Optimal Tensor Transport

Tanguy Kerdoncuff, Michaël Perrot, Rémi Emonet, Marc Sebban

Optimal Tensor Transport

Tanguy Kerdoncuff [1], **Michaël Perrot** [2], **Rémi Emonet** [1], **Marc Sebban** [1]

[1] Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, F-42023, Saint-Etienne, France

[2] Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 - CRIStAL, F-59000 Lille, France

{tanguy.kerdoncuff, remi.emonet, marc.sebban}@univ-st-etienne.fr

michael.perrot@inria.fr

Abstract

Optimal Transport (OT) has become a popular tool in machine learning to align finite datasets typically lying in the same vector space. To expand the range of possible applications, Co-Optimal Transport (Co-OT) jointly estimates two distinct transport plans, one for the rows (points) and one for the columns (features), to match two data matrices that might use different features. On the other hand, Gromov Wasserstein (GW) looks for a single transport plan from two pairwise intra-domain distance matrices. Both Co-OT and GW can be seen as specific extensions of OT to more complex data. In this paper, we propose a unified framework, called Optimal Tensor Transport (OTT), which takes the form of a generic formulation that encompasses OT, GW and Co-OT and can handle tensors of any order by learning possibly multiple transport plans. We derive theoretical results for the resulting new distance and present an efficient way for computing it. We further illustrate the interest of such a formulation in Domain Adaptation and Comparison-based Clustering.

1 Introduction

Comparing two probability measures in the form of empirical distributions is at the core of many machine learning tasks. Optimal Transport (OT) (Villani 2008; Peyré, Cuturi et al. 2019) is a popular tool that allows such comparisons for datasets typically lying in a common vector space. Given two point clouds and a metric allowing to evaluate the transportation cost between two samples, the goal of OT is to learn the transport plan that minimizes the alignment cost between the two sets, resulting in the so-called Wasserstein distance. OT has been shown to be of great interest when dealing with machine learning tasks. For example, unsupervised Domain Adaptation (DA) aims at benefiting from labeled data of a source domain to classify examples drawn from a different but related target domain. The DA theory prompts us to reduce the shift between the source and the target distributions, a task that can be addressed by aligning the two datasets using OT (Courty et al. 2016, 2017; Shen et al. 2018; Damodaran et al. 2018). OT has also been successfully used in generative adversarial networks (GAN) (Goodfellow et al. 2014) to minimize the divergence between training data and samples drawn from a generative model, leading to the WGAN (Arjovsky, Chintala, and Bottou 2017; Gulrajani et al. 2017).

In order to expand the range of possible applications, different variants have been proposed in the OT literature to tackle more complex settings. While the standard OT scenario assumes that the two datasets lie in the same feature space, Gromov Wasserstein (GW) (Memoli 2007; Mémoli 2011; Peyré, Cuturi, and Solomon 2016) extends the framework to incomparable spaces by allowing the alignment of two distributions when only the within-dataset pairwise distances are available. This approach is particularly well suited to deal with graphs described by their adjacency matrices (Xu et al. 2019; Xu, Luo, and Carin 2019; Chowdhury and Mémoli 2019). The GW discrepancy has been used efficiently in various applications such as heterogeneous DA (Yan et al. 2018), word translation (Alvarez-Melis and Jaakkola 2018) or GAN (Vayer et al. 2019; Bunne et al. 2019). Recently, Co-Optimal Transport (Co-OT) (Redko et al. 2020) extended the OT theory to datasets lying in different vector spaces. The idea is to jointly learn two transport plans. The first one aligns the examples as in standard OT while the second one aligns the most similar features. This has been shown to be of particular interest in heterogeneous DA and co-clustering.

Motivation and Contribution. While GW and Co-OT already cover a wide range of problems, we claim that many other scenarios are not covered by these two extensions. Let us suppose that both the source and target distributions are represented by a collection of graphs of the same size (in terms of nodes) but of different structure (in terms of edges). This is typically the case of two graphs evolving over time. In this case, the goal of OT would be to jointly align both the two collections of graphs and the nodes. It turns out that GW would be only able to handle the special case where there exist a known one-to-one correspondence between the graphs of the two collections. Another application is inspired from comparison-based learning. Let us consider a source and a target distribution represented by a set of users who watched movies, users providing a list of triplet comparisons of the form "movie x_i is closer to x_j than to x_k". In this case, neither GW nor Co-OT is able to align the two distributions because of the nature of this triplet-based representation. A last example comes from computer vision, where one may want to align two collections of images while preserving

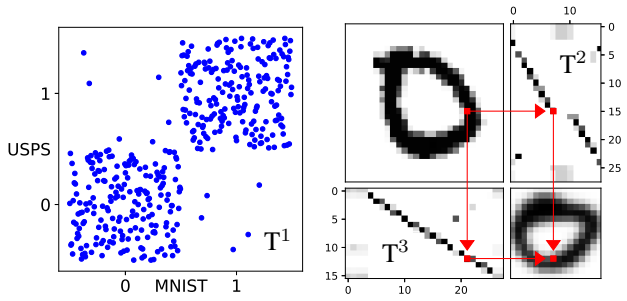

Figure 1: **(Left)** Transport plan T^1 between 400 images (only digits 0 and 1) of MNIST and USPS datasets; **(Right)** (top left) An example from MNIST and (bottom right) an example from USPS with a 90° right rotation; (top right) the OT plan T^2 between the rows of MNIST and USPS; (bottom left) the OT plan T^3 between the columns of MNIST and USPS; the arrows explain how to match the pixels between the two datasets using T^2 and T^3 obtained with OTT.

some inner structural information in rows and columns. It is worth noting that these three applications share a common characteristic: they can be represented in the form of third-order tensors. To solve OT tasks on such complex structures, it is necessary to design a framework generalizing the OT theory. This is the main contribution of this paper.

We propose *Optimal Tensor Transport* (OTT), a new OT formulation that can handle datasets represented as tensors of any order by potentially learning multiple transport plans. The underlying idea is to jointly match the different dimensions of each tensor with respect to their weights. Figure 1 illustrates this on a transportation problem between images from the MNIST (LeCun et al. 1998) to the USPS dataset (Friedman et al. 2001). Three transport plans are optimized in this scenario. T^1 is used to match the points (on the left), T^2 and T^3 preserve the structure by respectively mapping the pixel rows and pixel columns jointly (figure on the right). OTT effectively matches digits of the same class while only using supervision from the MNIST dataset. Note that the pixel-level transport plans are both close to the identity meaning that the structure of the images is automatically retrieved. We further illustrate this behaviour by extending this experiment in the supplementary material. From a theoretical perspective, we show that OTT encompasses both Co-OT and GW as well as standard OT. We also show that OTT can be seen as a distance between tensors of any order and thus it can be used to compute tensor barycenters. From an algorithmic point of view, we propose an efficient optimization scheme based on a stochastic mirror descent that allows a drastic reduction of the computational complexity.

The rest of this paper is organized as follows: Section 2 recalls some preliminary knowledge on OT, Co-OT and GW. Section 3 is dedicated to the introduction of our optimal tensor transport (OTT) setting. Section 4 proposes an efficient algorithm for solving OTT. We derive theoretical properties in Section 5 before presenting experimental results on DA and Comparison-based Clustering tasks in Section 6 .

2 Preliminary Knowledge

In this section, we recall the standard OT (Villani 2008; Peyré, Cuturi et al. 2019), the GW (Memoli 2007; Peyré, Cuturi, and Solomon 2016), and the Co-OT (Redko et al. 2020) formulations. Let p and q be two histograms of respective dimensions I and K. The set of coupling transport plans is defined as $\mathcal{U}_{pq} = \{T \in \mathbb{R}_+^{I\times K} | T\mathbb{1}_K = p, T^\top \mathbb{1}_I = q\}$ where $\mathbb{1}_R$ is a vector of ones of dimension R. The goal in discrete OT is to learn one (in standard OT and GW) or two (in Co-OT) transport plans. Note that for the sake of clarity, we only consider the discrete case here. Nevertheless, all the formulations presented in this section, as well as OTT, can be straightforwardly extended to the continuous case by replacing the sums by integrals over the compared distributions. In this case, the transport plans take the form of joint continuous measures. To prepare for our generalization, we unify the formulations below. In particular, we introduce subscripts and superscripts that are usually not used in the standard formulations. We denote the $(R-1)$-simplex $\Delta_R = \{(x_r)_{r\in[\![1,R]\!]} \in \mathbb{R}_+^R | \sum_{r=1}^R x_r = 1\}$.

Optimal Transport (Villani 2008). Let X and Y be two datasets defined over the same feature space $\mathcal{X}$ (*e.g.* $\mathcal{X} = \mathbb{R}^F$), with respectively $I_1 \in \mathbb{N}$ and $K_1 \in \mathbb{N}$ points with weights $p^1 \in \Delta_{I_1}$ and $q^1 \in \Delta_{K_1}$. The optimal transport plan between X and Y is obtained by solving:

$$\min_{T^1 \in \mathcal{U}_{p^1 q^1}} \sum_{i_1=1}^{I_1} \sum_{k_1=1}^{K_1} \mathcal{L}(X_{i_1}, Y_{k_1}) T^1_{i_1 k_1} \tag{1}$$

where X_{i_1} is example i_1 in dataset X. Here, $\mathcal{L}$ is a loss function which measures the cost of aligning two examples X_i and Y_k. An extension of OT which is conceptually different from what is covered in this article is the multi-marginal OT (Carlier 2003; Moameni 2014; Pass 2015; Friedland 2020) that aligns $R \geq 3$ datasets simultaneously: $\mathcal{L}$ becomes a function of R parameters and T^1 an R-order tensor.

Co-Optimal Transport (Redko et al. 2020). Co-Optimal Transport also aims at transporting points from two datasets X and Y. However, contrary to standard OT, these datasets may have different feature spaces $\mathcal{X} \subseteq \mathbb{R}^{I_2}$ and $\mathcal{Y} \subseteq \mathbb{R}^{K_2}$ of respective dimensions I_2 and K_2 and equipped with weights $p^2 \in \Delta_{I_2}$ and $q^2 \in \Delta_{K_2}$. The goal is to jointly match the points with a first transport plan T^1 and the features with a second one T^2. The Co-OT formulation is as follows:

$$\min_{\substack{T^1 \in \mathcal{U}_{p^1 q^1} \\ T^2 \in \mathcal{U}_{p^2 q^2}}} \sum_{i_1,i_2=1}^{I_1,I_2} \sum_{k_1,k_2=1}^{K_1,K_2} \mathcal{L}(X_{i_1 i_2}, Y_{k_1 k_2}) T^1_{i_1 k_1} T^2_{i_2 k_2} \tag{2}$$

where $X_{i_1 i_2}$ is the value of feature i_2 for example i_1.

