3D Neural Optimal Transport (Machine Learning 2023 Course)

Sergei Kholkin [1] **Artem Basharin** [1] **Anastasiia Batsheva** [1] **Maksim Bobrin** [1]

Abstract

In this paper, we explore the applicability of a neural network-based algorithm to compute optimal transport maps and plans for strong transport costs described in the article (Korotin et al., 2022). We evaluate the performance of this algorithm for the problem of unpaired object-to-object translation for 3-dimensional handwritten digits images generated from MNIST dataset (Deng, 2012) and show that optimal transport mapping preserves color.

Github repo: 3D Neural Optimal Transport
Presentation file: 3D Neural Optimal Transport

1. Introduction

Generative modeling involves the task of generating different modalities such as audio, video, text, image, from the empirical distribution of few training example. In today's world, these tasks are ubiquitous in areas such as biometric identification, speech recognition, medical diagnostics, and word processing.

The central problem of generative modeling is to train the model so that the distribution of the data it generates corresponds to the distribution of the training data. The best known solution method is generative adversarial networks (GANs), which consist of a generator and a discriminator that estimate the distance between the distributions of the generated and real data. Distance is often estimated using well-known metrics such as Kullback-Leibler divergence or Wasserstein distance.

An alternative approach to measuring distance and constructing a generative model is provided by optimal transport theory. In this paper, we investigate the applicability of this approach based on the method proposed in the article (Korotin et al., 2022). The authors propose a novel algorithm to compute deterministic and stochastic OT plans with deep neural networks. Their algorithm is designed for weak and strong optimal transport costs and generalizes previously known scalable approaches.

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Alexander Korotin <a.korotin@skoltech.ru>.

Final Projects of the Machine Learning 2023 Course, Skoltech, Moscow, Russian Federation, 2023.

2. Problem

Here we offer the reader a brief reminder of what optimal transport is. Of course, this field plays a major role in many scientific studies, but in order to understand further mathematical deductions and theoretical proofs, we want to first provide all the necessary terms and introduce the notation used afterwards.

2.1. Transport

Suppose we have two bunch of points A and B. And we want to **transport** (aka turn) A into B. In the easiest case we can just move each of A's point to one of B: $a_i \to b_j$

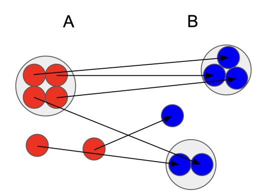

Figure 1. Transport between two point clouds.

2.2. Transport cost

Additionally we assume that every movement has a cost. To quantify this, let's say the transportation cost of moving one point from A to B is given by the L_2 distance:

$$c(a_i, b_j) = \|a_i - b_j\|_2^2$$

Then the total cost of the transport is defined as follows:

$$C(A, B) = \sum_{a_i \in A} \sum_{b_j \in B} c(a_i, b_j) \underbrace{T(a_i, b_j)}_{\text{plan}}$$

Here we use a special functional plan. In our simple example **transport plan** is jest a number of points we transport from

a_i to b_j. Plan function has several conditions where the most important ones are:

$$\sum_{b_j \in B} T(a^i, b^j) = w_a(a^i), \quad \sum_{a_i \in A} T(a^i, b^j) = w_b(b^j)$$

Here $w_a(a_i), w_b(b_j)$ - numbers of A's points in a_i and B's points in b_j respectively.

But generally optimal transport is about distributions movement. Here we suppose that the transport from an original distribution $p(x)$ to a new distribution $q(x)$ is needed.

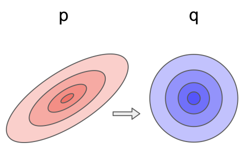

Figure 2. Transport between two distributions.

In such case we should change a little the transport plan's conditions:

$$\int T(x_p, x_q)\, dx_p = p(x_q), \quad \int T(x_p, x_q)\, dx_q = q(x_p)$$

and the total cost expression:

$$C(p, q) = \int\int c(x_p, x_q) T(x_p, x_q)\, dx_p\, dx_q$$

Then we come to the notion of the optimal transport as the transport that minimizes the total cost. It is this optimization problem that we will solve with the methods proposed in this article.

$$\boxed{\textbf{Optimal transport plan}: \arg\min_T C(p, q)}$$

3. Methods

3.1. Monge and Kantorovich OT

Discussion in the first section was informal, mostly providing intuition about Optimal transport theory. In current section, we introduce fundamental optimal transport formulations along with their motivations.

