p (we denote such images x) and outputs a score classifying it as real or generated. The training loss is defined such as the discriminator is rewarded for giving a higher score to the correct classification, and the generator is rewarded for fooling the discriminator and generating images following a similar distribution

to the real data. The goal of training is then to find a pair of (g, d) for which this game is at a Nash equilibrium. The generator's loss can then be written: $L_g = \sup_d \mathbb{E}_{x\sim p} \log[d(x)] + \mathbb{E}_{y\sim g} \log[1 - d(y)]$

This problem can be reframed using Optimal Transport by writing the target to minimize as the Earth-Mover distance (or Wasserstein-1 distance) between the distributions of x and y:

$$D_{\text{EMD}}(p, g) = \inf_{\gamma \in \Pi(p,g)} \mathbb{E}_{x,y\sim\gamma} c(\mathbf{x}, \mathbf{y})$$

where $\Pi(p, g)$ is the set of all joint distributions $\gamma(x, y)$ with marginals $p(x)$, $g(y)$, and where $c(x, y)$ is a cost function (for example Euclidean distance). The idea is to minimize this distance so that the generated samples follow a similar distribution to that of the original data.

Computing this distance is however intractable in practice. Some previous works have replaced it with its dual formulation which can be approximated relatively well empirically if we assume the discriminator is 1-Lipschitz, but the authors argue that such an approximation is very rough. Another method used by Genevay et al. [3] is to approximate the EMD using the Sinkhorn Distance instead [4]:

$$D_{\text{Sinkhorn}}(p, g) = \inf_{\gamma \in \Pi_\beta(p,g)} \mathbb{E}_{\mathbf{x},\mathbf{y}\sim\gamma} c(\mathbf{x}, \mathbf{y})$$

where Π_β denotes joint distributions with entropy $\geq \beta$. To compute this distance, they use mini-batches X, Y consisting of K data vectors x, y each. They first compute the cost matrix C where $C_{i,j} = c(x_i, y_j)$, and they define the set of possible soft matchings matrices as $\{M \in \mathbb{R}^{K\times K}, M_{i,j} \geq 0 \,\forall\, 0 \leq i, j \leq K, \sum_i M_{i,j} = \sum_j M_{i,j} = 1, -\operatorname{Tr}\left[M \log\left(M^{\mathrm{T}}\right)\right] \geq \alpha\}$ which defines the coupling distributions γ. Then the distance between the distributions p and g on the mini-batches X, Y can be approximated as:

$$\mathcal{W}_c(X, Y) = \inf_{M\in\mathcal{M}} \operatorname{Tr}\left[MC^{\mathrm{T}}\right] \tag{1}$$

which can be implemented in a tractable and scalable way using the Sinkhorn algorithm. However the authors signal that the problem of using mini-batches in this method is that it makes the estimator of the Sinkhorn Distance between the the distributions p and g biased.

Another GAN alternative proposed by Bellemare et al. [5] is to use the Energy Distance:

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

where $x, x' \sim p$ and $y, y' \sim g$ iid. This method doesn't suffer from the bias problem, but doesn't benefit from the advantages of using optimal transport to better represent the distance between the two distributions of original and generated data.

To close this gap, the authors propose a new distance function that combines the last 2 methods while avoiding their drawbacks.

2.2 Mini-batch energy distance

The authors propose the Mini-batch Energy Distance as a new distance for comparing mini-batch distributions. For independent mini-batches $\mathbf{X}, \mathbf{X'} \sim p$, $\mathbf{Y}, \mathbf{Y'} \sim g$ this distance is defined as:

$$D^2_{\text{MED}}(p, g) = 2\mathbb{E}\left[\mathcal{W}_c(\mathbf{X}, \mathbf{Y})\right] - \mathbb{E}\left[\mathcal{W}_c\left(\mathbf{X}, \mathbf{X}'\right)\right] - \mathbb{E}\left[\mathcal{W}_c\left(\mathbf{Y}, \mathbf{Y}'\right)\right]$$

where $\mathcal{W}_c$ is the mini-batch Sinkhorn distance previously defined in Eq. 1

This distance is a metric in the mathematical sense, and according to the authors, it is a highly discriminative distance function with statistically unbiased mini-batch gradients thanks to the $(\mathbf{X}, \mathbf{X}')$ and $(\mathbf{Y}, \mathbf{Y}')$ terms.

