Generative modeling project : Neural Optimal Transport

Paul Barbier[1] **and Bastien Le Chenadec**[1]

[1]École des Ponts ParisTech, Master MVA

1 Introduction

Optimal Transport (OT) is a mathematical framework that aims to find the most efficient way to transport a distribution of mass to another. This framework has been used extensively in the context of generative models, for instance as a loss function in the training of Generative Adversarial Networks (GANs) or by learning a mapping between two distributions. In this project, we aim to study the paper "Neural Optimal Transport" (Korotin, 2023) [1] which introduces an algorithm to train a neural network to learn the optimal transport between two distributions, with applications in image generation. We will introduce some optimal transport problems in various formulations, but we will focus on giving a general overview and some details and hypotheses will be omitted for the sake of clarity.

2 Background on Optimal Transport

2.1 Optimal Transport Problem

Let μ and ν be two probability distributions on $\mathcal{X}$ and $\mathcal{Y}$ respectively (typically $\mathcal{X}, \mathcal{Y} = \mathbb{R}^n, \mathbb{R}^m$). To give a meaning to "efficiently" transporting mass, we define a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that quantifies the cost of transporting a unit of mass in $\mathcal{X}$ to one in $\mathcal{Y}$. The (Monge) optimal transport problem consists in finding a **transport map** $T^* : \mathcal{X} \to \mathcal{Y}$ such that :

$$T^* \in \underset{T\#\mu=\nu}{\text{Argmin}} \int_{\mathcal{X}} c(x, T(x)) d\mu(x) \qquad \text{Cost}(\mu, \nu) = \int_{\mathcal{X}} c(x, T^*(x)) d\mu(x) \tag{1}$$

where $T\#\mu$ is the pushforward distribution of μ by T, defined by $(T\#\mu)(A) = \mu(T^{-1}(A))$ for any measurable set $A \subset \mathcal{Y}$. This formulation calls for a deterministic mapping from $\mathcal{X}$ to $\mathcal{Y}$, which is not always desirable or feasible under general assumptions. Kantorovich introduced a relaxed OT problem that aims at finding a **transport plan** $\pi^* \in \Pi(\mu, \nu)$ in the set of joint distributions on $\mathcal{X} \times \mathcal{Y}$ with marginals μ and ν such that :

$$\pi^* \in \underset{\pi\in\Pi(\mu,\nu)}{\text{Argmin}} \int_{\mathcal{X}\times\mathcal{Y}} c(x, y) d\pi(x, y) \qquad \text{Cost}(\mu, \nu) = \int_{\mathcal{X}\times\mathcal{Y}} c(x, y) d\pi^*(x, y) \tag{2}$$

In general the solution to the Kantorovich problem is stochastic, but in some cases it may be deterministic in which case it is also a solution to the Monge problem. Building on this idea of stochasticity in the solution, weak OT was introduced as a generalization of the Kantorovich problem, where the cost function is of the form $C : \mathcal{X} \times \mathcal{P}(\mathcal{Y}) \to \mathbb{R}$. In this case the weak OT problem writes :

$$\pi^* \in \underset{\pi\in\Pi(\mu,\nu)}{\text{Argmin}} \int_{\mathcal{X}} C(x, \pi(\cdot|x)) d\pi(x) \qquad \text{Cost}(\mu, \nu) = \int_{\mathcal{X}} C(x, \pi^*(\cdot|x)) d\pi^*(x) \tag{3}$$

where $\pi(\cdot|x)$ is the conditional distribution of π given x and $d\pi(x)$ is the marginal distribution of π on $\mathcal{X}$ (which is actually μ).

2.2 Weak OT duality

For weak OT cost C, and f defined on $\mathcal{Y}$ sufficiently regular, the weak C-transform of f is defined on $\mathcal{X}$ as :

$$f^C(x) = \inf_{\rho\in\mathcal{P}(\mathcal{Y})} \left\{ C(x, \rho) - \int_{\mathcal{Y}} f(y) d\rho(y) \right\} \tag{4}$$

The dual form of the weak OT problem is then :

$$f^* \in \underset{f}{\text{Argmax}} \int_{\mathcal{X}} f^C(x) d\mu(x) + \int_{\mathcal{Y}} f(y) d\nu(y) \qquad \text{Cost}(\mu, \nu) = \int_{\mathcal{X}} f^{*C}(x) d\mu(x) + \int_{\mathcal{Y}} f^*(y) d\nu(y) \tag{5}$$

3 Neural Optimal Transport

In (Korotin, 2023) [1], the authors aim to solve the weak OT problem with a neural network. First they reformulate the weak dual problem with noise outsourcing, then they introduce an algorithm to solve this problem.