First, as was stated before, OT can be viewed as a map from two arbitrary probability measures. Precisely, given two measurable spaces $\mathcal{X}, Y$ with measures α, β respectively, OT finds map $T : \mathcal{X} \to \mathcal{Y}$, which minimizes

$$\min_T \{ \int_X c(x, T(x)) d\alpha(x) : T_\# \alpha = \beta \} \tag{1}$$

However, such formulation does not allow to redistribute masses from single point, or when one measure is atomless. Hence, Kantorovich formulation extends Monge problem by *mass-splitting*, and can be stated as:

$$\text{Cost}(\alpha, \beta) = \inf_{\pi \in \Pi(\alpha, \beta)} \int_{\mathcal{X} \times \mathcal{Y}} c(x, y) d\pi(x, y) \tag{2}$$

In order to solve the above equation efficiently, one can view such optimization task in its dual formulation. Also, other computational tricks can be used, e.g entropy regularization, which leads to *Sinkhorn* algorithm.

3.2. Neural Optimal Transport

The idea of learning optimal transportation plans via Deep Neural Networks was proposed in (Korotin et al., 2022). From now, $\mathbb{P} \in \mathcal{P}(\mathcal{X})$ and $\mathbb{Q} \in \mathcal{P}(\mathcal{Y})$. Authors proposed generalization of Kantorovich OT (strong OT) by modifying objective using additional stochasticity in resulting distribution, namely:

$$\text{Cost}(\mathbb{P}, \mathbb{Q}) = \inf_{\pi \in \Pi(\mathbb{P}, \mathbb{Q})} \int_{\mathcal{X}} C(x, \pi(\cdot|x)) d\pi(x), \tag{3}$$

where $\pi(\cdot|x)$ is conditional distribution, taking input from input distribution and $C : \mathcal{X} \times \mathcal{P}(\mathcal{Y}) \to \mathbb{R}$ called *weak* cost, which in case of Euclidean domains, can be W_2 distance:

$$C(x, \mu) = \frac{1}{2} \int_{\mathcal{Y}} ||x - y||_2^2 d\mu(u) - \frac{\gamma}{2} \text{Var}(\mu) \tag{4}$$

Formulation of (3) is stated in primal form, and can be efficiently solved using principle of duality. Derivations can be found in original paper, here we state final objective, which will be used in our contribution:

$$\begin{aligned} \text{Cost}(\mathbb{P}, \mathbb{Q}) = \sup_f \inf_{\pi \in \Pi(\mathbb{P}, \mathbb{Q})} [& \int_{\mathcal{X}} C(x, \pi(\cdot|x)) d\mathbb{P}(x) \\ & - \int_{\mathbb{X}} (\int_{\mathbb{Y}} f(y) d\pi(y|x) d\mathbb{P}(x)) \\ & + \int_{\mathbb{Y}} f(y) d\mathbb{Q}(y)] \end{aligned}$$

In the above equation f and π can be substituted via learnable functions, for example, neural networks as follows:

$$\begin{aligned} \text{Cost}(\mathbb{P}, \mathbb{Q}) = \sup_f \inf_T & \int_X C(x, T_x \# S) d\mathbb{P}(x) \\ & - \int_X \int_{\mathcal{Z}} f(T_x(z)) d\mathbb{S}(z) d\mathbb{P}(x) \\ & + \int_{\mathcal{Y}} f(y) d\mathbb{Q}(y) \end{aligned}$$

3.3. OT meets GANs

One of the first approaches in image generation, which used OT cost as a proxy for optimization objective, was

Wasserstein GAN. In WGANs, two probability measures μ_0, μ_1 are taken and W_1 distance is taken between them:

$$W(\mu_0, \mu_1) = \sup_{||f|| \leq 1} \mathbb{E}_{x \sim \mu_0} f(x) - \mathbb{E}_{x \sim \mu_1} f(x)$$

With supremum taken over all Lipschitz continuous functions. In original paper it is shown that (where g_θ is Neural Network)

$$\nabla_\theta W(\mu_0, \mu_1) = -\mathbb{E}_{z \sim \rho}(\nabla_\theta f(g_\theta(z)))$$

However, Neural Optimal Transport drastically differs from WGAN as for optimization algorithm, as well for generative modelling formulation.

3.4. Extending to 3D

In original work, $f : \mathbb{R}^{3 \times W \times H} \to \mathbb{R}$ was a ResNet (He et al., 2015), while for the case of OT map $T : \mathbb{R}^{4 \times H \times W} \to \mathbb{R}^{3 \times W \times H}$ UNet architecture was used, where additional channel was added to incorporate noise. In our work, since we are dealing with 3D samples from MNIST 4.1, each convolutional transformation in UNet (Ronneberger et al., 2015) was replaced by its 3D counterpart. Since we are dealing with unpaired image to image translation, we are using weak optimal transport cost for the objective.