3 OT-GAN

The mini-batch energy distance is defined with respect to a given cost function. The authors argue that simple cost functions like the Euclidean norm do not work well in high-dimensional spaces and instead choose to learn the cost function adversarially. To do this, they use a "critic" neural network (analogous to the discriminator) which embeds the images into a latent space extracting their features. The transport cost between the embeddings of two images is then defined using the cosine similarity:

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

where $v_\eta(\mathbf{x})$ is the embedding of an image $\mathbf{x}$, and the parameters η of the critic neural network are trained to maximize the minibatch energy distance (while the generator is trained to minimize it).

The authors use alternating gradient descent to train the generator and critic neural networks, with more frequent gradient updates for the generator than the critic in order to regularize the cost function and make it more stable and less likely to degenerate.

The complete OT-GAN algorithm and architecture are presented below. First, two mini-batches are sampled from the dataset and two others are generated using the generator. These batches are then embedded to the feature space using the critic. Their embeddings are used to calculate the cost matrix. After that, we evaluate the soft-matching matrix using Sinkhorn's divergence, which allows us to compute the loss. Finally, the gradient of the loss is used to update either the weights of the generator or the critic neural net.

Algorithm 1 Optimal Transport GAN (OT-GAN) training algorithm with step size α, using mini-batch SGD for simplicity

Require: n_{gen}, the number of iterations of the generator per critic iteration
Require: η_0, initial critic parameters. θ_0, initial generator parameters
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

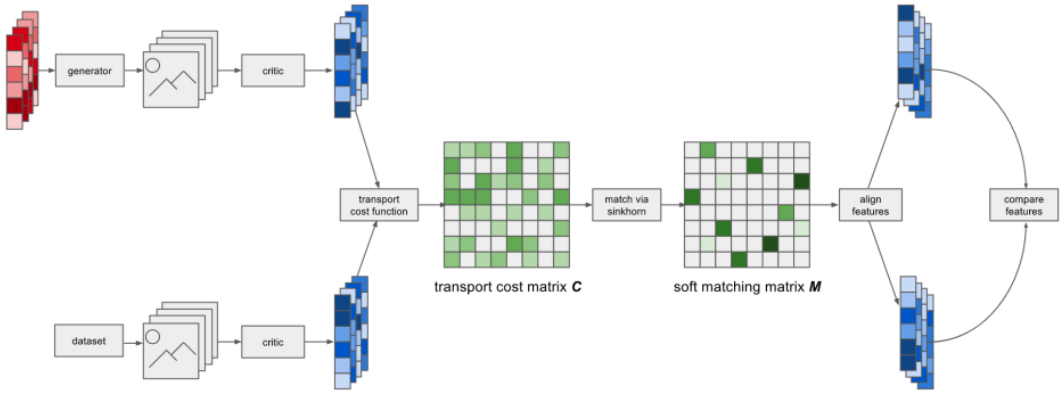

Figure 1: OT-GAN architecture and data processing [2]

Algorithm 1 Sinkhorn's divergence Algorithm

Require: $m, n \geq 0$ dimensions
Require: $k \geq 0$ number of iterations
Require: C Cost matrix, γ entropy regularization
$u \leftarrow (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n}) \in \mathbb{R}^n$
$M = e^{\frac{-C}{\gamma}}$
$a \in \mathbb{R}^n, b \in \mathbb{R}^m$ Probability vectors
for $i = 0, 1, ..., k$ **do**
$v = \frac{b}{K^T u}$
$u = \frac{a}{Kv}$
end for

4 Implementation and experiments

4.1 Paper's experiments

The paper's authors conducted experiments on several datasets and with different configurations to validate the proposed OT-GAN. In the first experiment they train it on 2D data from a gaussian mixture distribution, and show that even if the critic's training is stopped, the generator still converges to the true distribution of the data, unlike the original GAN model which observes a mode collapse (generated data only covers a small part of the distribution).