Gromov Wasserstein (Memoli 2007). Instead of having features describing the examples, let us consider that we only have access to within-dataset pairwise similarities or dissimilarities, that is X and Y are now square matrices of dimensions $I_1 \times I_1$ and $K_1 \times K_1$. It means that the two datasets may have different feature spaces, as in Co-OT, but

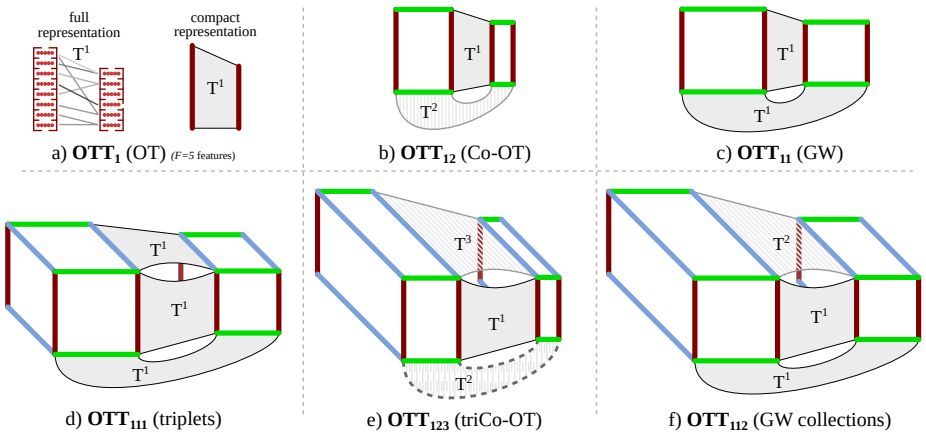

Figure 2: Various formulations of OTT with, each time, the two datasets and the different transport plans (best viewed in color). The subscripts of OTT correspond to the indices of the transport plans used in each dimension.

since these feature spaces are implicit, it is sufficient to learn a single transport plan T^1. The GW formulation is:

$$\min_{T^1 \in \mathcal{U}_{p^1 q^1}} \sum_{i_1,i_2=1}^{I_1,I_1} \sum_{k_1,k_2=1}^{K_1,K_1} \mathcal{L}(X_{i_1 i_2}, Y_{k_1 k_2}) T^1_{i_1 k_1} T^1_{i_2 k_2} \quad (3)$$

where $X_{i_1 i_2}$ is the (dis)similarity between examples i_1 and i_2. It is typical, for both Co-OT and GW, to use a comparison/loss function $\mathcal{L}$ (often the squared difference) that operates on two numbers. In Co-OT, $\mathcal{L}$ compares the value of a feature from one point in $\mathcal{X}$ with one feature from a point in $\mathcal{Y}$. In GW, it compares an entry of the pairwise matrix X to one in Y. Both formulations can be extended by allowing $\mathcal{L}$ to compare more complex entries such as F-dimensional vectors in $\mathbb{R}^F$. As illustrated in the top row of Figure 2, corresponding to the formulations of Equations (1), (2), and (3), although OT, Co-OT and GW solve different problems, they still share common principles. Below, we propose a new generalized OT formulation that encompasses all of them.

3 Optimal Tensor Transport (OTT)

Given the notational complexity involved in our generic formulation, let us first explain the intuition behind the subscripts associated with OTT as illustrated in Figure 2. Both Co-OT and GW work on matrices (that is tensors of order $D = 2$) and thus will be represented with 2 digits. Since Co-OT uses $A = 2$ different transport plans T^1 and T^2, computing Co-OT boils down to solving OTT_{12} as defined below. On the other hand, GW uses the same plan T^1 for both dimensions, thus corresponding to OTT_{11}. Note that dimensions that share a transport plan must have the same sizes. Thus, GW (OTT_{11}) deals with square matrices.

Starting to generalize, when working with tensors of order D, a given OT extension considers $A \leq D$ transport plans and associates a transport plan (index) to each dimension. This is done by specifying an *affectation function* $f : [\![1, D]\!] \to [\![1, A]\!]$ or equivalently, a D-tuple of transport plan indices, that is $f \in [\![1, A]\!]^D$. For instance, Co-OT uses $f = (1, 2)$ which corresponds to the subscript in OTT_{12}.

For a given $f \in [\![1, A]\!]^D$, we can now detail our OTT_f formulation (denoted OTT when no ambiguity arises) that defines a distance between two datasets X and Y, represented as order D+1 tensors of respective size $(I_{f(1)} ... I_{f(D)}, F)$ and $(K_{f(1)} ... K_{f(D)}, F)$. The first D dimensions will be matched between the two datasets using the transport plans, while the last dimension (F) is the feature dimension used to compare 2 points with the loss $\mathcal{L}$. To simplify the rest of the paper, we will suppose that $F = 1$, as done in Co-OT and GW above. The OTT distance between X and Y relies on finding a list of optimal transport plans $(T^a)_{a \in [\![1,A]\!]}$ under constraints on the marginals defined respectively by the weight vectors $(p^a)_{a \in [\![1,A]\!]}$ and $(q^a)_{a \in [\![1,A]\!]}$. OTT is defined as:

$$\text{OTT}_f(X, Y, (p^a)_a, (q^a)_a) = \min_{\forall a\, T^a \in \mathcal{U}_{p^a q^a}} \mathcal{E}_f\left(X, Y, (T^a)_a\right)$$

$$\text{where } \mathcal{E}_f\left(X, Y, (T^a)_{a \in [\![1,A]\!]}\right) = \quad (4)$$

$$\sum_{i_1,\ldots,i_D=1}^{I_{f(1)},\ldots,I_{f(D)}} \sum_{k_1,\ldots,k_D=1}^{K_{f(1)},\ldots,K_{f(D)}} \mathcal{L}(X_{i_1 \ldots i_D}, Y_{k_1 \ldots k_D}) \prod_{d=1}^{D} T^{f(d)}_{i_d k_d}$$

where $X_{i_1 \ldots i_D}$ is the entry at position $i_1 \ldots i_D$ in the tensor X.

From this general formulation and looking at Equations 1, 2 and 3 with the support of Figure 2, one can check that OT corresponds to OTT_1 (with F possibly > 1), Co-OT is equivalent to OTT_{12} and GW corresponds to OTT_{11}. Our OTT formulation makes it possible to handle new forms of datasets as illustrated in the second row of Figure 2. In the experiments (see Section 6), we will specifically consider two versions of OTT, each with order 3 tensors: (i) OTT_{111} corresponds to datasets of triplets (like GW but with triplets instead of pairs); (ii) OTT_{112} works with datasets that are collections of adjacency matrices. Figure 1 gives an illustration of a third kind of datasets, where OTT_{123} has been applied to collections of images, like Co-OT but with three dimensions.

It is worth mentioning that the question that we tackle here is reminiscing of another problem in the literature: the D-regular hypergraphs (Berge 1984) matching. Such a problem is indeed equivalent to OTT in the particular case where all the transport plans are identical. But it uses either a different

formulation or different constraints on the matching. Zass and Shashua (2008) proposes to find a soft matching between D-regular hypergraphs, with uniform inequality constraints, using a Kullback-Leibler objective function. Duchenne et al. (2011) also matches hypergraphs, with a formulation similar to OTT but uses only row constraints on the matching matrix. Finally, (Peyré et al. 2016) and (Ning and Georgiou 2014) propose to represent examples as PSD matrices and to align those matrices using a single transport plan where each entry is also a PSD matrix instead of a real value.

4 Algorithm to Solve OTT

In this section, we detail how to efficiently solve the main optimization problem behind Equation 4. The most used method for solving GW is called EGW (Peyré, Cuturi, and Solomon 2016). It can be seen as a Mirror Descent scheme (Beck and Teboulle 2003) with the Kullback-Leibler divergence on a regularized version of GW: $\min_{T\in\mathcal{U}_{p^1q^1}} \mathcal{E}(T) + KL(T, p^1q^{1\top})$. The idea of the Mirror Descent algorithm is to interpret the usual gradient descent, at a point x, as a minimization of the sum of a linearization of the desired function h: $\langle \nabla_x h, \bullet \rangle$ plus a regularization term $\epsilon\|x - \bullet\|_2^2$. Instead of using the Euclidean distance, Peyré, Cuturi, and Solomon (2016) use the KL divergence. Thus, at a point T^1, Peyré, Cuturi, and Solomon (2016) show that the minimization becomes equivalent to the entropy regularized OT problem (Cuturi 2013):

$$\min_{T\in\mathcal{U}_{p^1q^1}} \langle \nabla_{T^1}\mathcal{E}, T\rangle + \epsilon KL(T, p^1q^{1\top}). \tag{5}$$

Xu et al. (2019) based on the work of Xie et al. (2020) change the uniform distribution $p^1q^{1\top}$ in Equation (5) to the previous transport plan T^1. In fact, this is equivalent to applying a Mirror Descent algorithm on the original GW problem (see Equation (3)) instead of the regularized one (see supplementary material for more details). Thus, to solve the OTT problem, we use a Mirror Descent algorithm with the KL divergence. When the goal is to find multiple transport plans, we propose to use an alternating approach, similar to the Co-OT solver, where each transport plan is optimized in turn while the others remain fixed. In summary, we combine the ideas of existing solvers for Co-OT and GW and apply an alternate Mirror Descent algorithm with the KL divergence for OTT, with the main bottleneck being the computation of the gradient of $\mathcal{E}$. The pseudo-code of our approach is presented in Algorithm 1. The main steps are the following:

Algorithm 1: OTT

Require: datasets X, Y, weights $(p^a)_{a\in[\![1,A]\!]}, (q^a)_{a\in[\![1,A]\!]}$, loss function $\mathcal{L}$, nb. of samples M, regularization ϵ

1: $\forall a \in [\![1, A]\!],\ T^a = p^a q^{a\top}$
2: **for** s= 0 **to** S-1 **do**
3: **for** a= 1 **to** A **do**
4: $\widehat{\nabla_{T^a}\mathcal{E}} = M$ samples of the gradient using Equation (7).
5: $T^a = \min_{T\in\mathcal{U}_{p^a q^a}} \left\langle \widehat{\nabla_{T^a}\mathcal{E}}, T \right\rangle + \epsilon KL(T, T^a)$
6: **end for**
7: **end for**

Step 1: We initialize the transport plans (line 1) with the marginal product.