4. Experiments

4.1. Dataset preparation

3D colored samples from MNIST were generated using 2D MNIST dataset samples (Deng, 2012) by repeating them along new dimension and coloring into randomly chosen one of four colors: red, yellow, green, blue 4.1. Two experiments were carried out. Forward: distributions $\mathbb{P}$ and $\mathbb{Q}$ were represented by set of twos and fours respectively sampled from 2D MNIST. Backward: distributions $\mathbb{P}$ and $\mathbb{Q}$ were represented by set of fours and twos respectively sampled from 2D MNIST, transformed into 3D and colored. So we are trying to make an optimal transport map from the distribution of 3D twos into the distribution of 3D fours for the forward case and from the distribution of 3D fours into the distribution of 3D twos for the backward case. All the following experiment setup details were the same for both experiments

Algorithm 1 Generation algorithm

Input: 2D MNIST sample $X_{2D} \in \mathbb{Z}^{16 \times 16}[0, 255]$

$X_{3D}[i,j,k] = \begin{cases} X_{2D}[j,k] & \text{if } 6 \leq i < 12 \\ 0 & \text{otherwise} \end{cases}$ ▷ Repeat

$color \sim Cat(\{0\}, \{0,1\}, \{1\}, \{2\})$ ▷ Red, Yellow, Green, Blue

$X[c,i,j,k] = \begin{cases} 255 & \text{if } c \in color \text{ and } X_{3D}[i,j,k] > 0 \\ 0 & \text{otherwise} \end{cases}$

return X ▷ Colored 3D MNIST sample

4.2. Experiments set up

Experiments were carried out using PyTorch framework. Reproducing can be done using github repo. Final algorithm of optimization was as such 2

Algorithm 2 Neural optimal transport (NOT)

Input: distributions $\mathbb{P}, \mathbb{Q}$ accessible by samples; mapping network $T_\theta : \mathbb{R}^P \to \mathbb{R}^Q$;
potential network $f_\omega : \mathbb{R}^Q \to \mathbb{R}$; number of inner iterations K_T; Number of total iterations I_{total};
(strong) cost $C : \mathcal{X} \times \mathcal{P}(\mathcal{Y}) \to \mathbb{R}$
Output: learned stochastic OT map T_θ representing an OT plan between distributions $\mathbb{P}, \mathbb{Q}$
for $i = 1, 2, ...I_{total}$ **do**
  Sample batches $Y \sim \mathbb{Q}$, $X \sim \mathbb{P}$
  $\mathcal{L}_f \leftarrow \frac{1}{|X|} \sum_{x \in X} f_\omega(T_\theta(x)) - \frac{1}{|Y|} \sum_{y \in Y} f_\omega(y)$
  Update ω by using $\frac{\partial \mathcal{L}_f}{\partial \omega}$
  for $k_T = 1, 2, \ldots, K_T$ **do**
    Sample batch $X \sim \mathbb{P}$
    $\mathcal{L}_T \leftarrow \frac{1}{|X|} \sum_{x \in X} \left[C(x, T_\theta(x)) - f_\omega(T_\theta(x)) \right]$
    Update θ by using $\frac{\partial \mathcal{L}_T}{\partial \theta}$

For optimizing both transport network and potential network AdaM algorithm (Kingma & Ba, 2014) was uses with hyperparameters: lr = 0.0001, betas=$(0.9, 0.999)$, weight decay = $1e-10$. Number of total iterations I_{total} was set to 10000, number of inner iterations K_T was set to 10. Batch size equal to 64. Hardware used for optimization procedure: Nvidia A40 and Intel Xeon.

4.3. Results and discussion

Training was carried out for the full length while the notions of convergence in terms of transported samples and dynamics of optimized functionals were starting to appear since 5000 iterations approximately. Unfortunately the only way we can evaluate our model is visualization of samples since values of optimization objective isn't that much interpretable.

Figure 3. Forward experiment: Transport between colored 3D twos and colored 3D fours.