In the second experiment the authors train OT-GAN on the CIFAR-10 dataset. To show that OT-GAN's training is stable, they use simple convnet architectures for the generator and the critic without batch normalization, layer normalization, or other stabilizing additions. The OT-GAN model achieves an inception score of 8.47, outperforming all other models tested (DCGAN, Improved GAN, Denoising FM, WGAN-GP). However, to achieve this performance, the authors used a large batch size (8000) which required 8 GPUs to train the model.

The third experiment is similar to the second but here the authors use the Imagenet Dogs dataset to generate higher resolution images. The model outperforms DCGAN both visually and in the inception score.

In the fourth experiment, the authors test OT-GAN on the CUBS dataset for text-to-image generation and it outperforms the other models tested. Visually the generated images are very realistic.

4.2 Our Implementation

We aimed to implement OT-GAN from scratch as described in the paper. We chose to train and test it on the MNIST dataset, which made the training faster and less resource-consuming than if we had used other high-resolution datasets, while still remaining challenging. Unlike the paper's experiments, we only used 1 GPU for training and wanted to evaluate OT-GAN's performance in such case. Our GPU has 12 GB of memory which is equal to Colab's K80 GPUs.

At first, we tried to find a suitable model architecture for both our generator and critic. This was not easy for different reasons : first of all, since adversarial training is generally unstable, even if we did find a good architecture we were not assured that training it would yield good results in our run.

The second reason was that our loss was dependent on the critic. As we also trained the critic, our loss function was not the same over the epochs. This meant that we could not rely on the loss to keep track of the evolution of our model, and we usually had to wait until our model was fully trained to test our model visually by trying to generate images. The value of the loss also depends on the critic, so different architectures cannot be directly compared using the loss values.

The final issue we faced was that our model had numerous hyperparameters, such as learning rate, batch size, entropy regularization and number of Sinkhorn's divergence steps ; and also numerous parameters, for instance the number of output channels in our convolution layers. At first, we tried to tune these parameters using random search while testing different model architectures, but when a model performed poorly, it was difficult to know if it happened because of the model's architecture or because of the hyperparameters. Since we had limited computational power, we decided to set reasonable values for the hyperparameters and to mainly focus on looking for a functional model architecture.

After many trials, what we found to work best was to have a generator with a single linear layer followed by 4 convolutional layers, and a critic with 2 stacked fully-connected layers. After we adjusted the number of neurons and output channels in each layer to match the complexity of our training dataset, we managed to obtain good performance: As shown in Fig. 2, most of the generated images represent recognizable digits and many of them are hard to distinguish from actual MNIST images, although some of them still present artifacts. We also see that the generated images are diverse in their digit numbers and in their shapes, meaning that our model does not suffer from mode collapse. Moreover, the performance of our final model was relatively stable during training or when retrained, and we didn't need to use batch normalization or other regularization tricks often used to train GANs.

In our experiments we observed that increasing the batch size improves the quality of the generated images, so we used a batch size of 500, which is relatively large but fits in our single GPU. With this batch size, we still managed to obtain satisfying results, which shows that we don't need 8 GPUs like done in the paper to train OT-GAN.

We also tried training our model on a subset of MNIST with a single label. In this case we can observe in Fig. 3 an improvement in the quality of the generated images.