Step 2: We compute the gradient of $\mathcal{E}$. For the sake of clarity, we assume that the aligned tensors are "cubic", that is all their dimensions are of the same size N. In this case, the overall gradient with respect to T^a is a N^2 matrix:

$$\nabla_{T^a}\mathcal{E} = \sum_{\{d'|f(d')=a\}} \sum_{\substack{i_1,\dots,i_{d'-1}=1\\ i_{d'+1},\dots,i_D=1}}^{\substack{I_{f(1)},\dots,I_{f(d'-1)}\\ I_{f(d'+1)},\dots,I_{f(D)}}} \sum_{\substack{k_1,\dots,k_{d'-1}=1\\ k_{d'+1},\dots,k_D=1}}^{\substack{K_{f(1)},\dots,K_{f(d'-1)}\\ K_{f(d'+1)},\dots,K_{f(D)}}} \mathcal{L}\begin{pmatrix} X_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D},\\ Y_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}\end{pmatrix} \prod_{d=1|d\neq d'}^{D} T^{f(d)}_{i_d k_d}. \tag{6}$$

Note that computing the overall gradient exactly would be too expensive. Indeed, a naive approach leads to $O(N^{2D})$ operations which is prohibitively high. To simplify the computation, a first idea would be to generalize the approach used for GW by Peyré, Cuturi, and Solomon (2016) to our problem. This would reduce the complexity to $O(N^{D+1})$ for a particular class of functions $\mathcal{L}$, notably the square loss. We provide a proof of this approach in the supplementary material. Nevertheless, this remains too expensive as soon as $D = 3$. Thus, instead, we propose to use a stochastic Mirror Descent (Zhou et al. 2017; Zhang and He 2018; Hanzely and Richtárik 2021). This idea was used for the GW problem by Kerdoncuff, Emonet, and Sebban (2021) and we generalize it to our OTT problem. The main idea is to notice that the gradient of $\mathcal{E}$ with respect to T^a can be seen as a sum of expectations over matrices of size N^2, denoted $(\mathbf{C}^{d'})_{\{d'|f(d')=a\}}$, such that:

$$\mathbb{P}\left(\mathbf{C}^{d'} = \mathcal{L}\begin{pmatrix} X_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D},\\ Y_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}\end{pmatrix}\right) = \prod_{d=1|d\neq d'}^{D} T^{f(d)}_{i_d k_d}$$

$$\text{with} \sum_{\substack{i_1,\dots,i_{d'-1}=1\\ i_{d'+1},\dots,i_D=1}}^{\substack{I_{f(1)},\dots,I_{f(d'-1)}\\ I_{f(d'+1)},\dots,I_{f(D)}}} \sum_{\substack{k_1,\dots,k_{d'-1}=1\\ k_{d'+1},\dots,k_D=1}}^{\substack{K_{f(1)},\dots,K_{f(d'-1)}\\ K_{f(d'+1)},\dots,K_{f(D)}}} \prod_{\substack{d=1\\ d\neq d'}}^{D} T^{f(d)}_{i_d k_d} = 1$$

since $\forall a \in [\![1, A]\!],\ \sum_{i,k=1}^{I_a,K_a} T^a_{ik} = 1$. The gradient can then be reformulated as:

$$\nabla_{T^a}\mathcal{E} = \sum_{\{d'|f(d')=a\}} \mathbb{E}\left(\mathbf{C}^{d'}\right). \tag{7}$$

It means that one may obtain an unbiased estimate of the gradient in $O(MN^2)$ operations where M is the number of samples to estimate the expectations.

Step 3: The last step (line 5) requires to solve a regularized OT problem, that can be efficiently solved using a Sinkhorn solver (Xu et al. 2019; Cuturi 2013).

We refer the interested reader to Peyré, Cuturi, and Solomon (2016) and Xu et al. (2019) for an analysis of the efficiency of the Mirror Descent algorithm, and to Kerdoncuff, Emonet, and Sebban (2021) for investigations on the precision of the stochastic approximation of the gradient. We also provide in the supplementary material an experiment specific to our new formulation to show how well the gradient is approximated with an increasing order D of the tensors.

5 Theoretical Results

In this section, we derive two main theoretical results. Theorem 1 shows that as long as the cost function is a proper distance, then OTT is a distance between D-order tensors. Thus, we can naturally define an OTT barycenter between tensors. Theorem 2 states that the optimal barycenter can be found in closed form for particular loss functions.

Theorem 1. *OTT is a distance between weighted tensors $(X, (p^a)_{a\in[\![1,A]\!]})$ and $(Y, (q^a)_{a\in[\![1,A]\!]})$ represented in canonical form (Definition 1 in the supplementary material), for any affectation function f, as long as $\mathcal{L}$ is a proper distance.*

The proof is provided in the supplementary material. This result notably extends the distance proof of Co-OT (Redko et al. 2020) to matrices of different sizes and to non-uniform weights. Even though their comparison with OTT is out of the scope of this paper, notice that other distances exist between higher-order tensors (De Lathauwer, De Moor, and Vandewalle 2000; Liu, Liu, and Chan 2010; Lai et al. 2013).

Since OTT is a distance, we can define an OTT barycenter between several tensors with any affectation function f.

Definition 1. *(OTT barycenter) Assume that we are given $B \geq 1$ weighted tensors of sizes $((K_a^b)_{a\in[\![1,A]\!]})_{b\in[\![1,B]\!]}$ denoted $\left(X^b \in \mathbb{R}^{K^b_{f(1)}\dots K^b_{f(D)}}, (q^{a,b} \in \Delta_{K_a^b})_{a\in[\![1,A]\!]}\right)_{b\in[\![1,B]\!]}$. Let $\lambda \in \Delta_B$ be the weights quantifying the importance of each tensor. For fixed size $(I_a)_{a\in[\![1,A]\!]}$ and weights $(p^a \in \Delta_{I_a})_{a\in[\![1,A]\!]}$, the OTT barycenter is defined as*

$$\min_{X\in\mathbb{R}^{I_{f(1)}\dots I_{f(D)}}} \sum_{b=1}^{B} \lambda_b \mathit{OTT}(X, X^b, (p^a)_a, (q^{a,b})_a). \quad (8)$$

Note that the barycenter could also be defined in a similar manner with the marginals $(p^a)_{a\in[\![1,A]\!]}$ not fixed.

To solve Problem (8), we propose to minimize alternatively the objective function w.r.t. X and $(T^{a,b})_{a\in[\![1,A]\!]}$, the transport plans between X and X^b. The latters can be found independently for each $b \in [\![1, B]\!]$ using Algorithm 1. Interestingly, X can be obtained in closed form for particular loss functions, which generalizes, in particular to Co-OT, a known result for OT and GW (Peyré, Cuturi, and Solomon 2016). This is summarized in the next theorem.

Theorem 2. *Assume that the loss $\mathcal{L}$ is continuous and can be written as $\mathcal{L}(x, y) = f_1(x) + f_2(y) - h_1(x)h_2(y)$ with four functions (f_1, f_2, h_1, h_2) such that $\frac{f_1'}{h_1'}$ is invertible. Further assume that $\mathcal{L}(x, y) \underset{x\to\pm\infty}{\longrightarrow} +\infty$. For fixed $((T^{a,b})_{a\in[\![1,A]\!]})_{b\in[\![1,B]\!]}$, for all $(i_d \in [\![1, I_{f(d)}]\!])_{d\in[\![1,D]\!]}$, the optimal solution $X^*_{i_1,\dots,i_D}$ of Problem (8) is equal to*

$$\left(\frac{f_1'}{h_1'}\right)^{-1}\left(\sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K^b_{f(1)},\dots,K^b_{f(D)}} h_2(X^b_{k_1\dots k_D}) \prod_{d=1}^{D} \frac{T^{f(d),b}_{i_d k_d}}{p^{f(d)}_{i_d}}\right).$$

In particular, when $\mathcal{L}$ is the squared euclidean distance,

$$X^*_{i_1,\dots,i_D} = \sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K^b_{f(1)},\dots,K^b_{f(D)}} X^b_{k_1\dots k_D} \prod_{d=1}^{D} \frac{T^{f(d),b}_{i_d k_d}}{p^{f(d)}_{i_d}}.$$

Note that to obtain a barycenter using loss functions that are not covered by Theorem 2, for example the absolute loss, one can resort to a gradient based optimization scheme.

In the next section, we present experiments focused on 3D-tensors alignments. Nevertheless, it is worth noticing that our theoretical results and Algorithm 1 hold for any tensor order and thus might be used with $D = 4$, for example in comparison based learning tasks (Ghoshdastidar, Perrot, and von Luxburg 2019) or to match hypergraphs (Berge 1984).

6 Experiments

In this section, we illustrate the interest of OTT on two different tasks[1]. First, following the success of OT in Domain Adaptation (Courty et al. 2016), we propose to predict the genres of recent movies based on labeled older movies by relying only on users preferences. We advantageously use a 3D-tensor formulation to take into account the particularity of each user. In a second experiment, we use the OTT barycenter in a Comparison-Based Clustering task.

6.1 Domain Adaptation (DA)

We consider a DA task on the Movielens dataset (Harper and Konstan 2015). The goal is to adapt a model learned on old movies (source) to predict the genres of new movies (target).

Datasets. We build two 3-orders tensor X^s (source) and X^t (target) based on the ratings of the users. The entry (i, j, k) in X^s (and similarly for X^t) is 1 if the user i preferred the movie j over the movie k, -1 if the movie k is preferred over the movie j and 0 if the user i cannot choose. As the users did not rate every movie, we use the 0.33 percentile of their personal rates as a default rating. For both the new and old movies, we identify 4 different groups of movies: *Thriller/Crime/Drama* (T), *Fantasy/Sci-Fi* (F), *War/Western* (W), and *Children's/Animation* (C). We then create 6 pairwise binary classification datasets of 200 movies each by selecting 2 classes among the four aforementioned ones. We assume that we have access to all the labels for the old movies (source) but only to a single random label per class for the new movies (target). The goal is to learn a model that is as accurate as possible on the target. Since many movies have a small number of ratings and many users only rated a few movies, we focus on the 100 users with the highest number of ratings and the 200 most rated films for those users.

Baselines. Even though OTT_{122} is, to the best of our knowledge, the first algorithm that allows direct DA on such tensor-based datasets, we still propose various baselines by reducing the X^s and X^t tensors into matrices by averaging along one dimension. **Rdm** is a first naive baseline that simply outputs random labels. For the next three baselines, we average over the user dimension. Then, **SVM** applies a SVM (Cortes and Vapnik 1995) classifier only on the target domain, using the columns of the matrix as features. **S-GWL** (Xu, Luo, and Carin 2019) interprets the obtained matrices as adjacency matrices of graphs and matches the nodes of the two graphs. **GW** (Peyré, Cuturi, and Solomon

[1]The code to reproduce all the experiments is available online: https://github.com/Hv0nnus/Optimal_Tensor_Transport

Table 1: Accuracy on 6 DA tasks with the hyperparameters found using the unsupervised proposed method. To evaluate the best possible performance reachable by each method, AVG^{best} displays the accuracy with the best hyperparameters using the ground truth of the target domain.

Datasets	SVM	S-GWL	GW	Co-OT	OTT
T,F	62.5	63.0	62.0	72.0	**80.8**±1
T,C	69.0	77.0	78.0	83.0	**97.0**±0
T,W	32.5	61.0	63.0	65.5	**71.3**±5
F,C	**74.5**	72.0	74.0	74.0	70.2±4
F,W	53.0	53.0	60.5	47.0	**67.9**±2
C,W	60.0	57.0	52.0	67.5	**76.8**±6
AVG	58.6	63.8	64.9	68.2	**77.3**±3
AVG^{best}	58.6	66.3	71.0	70.7	**78.9**±3
Time (s)	0.1	94	673	4	5940

2016) solves the GW problem directly on the obtained matrices. The last baseline, **Co-OT**, uses a matrix obtained by averaging over one movie dimension, which leads to a matrix (users, movies). The two axes are then mapped jointly between the new and old movies. For all the methods that provide a transport plan T between the movies, the class of a target movie y^t_j is predicted via label propagation (Redko et al. 2019a) of the source label y^s, that is $y^t_k = y^s T_{\bullet k}$. The stochastic methods are run 10 times and the mean and standard deviation are reported.