3.1 Weak dual OT reformulation

From now on we consider $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subset \mathbb{R}^m$. We introduce $\mathcal{Z} \subset \mathbb{R}^d$ a latent space with a distribution $\rho \in \mathcal{P}(\mathcal{Z})$. The following result holds true under some basic assumptions on ρ :

$$f^C(x) = \inf_t \left\{ C(x, T\#\rho) - \int_{\mathcal{Z}} f(t(z))d\rho(z) \right\} \tag{6}$$

This result can be integrated against μ to replace the dual weak OT problem by a maximin problem :

$$\begin{aligned} \text{Cost}(\mu, \nu) &= \sup_f \inf_T \int_{\mathcal{Y}} f(y)d\nu(y) + \int_{\mathcal{X}} \left(C(x, T(x, \cdot)\#\rho) - \int_{\mathcal{Z}} f(T(x, z))d\rho(z) \right) d\mu(x) \\ &= \sup_f \inf_T \mathcal{L}(f, T) \end{aligned} \tag{7}$$

where $T : \mathcal{X} \times \mathcal{Z} \to \mathcal{Y}$. The proofs of these results are in the paper [1]. This reformulation can be interpreted in the following way : the solution to the weak OT problem is a **stochastic map** $T : \mathcal{X} \times \mathcal{Z} \to \mathcal{Y}$ that is split into two parts, a deterministic part $T(x, \cdot)$ and a stochastic part $z \mapsto T(x, z)$. $\mathcal{Z}$ is a latent space that models the randomness in the transport.

Solving (7) does not necessarily yield a solution T^* to the weak OT problem. However under sufficient convexity conditions on C the solution to the maximin problem is a solution to the weak OT problem. With that in mind the authors introduce an algorithm to solve this problem and restrict their attention to such cost functions.

3.2 SGAD

The authors prove that the space of neural networks is rich enough to approximate stochastic maps T which motivates the use of neural networks to solve (7). f and T in (7) are parametrized by neural networks f_ω and T_θ respectively. The authors use a stochastic gradient ascent descent (SGAD) algorithm to solve the problem. The algorithm is presented in Figure 1.

Algorithm 1 Stochastic Gradient Ascent Descent (SGAD) algorithm for Neural Optimal Transport

Input : distributions μ, ν, ρ accessible by samples, mapping network $T_\theta : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^d$, potential network $f_\omega : \mathbb{R}^n \to \mathbb{R}$, number of inner iterations K_T, (weak) cost $C : \mathcal{X} \times \mathcal{P}(\mathcal{Y}) \to \mathbb{R}$, empirical estimator $\widehat{C}(x, T(x, Z))$ for the cost
Output : learned stochastic OT map T_θ representing an OT plan between distributions μ, ν
repeat
  Sample batches $Y \sim \nu$, $X \sim \mu$, for each $x \in X$ sample batch $Z_x \sim \rho$
  $\mathcal{L}_f \leftarrow \frac{1}{|X|} \sum_{x \in X} \frac{1}{|Z_x|} \sum_{z \in Z_x} f_\omega(T_\theta(x, z)) - \frac{1}{|Y|} \sum_{y \in Y} f_\omega(y)$
  Update ω by using $\frac{\partial \mathcal{L}_f}{\partial \theta}$
  for $k_T = 1, 2, \ldots, K_T$ **do**
    Sample batch $X \sim \mu$, for each $x \in X$ sample batch $Z_x \sim \rho$
    $\mathcal{L}_T \leftarrow \frac{1}{|X|} \sum_{x \in X} \left[\widehat{C}(x, T_\theta(x, Z_x)) - \frac{1}{|Z_x|} \sum_{z \in Z_x} f_\omega(T_\theta(x, z)) \right]$
    Update θ by using $\frac{\partial \mathcal{L}_\theta}{\partial \theta}$
until converged

Figure 1: Neural Optimal Transport algorithm [1]

4 EXPERIMENTS

For this project, we conducted experiments using the code provided by the author. You can find our scripts at https://github.com/bastienlc/NOT. We've done a first experiment with generated synthetic data and a second one with a more realistic use-case using a large dataset.

4.1 Synthetic data

4.2 Real dataset

Authors made experiments using faces from animes and real persons and the results were funny so we decided to make a similar one with a different dataset. More precisely, we took the CelebA dataset [2] which comprises 200k images of celebrity faces with attribute annotations. For the second dataset, we chose CartoonSet100K, a dataset introduced in [3]. It's a large-scale dataset containing 100k of 2D generated cartoon avatar images.