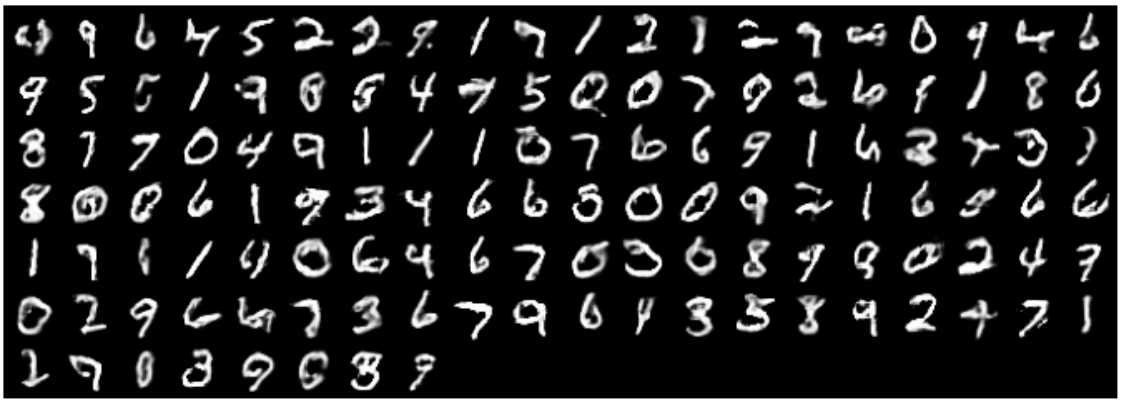

Figure 2: Images generated by our final model

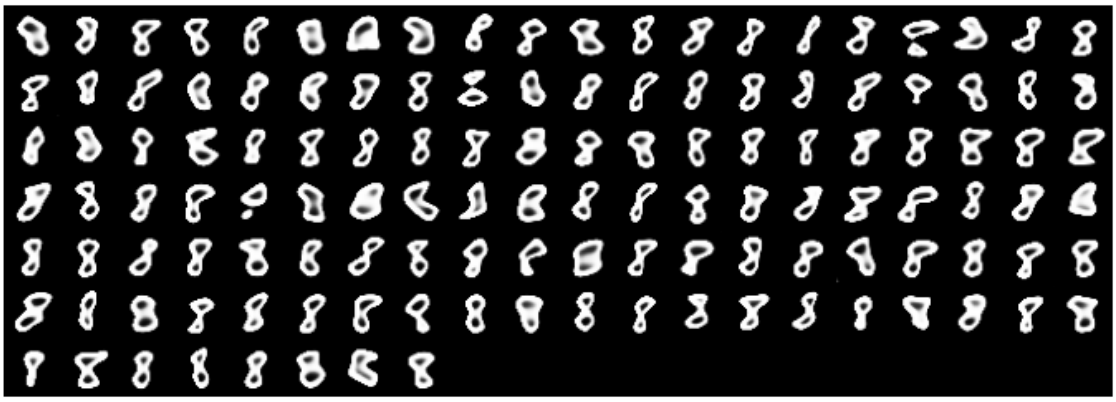

Figure 3: Images generated by a model trained on a single label of MNIST (8)

In addition, we performed an ablation experiment where we train a model with the same architecture but we use the classic Cross-entropy loss instead of the loss induced by the mini-batch energy distance. We observed that such model performed very poorly compared with OT-GAN, which stresses the importance of using Optimal Transport to define an unbiased and discriminative loss.

5 Conclusion

In this project we had the opportunity to apply Optimal Transport to improve generative modeling with GANs by using the Mini-batch energy distance, which relies on a batched version of the Sinkhorn Distance and produces an unbiased metric to compare the distributions of the original and the generated data. This allows us to compute the model's loss and update its weights so as to reduce this loss. Using optimal transport instead of simple metrics like the Euclidean distance makes the model more robust and less prone to mode collapse or vanishing gradients. We can also conclude from our experiments that using a large batch size improves the performance of OT-GAN, but a batch size of 500 yields good performance already and can be done using a single GPU.

References

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661

[2] T. Salimans, H. Zhang, A. Radford, and D. Metaxas, "Improving GANs using optimal transport," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rkQkBnJAb

[3] A. Genevay, G. Peyré, and M. Cuturi, "Learning generative models with sinkhorn divergences," 2017. [Online]. Available: https://arxiv.org/abs/1706.00292

[4] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in Neural Information Processing Systems*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013. [Online]. Available: https://proceedings.neurips.cc/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf

[5] M. G. Bellemare, I. Danihelka, W. Dabney, S. Mohamed, B. Lakshminarayanan, S. Hoyer, and R. Munos, "The cramer distance as a solution to biased wasserstein gradients," *CoRR*, vol. abs/1705.10743, 2017. [Online]. Available: http://arxiv.org/abs/1705.10743