Experimental setup and hyperparameter tuning. As the initialization is key to avoid local minima, we take advantage of both the labels and our stochastic algorithm by sampling only the labelled points in the source and target for the first gradient estimation. The squared euclidean loss is used for $\mathcal{L}$ and we estimate the gradient of OTT using $M = 1000$ samples. S is set to 1000 iterations in Algorithm 1. For each method that uses the OT Sinkhorn solver, notably OTT, we replace it with the semi-supervised algorithm OTDA proposed by Courty et al. (2016) which adds a $l_p - l_1$ regularization to take advantage of the available source labels. In DA, tuning the hyperparameters is often key as there is not enough target labeled movies. As the goal of DA is to reduce the divergence between the two datasets (Ben-David et al. 2007; Redko et al. 2019b), we can use the distance between the source and the target as a criterion to choose the hyperparameters for each method. To compute the OTT distance, we resort to the sampling scheme already used to approximate the GW distance in Kerdoncuff, Emonet, and Sebban (2021). The Kullback-Leibler regularization parameter ϵ of the Sinkhorn method (Cuturi 2013) is selected in the range $[10^{-5}, 10^{2}]$ and the class regularization η of OTDA (Courty et al. 2016) in $[10^{-4}, 10^{1}]$. The hyperparameters selection is limited to 24 hours for each method and dataset.

Results. The accuracy of each method is reported in Table 1. OTT achieves better performances than the other baselines on 5 out of 6 datasets. This result was expected as OTT is the only method which takes full advantage of the 3D structure of the data. Interestingly, OTT still behaves better than the baselines even when one uses the ground truth over the target domain to tune their hyperparameters (that would be cheating) as shown in the line AVG$^{\text{best}}$ of Table 1.

We now analyze the impact of the different hyperparameters on the accuracy. We report the results on each dataset in the supplementary material and only consider the global average in Figure 3. The leftmost plot displays the accuracy for increasing values of the KL regularization parameter ϵ. The black markers correspond to the lowest achieved distance for each method. It is worth noting that this usually corresponds to a reasonable accuracy, which supports our hyperparameter tuning procedure. We notice a similar behaviour for the η parameter of OTDA as reported in the supplementary material. In Figure 3 (middle), we report the target accuracy with respect to the number of target labels available. We can notice that OTT is always better, even in the completely unsupervised scenario. Lastly, in the experiments reported in Table 1, we never use the fact that the users comparing the movies are the same for both old and new movies. Here, we study the impact of making this information available. To this end, we fix the transport plan for an increasing number of users. Figure 3 (right) shows that this information greatly improves the target accuracy of the methods that can handle it, especially OTT. Interestingly, as indicated with the black marker, the smallest distance is achieved with the highest number of known pairings, which corresponds to the highest number of constraints on the users transport plan. This supports the key assumption of this experiment: a good matching between users leads to a better matching of similar movies. This also highlights a limit of a mirror descent-based solver as it struggles to find the global minimum without this additional information.

6.2 Comparison Based Clustering

In this second series of experiments, we show that OTT barycenters can be used competitively to address an unbalanced comparison-based clustering task.

Comparison-based learning deals with the problem of learning from examples when neither an explicit representation nor a pairwise distance matrix is available (Vikram and Dasgupta 2016; Ukkonen 2017; Emamjomeh-Zadeh and Kempe 2018; Perrot, Esser, and Ghoshdastidar 2020). Instead, it is assumed that only triplet comparisons of the form "example x_i is closer to x_j than to x_k" are available. This field stems from the fact that relative judgments are usually easier than absolute ones for human observers (Shepard 1962; Young 1987; Stewart, Brown, and Chater 2005). For example, triplet-based queries are easier to answer than exact distance estimations. Given a set of examples and a given number of triplet comparisons, a dataset can be represented as a third order tensor where the entry (i, j, k) contains 1 if example x_i is closer to x_j than to x_k and -1 otherwise. In comparison based clustering, the goal is to identify relevant groups in the examples, using only the information contained in the aforementioned tensor. As the three dimensions of the cubic tensor correspond to the same points we will use the same tranport plan for all the dimensions, that is OTT_{111}.

Setting. To show the interest of our method for clustering unbalanced triplet datasets, we take inspiration from the experimental setup of Perrot, Esser, and Ghoshdastidar (2020).

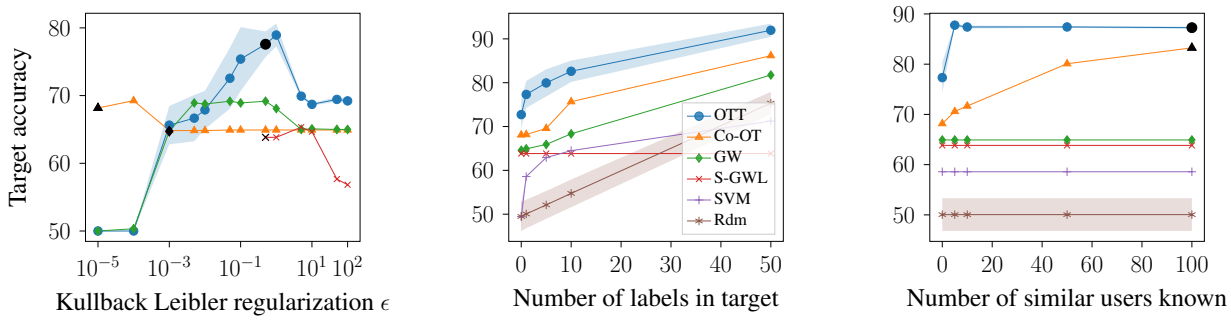

Figure 3: Target accuracy averaged over all the datasets. The shadow area represents the standard deviation for the stochastic methods. When relevant, the black symbols correspond to the parameter values achieving, for each method, the lowest distances between the datasets on average. (**Left**) Target accuracy for various values of ϵ. (**Middle**) Target accuracy for an increasing target supervision. (**Right**) Target accuracy for an increasing number of similar known users who rated both old and new movies.

Table 2: ARI (in percentage) for unbalanced comparison-based clustering on MNIST. Each line corresponds to the average over 10 combinations of classes, each run 10 times.

nb. points per class	AddS3	AddS3$_s$	t-STE	t-STE$_s$	OTT
200,20,20	43±12	80±17	56±7	**91**±4	**91**±4
30,3,1	28±9	82±17	49±14	**91**±14	89±14
30,3,3	37±13	78±23	52±09	**93**±19	87±19
300,30,10	28±4	83±07	48±10	**89**±4	**89**±04
AVG	34±9	81±16	51±10	**91**±8	89±10

For a given dataset, we find the OTT$_{111}$ barycenter ($b = 1$) of size (I_1, I_1, I_1) where I_1 is the number of clusters that we are looking for. The intuition is that similar examples should be sent by the transport plan to the same point in the barycenter since the latter summarizes the initial points.

Datasets. We consider some 3-class unbalanced subsamples of the MNIST dataset (LeCun et al. 1998). For a given number of examples per class (for example, 200,20,20), we consider 10 random draws for the 3 classes and, for each of these, we further consider 10 random draws for the actual images. Given N points in each unbalanced dataset, we randomly select $N \log(N)^3$ triplets of the form $d(x_i, x_j) > d(x_i, x_k)$ as suggested by Perrot, Esser, and Ghoshdastidar (2020). The distance between two digits is the euclidean distance after an UMAP projection in 2 dimensions. To simulate a real dataset, some noise is added by randomly flipping $d(x_i, x_j) > d(x_i, x_k)$ to $d(x_i, x_j) < d(x_i, x_k)$ with probability 0.1 for each triplet selected.

Baselines. We use two main triplet clustering baselines: (i) **t-STE** (Van Der Maaten and Weinberger 2012) which projects the triplets into a vector space followed by k-means (Lloyd 1982), and (ii) **AddS3** (Perrot, Esser, and Ghoshdastidar 2020) which estimates a pairwise similarity matrix also followed by k-means. Moreover, as the OT formulation requires the marginal as a prior, we assume that the proportions of the clusters are known. To stay fair, we propose two variants (AddS3$_s$, t-STE$_s$) of the previous baselines where we replace the k-means step by an OT barycenter step which takes the marginal information into account.

Hyperparameters. We use default hyperparameters, reported in the supplementary material, for t-STE, AddS3, and OTT with the KL regularization parameter set to $\epsilon = 0.1$. To ensure convergence, we also set the number of samples $M = 100$ and the number of iteration $S = 500$ between each of the 20 barycenter updates. Finally, to take advantage of the closed form derived in Theorem 2, we use the squared euclidean loss for OTT.

Results. The Adjusted Rand Index (ARI) (Hubert and Arabie 1985) between the predicted clusters and the ground truth is displayed in Table 2. Overall, OTT has better performances than AddS3$_s$ on average on all datasets while being slightly worse than t-STE$_s$. Furthermore, for both AddS3 and t-STE, using the unbalancedness information improves the performances. The closeness between our approach and t-STE$_s$ is further investigated in the supplementary material, where we show a theoretical connection between t-STE and the OTT barycenter. The choice of the unbalanced setting is motivated by the fact that the two other baselines do not take into account this information during their first step, while OTT directly uses it in its unique step.

7 Conclusion

We presented OTT, a new OT formulation that can be used to align high dimensional tensors using potentially several transport plans. OTT generalizes various existing OT problems, such as GW and Co-OT, by defining a new tensor distance. We proposed an efficient algorithm to solve the underlying problem and demonstrated the competitiveness of OTT in DA and Comparison-based clustering. While our new approach unlocks new applications, this comes with a cost. First, despite having access to a solver that drastically reduces the computational complexity of the formulation, it still does not scale well on large datasets with high order tensors. Finally, we leave for future work a natural extension, Fused-OTT, inspired by Vayer et al. (2020), that would combine several OTT problems together. This approach could allow us to align datasets that are independently represented by multiple tensors of potentially different orders.

Acknowledgements

This paper is part of the TADALoT Project funded by the region Auvergne-Rhône-Alpes (France) with the Pack Ambition Recherche (2017, 17 011047 01).

References

Alvarez-Melis, D.; and Jaakkola, T. 2018. Gromov-Wasserstein Alignment of Word Embedding Spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*.

Beck, A.; and Teboulle, M. 2003. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*.

Ben-David, S.; Blitzer, J.; Crammer, K.; Pereira, F.; et al. 2007. Analysis of representations for domain adaptation. *Advances in Neural Information Processing Systems*.

Berge, C. 1984. *Hypergraphs: combinatorics of finite sets*. Elsevier.

Bunne, C.; Alvarez-Melis, D.; Krause, A.; and Jegelka, S. 2019. Learning Generative Models across Incomparable Spaces. In *International Conference on Machine Learning*.