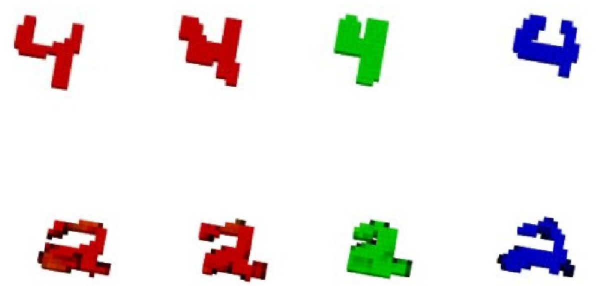

Figure 4. Backward experiment: Transport between colored 3D fours and colored 3D twos.

As you can see for both forward and backward experiments quality of transported samples $T(x) \sim \mathbb{P}$ is quite good. As you can see color of digits stays the same during optimal transport. So the hypothesis that optimal transport mapping preserves color holds true.

5. Conclusions

At the end we were able to apply Neural Optimal Transport methods (Korotin et al., 2022) to perform unpaired object-to-object deterministic translation via computing explicit optimal transport map in 3D colored MNIST setting. Results of optimal transport mapping were good quality.

Hypothesis that during optimal transport from one measure to another will not change qualities of samples not related to change of measures, such as color has been confirmed. Another confirmation that Neural Optimal Transport methods (Korotin et al., 2022) can be successfully applied to good quality style transfer.

Successful extension of domain to 3D can inspire us to go further and apply Neural Optimal Transport methods (Korotin et al., 2022) to another tasks in 3D image domain or even 3D Point Clouds Generative modeling (Achlioptas et al., 2017)

In total we have:

- Successefully applied NOT (Korotin et al., 2022) methods for style transfer
- Trained good quality model for 3D unpaired object-to-object translation
- Showed that NOT methods do work on 3D image domain
- Confirmed hypothesis that color of digits do not change through Optimal Transport mapping

References

Achlioptas, P., Diamanti, O., Mitliagkas, I., and Guibas, L. J. Learning representations and generative models for 3d point clouds. In *International Conference on Machine Learning*, 2017.

Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Korotin, A., Selikhanovych, D., and Burnaev, E. Neural optimal transport. *CoRR*, abs/2201.12220, 2022. URL `https://arxiv.org/abs/2201.12220`.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015.

A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

Sergei Kholkin (25% of work)

- Reviewing literate on the topic (3 papers)
- Coding the main algorithm
- Prepairing 3D T and f models
- Preparing the GitHub Repo
- Preparing the Sections 4 and 5 of this report
- Performing experiments

Anastasia Batsheva (25% of work)

- Creating correct 3D MNIST dataset from the regular 2D MNIST
- Implementing support functions for visualisation
- Preparing the Sections 1 and 2 of this report
- Preparing a significant part of the presentation

Artem Basharin (25% of work)

- Literature review on the topic
- Implementing utilitary functions for coloring 3D MNIST dataset
- Preparing a part of the presentation

Maxim Bobrin (25% of work)

- Formating & preparing github repo
- Running experiments
- Finding best models to work for 3D
- Writing Methods section

B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paper was used.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   General comment: If the answer is **yes**, students must explicitly clarify to which extent (e.g. which percentage of your code did you write on your own?) and which code was used.

   Students' comment: None

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   Students' comment: None

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   Students' comment: None

4. A complete description of the data collection process, including sample size, is included in the report.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   Students' comment: None

5. A link to a downloadable version of the dataset or simulation environment is included in the report.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   Students' comment: None

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   Students' comment: None

7. An explanation of how samples were allocated for training, validation and testing is included in the report.

   - [x] Yes.
   - [ ] No.
   - [ ] Not applicable.

   Students' comment: None

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.

   - [ ] Yes.
   - [ ] No.
   - [x] Not applicable.

   Students' comment: First set of hyperparameters we chose worked so haven't felt any need in such an analysis

9. The exact number of evaluation runs is included.

   - [ ] Yes.
   - [ ] No.
   - [x] Not applicable.

   Students' comment: None

10. A description of how experiments have been conducted is included.

    - [x] Yes.
    - [ ] No.
    - [ ] Not applicable.

    Students' comment: Also check https://github.com/skylooop/3DNOT for this

11. A clear definition of the specific measure or statistics used to report results is included in the report.

    - [ ] Yes.
    - [ ] No.
    - [x] Not applicable.

    Students' comment: The only way we can evaluate out model performance is quality of samples

12. Clearly defined error bars are included in the report.

    - [ ] Yes.
    - [ ] No.

- ☑ Not applicable.

Students' comment: The only way we can evaluate out model performance is quality of samples

13. A description of the computing infrastructure used is included in the report.

- ☑ Yes.
- ☐ No.
- ☐ Not applicable.

Students' comment: see 4