For this large-scale experiment we used a cloud instance with a NVidia A100 GPU with 40GB of memory. We increased the batch size per dataset from 64 to 128 and we used 128×128 images in place of 64×64 to get generated images with a

Figure 2: Images samples from CartoonSet100K

nicer resolution. Increasing the batch size lead to better memory utilisation and quicker convergence as we'll see in the next paragraphs.

On the figure below, you can see how the loss function evolves throughout the training. Unusually, the loss function here is fluctuating around the constant value 0 and that's expected: explain why it's expected TODO.

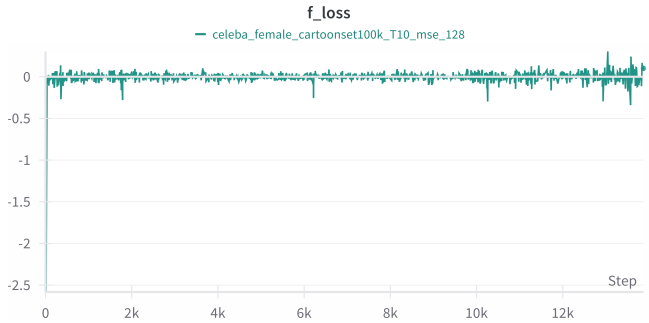
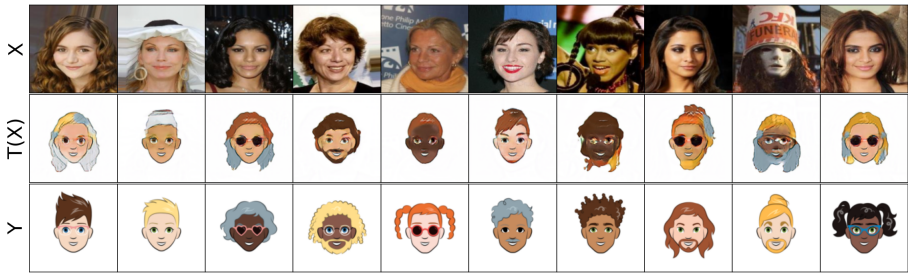
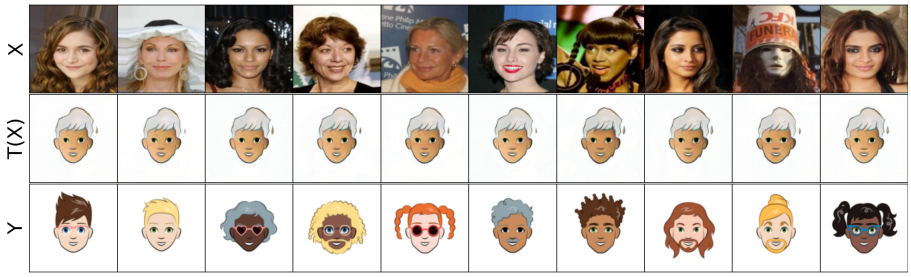


Figure 3: Loss function

(a) Iteration 1



(b) Iteration 5000



(c) Iteration 13800

Figure 4: Test images

Those experiments are quite long in time (about a day for 1̃5k iterations with the parameters given above) so we restrict ourselves to only train the strong formulation model and the results for a fixed sample of the test set are given in 4.

4.2.1 Observations

Figure 4a lets us see the model initialisation.

Furthermore, we witnessed realistic mappings of the real faces onto the cartoon avatar distribution around iteration 5000, see 4b. This is far less than what the authors observed in their experiments, this number was closer to 40k iterations for convergence. However, our images have an empty background so it might be easier in comparison with anime face images which contain a complex surrounding as they are extracted from animes. One can see that the color of the skin and the hair map well. However, the model struggles when there are glasses on the target image. It translates glasses into dark black holes. It's very surprising to see new avatars thanks to those mappings.

At this point, we thought we were far from the end and we decided to continue the training for a night. Unfortunately, few thousands iterations later mode collapses appeared with the same avatar being mapped for a given batch. Surprisingly, the constant predicted face is changing with training without changing the mode collapse.

5 CONCLUSION

Random sentence for the conclusion in the meantime.

REFERENCES

[1] Alexander Korotin, Daniil Selikhanovych, and Evgeny Burnaev. Neural optimal transport. *arXiv (Cornell University)*, 1 2022. doi: 10.48550/arxiv.2201.12220. URL https://arxiv.org/abs/2201.12220.

[2] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[3] Amelie Royer, Konstantinos Bousmalis, Stephan Gouws, Fred Bertsch, Inbar Mosseri, Forrester Cole, and Kevin Murphy. XGAN: unsupervised image-to-image translation for many-to-many mappings. *CoRR*, abs/1711.05139, 2017. URL http://arxiv.org/abs/1711.05139.

APPENDIX

A FIGURES