Carlier, G. 2003. On a class of multidimensional optimal transportation problems. *Journal of convex analysis*.

Chowdhury, S.; and Mémoli, F. 2019. The Gromov–Wasserstein distance between networks and stable network invariants. *Information and Inference: A Journal of the IMA*.

Cortes, C.; and Vapnik, V. 1995. Support-vector networks. *Machine learning*.

Courty, N.; Flamary, R.; Habrard, A.; and Rakotomamonjy, A. 2017. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems*.

Courty, N.; Flamary, R.; Tuia, D.; and Rakotomamonjy, A. 2016. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9): 1853–1865.

Cuturi, M. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*.

Damodaran, B. B.; Kellenberger, B.; Flamary, R.; Tuia, D.; and Courty, N. 2018. DeepJDOT: Deep Joint Distribution Optimal Transport for Unsupervised Domain Adaptation. In *European Conference on Computer Vision*. Springer.

De Lathauwer, L.; De Moor, B.; and Vandewalle, J. 2000. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*.

Duchenne, O.; Bach, F.; Kweon, I.-S.; and Ponce, J. 2011. A tensor-based algorithm for high-order graph matching. *IEEE transactions on pattern analysis and machine intelligence*.

Emamjomeh-Zadeh, E.; and Kempe, D. 2018. Adaptive Hierarchical Clustering Using Ordinal Queries. In *Symposium on Discrete Algorithms*.

Friedland, S. 2020. Tensor optimal transport, distance between sets of measures and tensor scaling. *arXiv preprint arXiv:2005.00945*.

Friedman, J.; Hastie, T.; Tibshirani, R.; et al. 2001. *The elements of statistical learning*. Springer series in statistics New York.

Ghoshdastidar, D.; Perrot, M.; and von Luxburg, U. 2019. Foundations of Comparison-Based Hierarchical Clustering. In *Advances in Neural Information Processing Systems*.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. *Advances in Neural Information Processing Systems*.

Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; and Courville, A. C. 2017. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems*.

Hanzely, F.; and Richtárik, P. 2021. Fastest rates for stochastic mirror descent methods. *Computational Optimization and Applications*.

Harper, F. M.; and Konstan, J. A. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems*.

Hubert, L.; and Arabie, P. 1985. Comparing partitions. *Journal of classification*.

Kerdoncuff, T.; Emonet, R.; and Sebban, M. 2021. Sampled Gromov Wasserstein. *Machine Learning*.

Lai, Z.; Xu, Y.; Yang, J.; Tang, J.; and Zhang, D. 2013. Sparse tensor discriminant analysis. *IEEE transactions on Image processing*.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*.

Liu, Y.; Liu, Y.; and Chan, K. C. 2010. Tensor distance based multilinear locality-preserved maximum information embedding. *IEEE Transactions on neural networks*.

Lloyd, S. 1982. Least squares quantization in PCM. *IEEE transactions on information theory*.

Memoli, F. 2007. On the use of Gromov-Hausdorff Distances for Shape Comparison. In *Eurographics Symposium on Point-Based Graphics*. The Eurographics Association.

Mémoli, F. 2011. Gromov–Wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*.

Moameni, A. 2014. Multi-marginal Monge–Kantorovich transport problems: A characterization of solutions. *Comptes Rendus Mathematique*.

Ning, L.; and Georgiou, T. T. 2014. Metrics for matrix-valued measures via test functions. In *53rd IEEE Conference on Decision and Control*, 2642–2647. IEEE.

Pass, B. 2015. Multi-marginal optimal transport: theory and applications. *ESAIM: Mathematical Modelling and Numerical Analysis*.

Perrot, M.; Esser, P. M.; and Ghoshdastidar, D. 2020. Near-optimal comparison based clustering. *arXiv preprint arXiv:2010.03918*.

Peyré, G.; Chizat, L.; Vialard, F.-X.; and Solomon, J. 2016. Quantum optimal transport for tensor field processing. *arXiv preprint arXiv:1612.08731*.

Peyré, G.; Cuturi, M.; and Solomon, J. 2016. Gromov-wasserstein averaging of kernel and distance matrices. In *International Conference on Machine Learning*.

Peyré, G.; Cuturi, M.; et al. 2019. Computational optimal transport. *Foundations and Trends® in Machine Learning*.

Redko, I.; Courty, N.; Flamary, R.; and Tuia, D. 2019a. Optimal transport for multi-source domain adaptation under target shift. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR.

Redko, I.; Morvant, E.; Habrard, A.; Sebban, M.; and Bennani, Y. 2019b. *Advances in domain adaptation theory*. Elsevier.

Redko, I.; Vayer, T.; Flamary, R.; and Courty, N. 2020. CO-Optimal Transport. In *Advances in Neural Information Processing Systems*.

Shen, J.; Qu, Y.; Zhang, W.; and Yu, Y. 2018. Wasserstein distance guided representation learning for domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Shepard, R. N. 1962. The analysis of proximities: Multidimensional scaling with an unknown distance function. I. *Psychometrika*.

Stewart, N.; Brown, G. D. A.; and Chater, N. 2005. Absolute identification by relative judgment. *Psychological review*.

Ukkonen, A. 2017. Crowdsourced correlation clustering with relative distance comparisons. *arXiv preprint arXiv:1709.08459*.

Van Der Maaten, L.; and Weinberger, K. 2012. Stochastic triplet embedding. In *2012 IEEE International Workshop on Machine Learning for Signal Processing*. IEEE.

Vayer, T.; Chapel, L.; Flamary, R.; Tavenard, R.; and Courty, N. 2020. Fused Gromov-Wasserstein distance for structured objects. *Algorithms*.

Vayer, T.; Flamary, R.; Tavenard, R.; Chapel, L.; and Courty, N. 2019. Sliced Gromov-Wasserstein. In *Advances in Neural Information Processing Systems*.

Vikram, S.; and Dasgupta, S. 2016. Interactive bayesian hierarchical clustering. In *International Conference on Machine Learning*.

Villani, C. 2008. *Optimal transport: old and new*. Springer Science & Business Media.

Xie, Y.; Wang, X.; Wang, R.; and Zha, H. 2020. A fast proximal point method for computing exact wasserstein distance. In *Uncertainty in Artificial Intelligence*.

Xu, H.; Luo, D.; and Carin, L. 2019. Scalable gromov-Wasserstein learning for graph partitioning and matching. In *Advances in Neural Information Processing Systems*.

Xu, H.; Luo, D.; Zha, H.; and Duke, L. C. 2019. Gromov-Wasserstein Learning for Graph Matching and Node Embedding. In *International Conference on Machine Learning*.

Yan, Y.; Li, W.; Wu, H.; Min, H.; Tan, M.; and Wu, Q. 2018. Semi-Supervised Optimal Transport for Heterogeneous Domain Adaptation. In *International Joint Conference on Artificial Intelligence*.

Young, F. W. 1987. *Multidimensional scaling: History, theory, and applications.* Lawrence Erlbaum Associates.

Zass, R.; and Shashua, A. 2008. Probabilistic graph and hypergraph matching. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE.

Zhang, S.; and He, N. 2018. On the convergence rate of stochastic mirror descent for nonsmooth nonconvex optimization. *arXiv preprint arXiv:1806.04781*.

Zhou, Z.; Mertikopoulos, P.; Bambos, N.; Boyd, S.; and Glynn, P. W. 2017. Stochastic mirror descent in variationally coherent optimization problems. *Advances in Neural Information Processing Systems*.

Supplementary Material: Optimal Tensor Transport

In this supplementary material, we provide details on the various results presented in the main paper as well as complementary experiments. It is organised as follows. First, in Section 1 we illustrate the interest of our method compared to Co-OT. Then, in Section 2, we formally show that the approach of Xu et al. (2019) used for GW is a Mirror Descent algorithm. In Section 3, we formally investigate the computational complexity of the sampling approach for the gradient approximation with $D \leq 2$. In Section 4 we provide the proofs of the different theoretical results. Finally, in Section 5, we provide details on the hyperparameters used in the experiments as well as additional results.

Notations

To simplify the notations, we will use the following shortcuts:

$$\sum_{\forall d\ i_d} = \sum_{i_1,\dots,i_D=1}^{I_{f(1)},\dots,I_{f(D)}}, \qquad \sum_{\forall d\neq d'\ i_d} = \sum_{\substack{i_1,\dots,i_{d'-1}=1 \\ i_{d'+1},\dots,i_D=1}}^{\substack{I_{f(1)},\dots,I_{f(d'-1)} \\ I_{f(d'+1)},\dots,I_{f(D)}}} \quad \text{and} \quad \prod_{d\neq d'} = \prod_{\substack{d=1 \\ d\neq d'}}^{D}. \tag{1}$$

1 Illustration of the difference between OTT and Co-OT

In this section, we provide the same pixel transportation image as the one provided in (Redko et al. 2020) for Co-OT, but for both Co-OT and OTT$_{123}$. The idea is to create an image of size 28 by 28 (the same size as the MNIST images) with a different color in each pixel and then to use the transport plans learned to map MNIST onto USPS to transform this image. It gives us new images (one for Co-OT and one for OTT) of the same size as the USPS images that illustrate how both methods alter the pictures when performing transportation. This is illustrated Figure 1. Note that the transport plans used for OTT are the ones used in Figure 1 in the main paper.

For Co-OT, many pixels carry no information as we use only 0 and 1 labels, thus the colored USPS image has some coherence only in the middle of the image. On the other hand, with OTT$_{123}$ we force columns to be mapped on (whole) columns and rows to be mapped on (whole) rows. By doing so, OTT can extrapolate using the row/column structure inherent to images and the color visualisation is much smoother than for Co-OT.

2 Mirror Descent

In this section, we prove that the method proposed by (Xu et al. 2019) is a Mirror Descent algorithm on the original GW problem. More specifically, we prove that Equation (7) in their Section 3.1 is the same step as the step used in a Mirror Descent algorithm (Beck and Teboulle 2003) with Kullback-Leibler divergence. The Mirror Descent method, at a given point T_s at the iteration s, searches for the next minimum T_{s+1} with,

$$T_{s+1} = \underset{T\in\mathcal{U}_{\mu\nu}}{\operatorname{argmin}} \langle \nabla_{T_s}\mathcal{E}, T\rangle + \epsilon KL(T, T_s). \tag{2}$$

Which is equivalent to,

$$T_{s+1} = \underset{T\in\mathcal{U}_{\mu\nu}}{\operatorname{argmin}} \langle \nabla_{T_s}\mathcal{E} - \epsilon log(T_s), T\rangle + \epsilon \langle T, log(T)\rangle . \tag{3}$$

The only difference is the missing factor 2 in (Peyré, Cuturi, and Solomon 2016) that should be here due to the derivative, but both problems are equivalent with a re-scaling of ϵ by a factor 2.

3 Complexity of the gradient computation of OTT

We prove in this section that the gradient of OTT, which is necessary for the Mirror Descent algorithm (Beck and Teboulle 2003), can be computed in a time complexity of $O(N^{D+1})$ for particular loss functions. This generalizes a known result for GW (Peyré, Cuturi, and Solomon 2016). Note that, in the main paper, we propose a more efficient approach based on sampling. We only mention this result for the sake of comparison with state of the art approaches.

Here, we assume that the feature dimension F is small and that every other dimension of the tensor is N to simplify the notations. We also assume that the order of the tensor D is fixed and small and we analyze the time complexity only with respect to N. First, we recall the gradient of $\mathcal{E}$ using the notations introduced at the start of this supplementary material:

$$\nabla_{T^a}\mathcal{E} = \sum_{\{d'|f(d')=a\}} \sum_{\forall d\neq d'\ i_d} \sum_{\forall d\neq d'\ k_d} \mathcal{L}(X_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}, Y_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}) \prod_{d\neq d'} T^{f(d)}_{i_d k_d}. \tag{4}$$

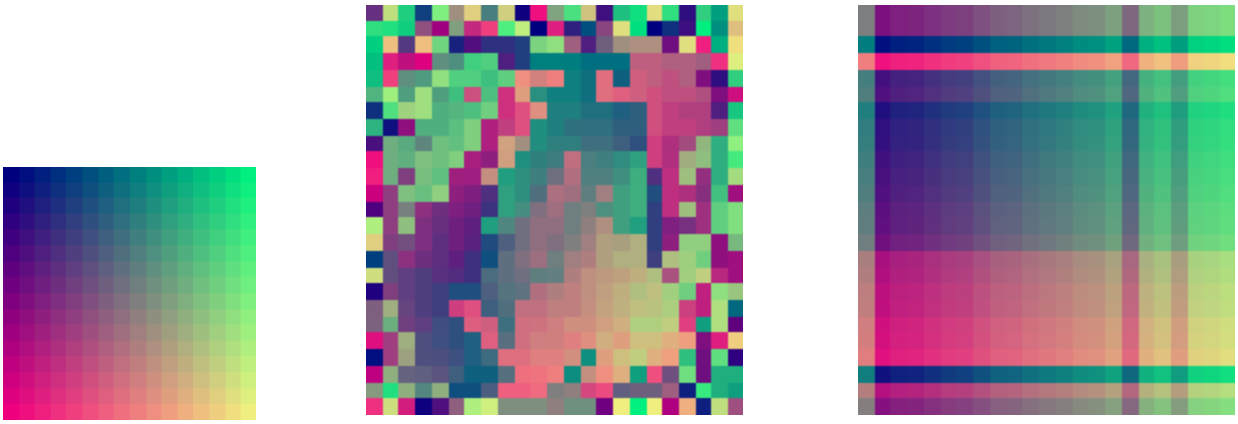

Figure 1: (**Left**) MNIST image with a color associated with each pixel. (**Middle** and **Right**) USPS image colored by the transportation of the left image by the transport plans found with Co-OT and OTT_{123} respectively. Few rows and columns are deleted in this representation as they have no mass.

We suppose that the loss $\mathcal{L}$ is continuous and can be written as $\mathcal{L}(x, y) = f_1(x) + f_2(y) - h_1(x)h_2(y)$ with four functions $(f_1, f_2 : \mathbb{R}^F \to \mathbb{R})$ and $(h_1, h_2 : \mathbb{R}^F \to \mathbb{R}^F)$. This is notably the case for the squared Euclidean distance or the Kullback-Leibler divergence. As each element of the first sum in Equation (4) will be computed independently, we can fix d'. We thus have to compute the following term,

$$\begin{aligned}
&\sum_{\forall d \neq d'} \; \sum_{i_d} \; \sum_{\forall d \neq d'} \; \sum_{k_d} f_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}) \prod_{d \neq d'} T^{f(d)}_{i_d k_d} \\
&+ \sum_{\forall d \neq d'} \; \sum_{i_d} \; \sum_{\forall d \neq d'} \; \sum_{k_d} f_2(Y_{k_1 \dots k_{d'-1}, \bullet, k_{d'+1} \dots k_D}) \prod_{d \neq d'} T^{f(d)}_{i_d k_d} \\
&+ \sum_{\forall d \neq d'} \; \sum_{i_d} \; \sum_{\forall d \neq d'} \; \sum_{k_d} h_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}) h_2(Y_{k_1 \dots k_{d'-1}, \bullet, k_{d'+1} \dots k_D}) \prod_{d \neq d'} T^{f(d)}_{i_d k_d}.
\end{aligned} \tag{5}$$

Note that $X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}$ is a vector of size $(N, 1, F)$ and as f_1 is applied only on the features dimension, $f_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D})$ is a vector of size $(N, 1)$. Similarly, $f_2(Y_{k_1 \dots k_{d'-1}, \bullet, k_{d'+1} \dots k_D})$ is a vector of size $(1, N)$. As the gradient is a $(N \times N)$ matrix, the sum between the first two terms should be understood as a broadcasting sum. The same holds for $h_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D})$ and $h_2(Y_{k_1 \dots k_{d'-1}, \bullet, k_{d'+1} \dots k_D})$ of size $(N \times F)$ and $(F \times N)$, the product is a matrix of size $(N \times N)$. The first two double sums can be computed as,

$$\sum_{\forall d \neq d'} \; \sum_{i_d} f_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}) \prod_{d \neq d'} p^{f(d)}_{i_d} + \sum_{\forall d \neq d'} \; \sum_{k_d} f_2(Y_{k_1 \dots k_{d'-1}, \bullet, k_{d'+1} \dots k_D}) \prod_{d \neq d'} q^{f(d)}_{k_d}. \tag{6}$$

There is $D - 1$ sums, each of them operating over N indices. Hence, the complexity of each sum is $O(N^{D-1}N)$. As a consequence, the overall complexity of Equation (6) is $O(N^D)$.

The last term requires several tensor/matrix multiplications as it can be reformulated as,

$$\sum_{\forall d \neq d'} \; \sum_{i_d} h_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}) \left(\sum_{\forall d \neq d'} \; \sum_{k_d} h_2(Y_{k_1 \dots k_{d'-1}, \bullet, k_{d'+1} \dots k_D}) \prod_{d \neq d'} T^{f(d)}_{i_d k_d} \right). \tag{7}$$

The $D - 1$ internal sums can be seen as $D - 1$ tensor/matrices multiplications, each between the index d of the tensor and the second index of the matrix $T^{f(d)}$. Each of these multiplications have a time complexity of $O(N^{D+1})$. We provide an example after the proof, for $D = 3$. If we call this new tensor H, we can reformulate the problem as,

$$\sum_{\forall d \neq d'} \; \sum_{i_d} h_1(X_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}) H_{i_1 \dots i_{d'-1}, \bullet, i_{d'+1} \dots i_D}. \tag{8}$$

This can be seen as a standard multiplication of two tensors on every dimension except on the dimension d' for both tensors. This is equivalent to the multiplication of two matrices of size (N, N^{D-1}) and (N^{D-1}, N). The time complexity is again $O(N \times N^{D-1} \times N) = O(N^{D+1})$.

Finally, the entire time complexity is $O(N^{D+1})$, which is better than the naive $O(N^{2D})$.

Example for $D = 3$. We explain how to compute Equation (7) with $D = 3$ with a $O(N^{3+1})$ time complexity. We will suppose, without loss of generality, that $d' = 3$ in this example. We are interested in the tensor H of Equation (8) or equivalently of the inner sum of Equation (7) for all $(i_1, i_2) \in [\![1, N]\!]^2$,

$$\sum_{k_1=1}^{N} \sum_{k_2=1}^{N} h_2(Y_{k_1,k_2,\bullet}) T^{f(1)}_{i_1,k_1} T^{f(2)}_{i_2,k_2} = H_{i_1,i_2,\bullet}. \tag{9}$$

We rearrange the terms to make the multiplication between a 3-order tensor and a transport plan appear,

$$\forall (i_1, i_2) \in [\![1, N]\!]^2 \ \sum_{k_1=1}^{N} \left(\sum_{k_2=1}^{N} h_2(Y_{k_1,k_2,\bullet}) T^{f(2)}_{i_2,k_2} \right) T^{f(1)}_{i_1,k_1}. \tag{10}$$

For all $(k_1, i_2) \in [\![1, N]\!]^2$ we note $H'_{k_1,i_2,\bullet} = \sum_{k_2=1}^{N} h_2(Y_{k_1,k_2,\bullet}) T^{f(2)}_{i_2,k_2}$. H' can be computed with $N^2 \times N \times N$ operations as a multiplication between a tensor (N, N, N, F) and a matrix (N, N) along the second dimension on both sides. We now have a similar formulation as in Equation (10) but with only one transport plan left,

$$\forall (i_1, i_2) \in [\![1, N]\!]^2 \ \sum_{k_1=1}^{N} H'_{k_1,i_2,\bullet} T^{f(1)}_{i_1,k_1}. \tag{11}$$

We apply the same process, and define for all $(i_1, i_2) \in [\![1, N]\!]^2$, $H_{i_1,i_2,\bullet} = \sum_{k_1=1}^{N} H'_{k_1,i_2,\bullet} T^{f(1)}_{i_1,k_1}$. This new tensor H can be computed with $N^2 \times N \times N$ operations. The difference is that the dimension of the sum is the first one for the tensor H'.

We finally have to compute,

$$\sum_{i_1=1}^{N} \sum_{i_2=1}^{N} h_1(X_{i_1,i_2,\bullet}) H_{i_1,i_2,\bullet}, \tag{12}$$

which is equivalent to Equation (7). We can see it as a multiplication between two tensors of size (N, N, N, F) along the first two dimensions as well as the last dimension. Thus the time complexity is $O(N \times N^2 \times N)$, that is $O(N^4)$.

4 Theoretical results

We now prove that OTT is a distance for weighted tensors.

Theorem 1. *OTT is a distance between weighted tensors represented in canonical form $(X, (p^a)_{a\in[\![1,A]\!]})$ and $(Y, (q^a)_{a\in[\![1,A]\!]})$ for any affectation function f, as long as $\mathcal{L}$ is a proper distance.*

First we properly define the canonical form of a tensor. Note that, here, we will assume that two tensors are equal if, and only if, they are equal in canonical form. This is quite natural in an OT context where the goal is to align datasets. Nevertheless, if we now consider that two tensors are equal if, and only if, they are equal in their original forms then OTT is only a pseudo-distance (as is GW (Chowdhury and Mémoli 2019)) since the identity of indiscernibles would not hold anymore.

Definition 1 (Inspired by the work of Chowdhury and Mémoli (2019))**.** *A weighted tensor $(X, (p^a)_{a\in[\![1,A]\!]})$ associated with an affectation function $f : [\![1, D]\!] \to [\![1, A]\!]$, is in canonical form if it respects the three following rules.*

- *All the weights should be strictly positive. If there is a $a \in [\![1, A]\!]$ and $i \in [\![1, I^a]\!]$ such that $p^a_i = 0$ this weight is deleted along all the corresponding values in the tensor X. This is a natural reduction as if the weight is equal to 0 this is equivalent to not having a point.*
- *There is no duplicated points. If two points are equal, they should be merged together. Two points $(i_a, i'_a) \in [\![1, I_a]\!]^2$ for a fixed $a \in [\![1, A]\!]$ are equal if,*

$$\forall d' \in [\![1, d]\!] | f(d') = a, X_{\bullet_1,\dots,\bullet_{d'-1},i_a,\bullet_{d'+1},\dots,\bullet_D} = X_{\bullet_1,\dots,\bullet_{d'-1},i'_a,\bullet_{d'+1},\dots,\bullet_D}. \tag{13}$$

  Those extracted $(D-1)$-order tensors define entirely each of those points. If two points are equal, then we delete one of them and add the two probabilities $p^a_{i_a}$ and $p^a_{i'_a}$. Notice that we look at every dimension of the tensor associated with the weight p^a and delete simultaneously the points in every dimensions. This result is quite logical in a vector space, if two points are in the same location, they should be merged.
- *Note that, for a given dataset, one may, in each dimension $a \in [\![1, A]\!]$, permute the objects without changing the nature of the dataset. The tensor Y is a permutation of X if there exist a permutation function σ^a for all $a \in [\![1, A]\!]$ such that $Y_{\sigma^{f(1)}(i_1),\dots,\sigma^{f(D)}(i_D)} = X_{i_1,\dots,i_D}$. Then, assuming that we have access to a strict total order on tensors, a weighted dataset is in canonical form if it is the smallest permutation with respect to the given order. An example of strict total order that can be used for this purpose is the lexicographic order.*

Note that, none of those three modifications of the tensor X will change the transport plan nor the OTT distance that depends on X. It means that we never need to explicitly transform a tensor into its canonical form. Instead, it is a theoretical construction that simplifies the proof of Theorem 1.

Proof. We will now prove that OTT is a distance.

Symmetry As $\mathcal{L}$ is symmetric, OTT is also symmetric.

Positiveness As $\mathcal{L}$ and T are always positive, OTT is always positive.

Identity of indiscernibles To prove that $OTT(X, X, (p^a)_{a\in[\![1,A]\!]}, (p^a)_{a\in[\![1,A]\!]}) = 0$, we set every transport plan T^a to the identity matrix. As $\forall x\ \mathcal{L}(x, x) = 0$, the entire sum is 0. But because the minimum is smaller than this particular case and also always positive, we have the desired result $OTT(X, X, (p^a)_{a\in[\![1,A]\!]}, (p^a)_{a\in[\![1,A]\!]}) = 0$.

We will now prove the opposite, let (X, Y) be D-order tensors datasets of sizes $(I_{f(1)} \times ... \times I_{f(D)} \times F, K_{f(1)} \times ... \times K_{f(D)} \times F)$ with weights $(p^a \in \Delta_{I_a}, q^a \in \Delta_{K_a})_{a\in[\![1,A]\!]}$ in a canonical form (Definition 1). The canonical form is inspired from the proof that GW is a distance for graphs (Chowdhury and Mémoli 2019). We suppose that $OTT(X, Y) = 0$, we will show that $X = Y$ and for all $a \in [\![1, A]\!]$, $p^a = q^a$. We will note $(T^a)_{a\in[\![1,A]\!]}$ the optimal transport plans.

We will proceed by contradiction and suppose that there exist two strictly positive values in the same row of a transport plan, more precisely we suppose that there exist $a \in [\![1, A]\!]$, $i \in [\![1, I_a]\!]$, $(k, k') \in [\![1, K_a]\!]^2$ such that $T^a_{i,k} > 0$ and $T^a_{i,k'} > 0$. We fix $d' \in \{d' \in [\![1, D]\!] | f(d') = a\}$ and all the indices $(k_1...k_{d'-1}, k_{d'+1}...k_D) \in ([\![1, K_{f(1)}]\!]...[\![1, K_{f(d'-1)}]\!], [\![1, K_{f(d'+1)}]\!]...[\![1, K_{f(D)}]\!])$. As all the marginals are strictly positive, there exist $(i_1...i_{d'-1}, i_{d'+1}...i_D) \in ([\![1, I_{f(1)}]\!]...[\![1, I_{f(d'-1)}]\!], [\![1, I_{f(d'+1)}]\!]...[\![1, I_{f(D)}]\!])$ such that $T^{f(d)}_{i_d,k_d} > 0$ for all $d \in [\![1, D]\!]$ with $d \neq d'$. As the transport plans are strictly positive on those indices and $T^a_{i,k} > 0$ and $T^a_{i,k'} > 0$, the transport plans product is strictly positive, thus the loss should be equal to 0,

$$\mathcal{L}\left(X_{i_1...i_{d'-1},i,i_{d'+1}...i_D}, Y_{k_1...k_{d'-1},k,k_{d'+1}...k_D}\right) = 0, \tag{14}$$

$$\mathcal{L}\left(X_{i_1...i_{d'-1},i,i_{d'+1}...i_D}, Y_{k_1...k_{d'-1},k',k_{d'+1}...k_D}\right) = 0. \tag{15}$$

As the loss is a distance, we have $Y_{k_1...k_{d'-1},k,k_{d'+1}...k_D} = Y_{k_1...k_{d'-1},k',k_{d'+1}...k_D}$. This is true for all $d' \in \{d' \in [\![1, D]\!] | f(d') = a\}$ and all the indices $(k_1...k_{d'-1}, k_{d'+1}...k_D) \in ([\![1, K_{f(1)}]\!]...[\![1, K_{f(d'-1)}]\!], [\![1, K_{f(d'+1)}]\!]...[\![1, K_{f(D)}]\!])$, thus those two points should have been merged, this is in contradiction with the canonical form assumption. We can do the same for the columns of the transport plans instead of the rows.

We know that a transport plan has only one element in each of its rows and columns, thus it is necessarily a square matrix as all the marginals are strictly positive. Let α be the smallest strictly positive value of the transport plan $(T^a)_{a\in[\![1,A]\!]}$. We also define the permutation matrices P^a associated to each T^a by replacing each strictly positive value in T^a to 1 in P^a. We have $\alpha^{I_{f(a)}} P^a \leq T^a$ elements-wise, thus

$$0 \leq \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}(X_{i_1,...,i_D}, Y_{k_1,...,k_D}) \prod_{d=1}^{D} \alpha^{I_{f(d)}} P^{f(d)}_{i_d,k_d} \tag{16}$$

$$\leq \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}(X_{i_1,...,i_D}, Y_{k_1,...,k_D}) \prod_{d=1}^{D} T^{f(d)}_{i_d,k_d} \tag{17}$$

$$=0. \tag{18}$$

If we note σ^a the permutation function associated with P^a, we have,

$$0 = \sum_{\forall d\ i_d} \mathcal{L}(X_{i_1,...,i_D}, Y_{\sigma^{f(1)}(i_1),...,\sigma^{f(D)}(i_D)}). \tag{19}$$

Since datasets are invariant to points permutations, if both X and Y are in canonical form, they are equal. Thus the transport plans $(T^a)_{a\in[\![1,A]\!]}$ are diagonal matrices. Therefore, the weights p^a and q^a are also equal for all $a \in [\![1, A]\!]$.

Thus, we have $OTT(X, Y, (p^a)_{a\in[\![1,A]\!]}, (q^a)_{a\in[\![1,A]\!]}) = 0 \iff (X, (p^a)_{a\in[\![1,A]\!]}) = (Y, (q^a)_{a\in[\![1,A]\!]})$.

Triangle inequality We now prove the triangle inequality. Let (X, Y, Z) be D-order tensors datasets of sizes $(I_{f(1)} \times ... \times I_{f(D)} \times F, K_{f(1)} \times ... \times K_{f(D)} \times F, J_{f(1)} \times ... \times J_{f(D)} \times F)$ with weights $(p^a \in \Delta_{I_a}, q^a \in \Delta_{K_a}, r^a \in \Delta_{J_a})_{a\in[\![1,A]\!]}$. We suppose $\forall d \in [\![1, D]\!]\ r^{f(d)} > 0$ as we can delete the corresponding $(D-1)$-order tensor in the tensor Z if one term is 0 and it won't change the transport plan nor the distance. We note $(T^a_{xy})_{a\in[\![1,A]\!]}$ the optimal transport plans between X and Y and $(T^a_{yz})_{a\in[\![1,A]\!]}$ the optimal transport plans between Y and Z. Similarly to Redko et al. (2020) we will use the gluing lemma (Villani 2008) to construct a coupling between X and Z,

$$\forall a \in [\![1, A]\!]\ (T^a_{xz}) = (T^a_{xy})\, \mathrm{diag}\left(\frac{1}{q^a}\right) (T^a_{yz}). \tag{20}$$

Let $a \in [\![1, A]\!]$, we show that $(T^a_{xz}) \in \mathcal{U}_{p^a r^a}$:

$$\forall j \in [\![1, J_a]\!] \ \sum_{i=1}^{I_a} (T^a_{xz})_{ij} = \sum_{i=1}^{I_a} \sum_{k=1}^{K_a} (T^a_{xy})_{ik} \frac{1}{q^a_k} (T^a_{yz})_{kj} \tag{21}$$

$$= \sum_{k=1}^{K_a} (T^a_{yz})_{kj} \tag{22}$$

$$= r^a_j, \tag{23}$$

$$\forall i \in [\![1, I_a]\!] \ \sum_{j=1}^{J_a} (T^a_{xz})_{ij} = \sum_{j=1}^{J_a} \sum_{k=1}^{K_a} (T^a_{xy})_{ik} \frac{1}{q^a_k} (T^a_{yz})_{kj} \tag{24}$$

$$= \sum_{k=1}^{K_a} (T^a_{xy})_{ik} \tag{25}$$

$$= p^a_i. \tag{26}$$

We now prove the triangle inequality:

$$OTT(X, Z, (p^a)_{a\in[\![1,A]\!]}, (r^a)_{a\in[\![1,A]\!]}) \tag{27}$$

$$\leq \sum_{i_1,\dots,i_D=1}^{I_{f(1)},\dots,I_{f(D)}} \sum_{j_1,\dots,j_D=1}^{J_{f(1)},\dots,J_{f(D)}} \mathcal{L}(X_{i_1,\dots,i_D}, Z_{j_1\dots,j_D}) \prod_{d=1}^{D} (T^{f(d)}_{xz})_{i_d,j_d} \tag{28}$$

$$= \sum_{i_1,\dots,i_D=1}^{I_{f(1)},\dots,I_{f(D)}} \sum_{j_1,\dots,j_D=1}^{J_{f(1)},\dots,J_{f(D)}} \sum_{k_1,\dots,k_D=1}^{K_{f(1)},\dots,K_{f(D)}} \mathcal{L}\left(X_{i_1,\dots,i_D}, Z_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} \frac{(T^{f(d)}_{xy})_{i_d k_d} (T^{f(d)}_{yz})_{k_d j_d}}{q^{f(d)}_{k_d}} \tag{29}$$

$$\leq \sum_{\forall d\ i_d} \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(X_{i_1,\dots,i_D}, Y_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} \frac{(T^{f(d)}_{xy})_{i_d k_d} (T^{f(d)}_{yz})_{k_d j_d}}{q^{f(d)}_{k_d}}$$
$$+ \sum_{\forall d\ i_d} \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(Y_{k_1,\dots,k_D}, Z_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} \frac{(T^{f(d)}_{xy})_{i_d k_d} (T^{f(d)}_{yz})_{k_d j_d}}{q^{f(d)}_{k_d}} \tag{30}$$

$$= \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}\left(X_{i_1,\dots,i_D}, Y_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} (T^{f(d)}_{xy})_{i_d k_d} \sum_{\forall d\ j_d} \prod_{d=1}^{D} \frac{(T^{f(d)}_{yz})_{k_d j_d}}{q^{f(d)}_{k_d}}$$
$$+ \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(Y_{k_1,\dots,k_D}, Z_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} (T^{f(d)}_{yz})_{k_d j_d} \sum_{\forall d\ i_d} \prod_{d=1}^{D} \frac{(T^{f(d)}_{xy})_{i_d k_d}}{q^{f(d)}_{k_d}} \tag{31}$$

$$= \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}\left(X_{i_1,\dots,i_D}, Y_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} (T^{f(d)}_{xy})_{i_d k_d} \prod_{d=1}^{D} \sum_{j_d=1}^{J_{f(d)}} \frac{(T^{f(d)}_{yz})_{k_d j_d}}{q^{f(d)}_{k_d}}$$
$$+ \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(Y_{k_1,\dots,k_D}, Z_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} (T^{f(d)}_{yz})_{k_d j_d} \prod_{d=1}^{D} \sum_{i_d=1}^{I_{f(d)}} \frac{(T^{f(d)}_{xy})_{i_d k_d}}{q^{f(d)}_{k_d}} \tag{32}$$

$$= \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}\left(X_{i_1,\dots,i_D}, Y_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} (T^{f(d)}_{xy})_{i_d k_d}$$
$$+ \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(Y_{k_1,\dots,k_D}, Z_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} (T^{f(d)}_{yz})_{k_d j_d} \tag{33}$$

$$= OTT(X, Y, (p^a)_{a\in[\![1,A]\!]}, (q^a)_{a\in[\![1,A]\!]}) + OTT(Y, Z, (q^a)_{a\in[\![1,A]\!]}, (r^a)_{a\in[\![1,A]\!]}). \tag{34}$$

In Equation (31), the product/sum inversion is possible as no element in the product depends on $(j_d)_{d\in[\![1,D]\!]}$. Similarly, in Equation (32), the sum/product inversion is allowed as only one term in the product depends on the sum. In addition, each of those sums are equal to 1 by definition of $(T^a_{xy})_{a\in[\![1,A]\!]}$, this leads to Equation (33).

OTT respects the triangle inequality, thus it is a distance. □

We now prove Theorem 2 with a more general formulation that the one proposed in the paper. More precisely, we extend it to the case where the loss is a function of $\mathbb{R}^F \to \mathbb{R}$ with F not necessarily restricted to 1 as supposed in the paper to simplify the notations. First we recall the definition of an OTT barycenter.

Definition 2. *(OTT barycenter) Given $B \in \mathbb{N}$ weighted tensors of sizes $((K_a^b)_{a\in[\![1,A]\!]})_{b\in[\![1,B]\!]}$, $\left(X^b \in \mathbb{R}^{K^b_{f(1)}\dots K^b_{f(D)}\times F}, (q^{a,b} \in \Delta_{K_a^b})_{a\in[\![1,A]\!]}\right)_{b\in[\![1,B]\!]}$. Let $\lambda \in \Delta_B$ be the weights quantifying the importance of each tensor. For fixed size $(I_a)_{a\in[\![1,A]\!]}$ and weights $(p^a \in \Delta_{I_a})_{a\in[\![1,A]\!]}$, the OTT barycenter is defined as follows:*

$$\min_{X\in\mathbb{R}^{I_{f(1)}\cdots I_{f(D)}\times F}} \sum_{b=1}^{B} \lambda_b OTT(X, X^b, (p^a)_{a\in[\![1,A]\!]}, (q^{a,b})_{a\in[\![1,A]\!]}). \tag{35}$$

Theorem 2. *Assume that the loss $\mathcal{L}$ is continuous and can be written as $\mathcal{L}(x,y) = f_1(x) + f_2(y) - h_1(x)h_2(y)$ with four functions $(f_1, f_2 : \mathbb{R}^F \to \mathbb{R})$ and $(h_1, h_2 : \mathbb{R}^F \to \mathbb{R}^F)$ such that the function $(\nabla h_1)^{-1l}\nabla f_1 : \mathbb{R}^F \longrightarrow \mathbb{R}^F$ is invertible, were the $F \times F$ matrix $(\nabla h_1)^{-1l}(x)$ is the left inverse of the $F \times F$ matrix $\nabla h_1(x)$, $\forall x \in \mathbb{R}^F$. We also suppose that, $\forall r \in [\![1, F]\!]\, \mathcal{L}(x,y) \underset{x_r\to\pm\infty}{\longrightarrow} +\infty$. For fixed $(T^{a,b})_{a\in[\![1,A]\!]}$, the optimal solution X^* of Problem (8) reads,*

$$X^*_{i_1,\dots,i_D} = \left((\nabla h_1)^{-1l}\nabla f_1\right)^{-1} \left(\sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K^b_{f(1)},\dots,K^b_{f(D)}} h_2(X^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \frac{T^{f(d),b}_{i_d k_d}}{p^{f(d)}_{i_d}}\right) \tag{36}$$

for all $(i_d \in [\![1, I_{f(d)}]\!])_{d\in[\![1,D]\!]}$.

In particular, when $\mathcal{L}$ is the squared euclidean distance,

$$X^*_{i_1,\dots,i_D} = \sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K^b_{f(1)},\dots,K^b_{f(D)}} X^b_{k_1\dots k_D} \prod_{d=1}^{D} \frac{T^{f(d),b}_{i_d k_d}}{p^{f(d)}_{i_d}}. \tag{37}$$

Proof. For fixed $((T^{a,b})_{a\in[\![1,A]\!]})_{b\in[\![1,B]\!]}$, we are looking for the derivative of Equation (35) with respect to $X_{i_1,\dots,i_D}$ and we equate it to 0 to find the optimal solution. $X_{i_1,\dots,i_D}$ might be a vector if the features dimension is not reduced to 1. Each gradient in the following equation is a vector of size F.

$$0 = \sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K^b_{f(1)},\dots,K^b_{f(D)}} \nabla_{X_{i_1,\dots,i_D}} \mathcal{L}(X_{i_1,\dots,i_D}, X^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{38}$$

$$\iff 0 = \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} \left(\nabla f_1\left(X_{i_1,\dots,i_D}\right) - \nabla h_1(X_{i_1,\dots,i_D}) h_2(X^b_{k_1,\dots,k_D})\right) \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{39}$$

$$\iff 0 = \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} \left(\nabla f_1\left(X_{i_1,\dots,i_D}\right) - \nabla h_1(X_{i_1,\dots,i_D}) h_2(X^b_{k_1,\dots,k_D})\right) \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{40}$$

$$\iff 0 = \nabla f_1\left(X_{i_1,\dots,i_D}\right) \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{41}$$

$$- \nabla h_1(X_{i_1,\dots,i_D}) \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(X^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{42}$$

$$\iff \nabla f_1\left(X_{i_1,\dots,i_D}\right) \sum_{b=1}^{B}\lambda_b \prod_{d=1}^{D} p^{f(d)}_{i_d} = \nabla h_1(X_{i_1,\dots,i_D}) \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(X^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{43}$$

$$\iff (\nabla h_1)^{-1l}(X_{i_1,\dots,i_D})\nabla f_1(X_{i_1,\dots,i_D}) \prod_{d=1}^{D} p^{f(d)}_{i_d} = \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(X^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} T^{f(d),b}_{i_d k_d} \tag{44}$$

$$\iff X_{i_1,\dots,i_D} = \left((\nabla h_1)^{-1l}\nabla f_1\right)^{-1} \left(\sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(X^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \frac{T^{f(d),b}_{i_d k_d}}{p^{f(d)}_{i_d}}\right) \tag{45}$$

There is only one vector $X_{i_1,\dots,i_D}$ found as $(\nabla h_1)^{-1l}\nabla f_1$ is invertible. Thus, if we suppose that the gradient is a maximum or an inflection point, then there is no minimum in $\mathbb{R}^{I_{f(1)}\times\dots\times I_{f(D)}\times F}$ as OTT is continuous. This is in contradiction with the hypothesis that the loss tends to $+\infty$ when $x_r \longrightarrow +\infty$ for any $r \in [\![1, F]\!]$. Thus the value obtained is a minimum.

For the squared euclidean distance, for any $x, y \in \mathbb{R}^F$, $\nabla h_1(x)$ is the identity matrix, $h_2(y) = 2y$ and $\nabla f_1(x) = 2x$. □

5 Experiments

All the experiments were conducted on a linux internal cluster using only CPUs with 16GB of RAM. The important softwares and versions for reproducing the experiments are:

- python version: 3.7.9
- numpy version: 1.19.1
- scipy version: 1.5.2
- sklearn version: 0.23.2
- umap version: 0.4.1

As some experiments depend on randomness, the seeds to reproduce them were fixed using the $numpy.seed$ function. The exact seeds that were randomly fixed are available in the provided code to reproduce the experiments.

5.1 Impact of the order of the tensors on the gradient approximation

In this section, we will analyze the impact of the order of the tensors X and Y on the gradient approximation that would lead to a discussion on the choice of the number of samples M. Note that experiments for the choice of M were already conducted by Kerdoncuff, Emonet, and Sebban (2021), for the Gromov Wasserstein problem. They concluded that choosing M equal to 1 was sufficient to have a good approximation of the gradient, we will see that this is no longer true for $D > 2$.

In these experiments, we generate a pair of graphs, a pair of 3-regular hypergraphs (Berge 1984) and a pair of 4-regular hypergraphs. To do so, we sample 100 points in a 10 dimensional space using 3 Gaussians with random mean and variance, this correspond to the nodes of the hypergraphs. Then the edges of the hypergraphs are created by the D-tuple formed by the D nearest neighbors of each point and we symmetrize the hypergraphs. We give an example for $D = 3$ and the tensor X that represents the hypergraph. For a point x_i, we find the 3 nearest neighbors, noted (x_i, x_j, x_l) and add a value in the tensor X at the position (i, j, l), (i, l, j), (j, i, l), (j, l, i), (l, i, j) and (l, j, i) to ensure the symmetry of the tensor X.

The OTT algorithm, with random marginals, is computed to the iteration S. Then we look at the relative difference in Frobenius norm between the estimated gradient $\widehat{\nabla\mathcal{E}}$ for different values of M and the real one $\nabla\mathcal{E}$: $\frac{\|\nabla\mathcal{E}-\widehat{\nabla\mathcal{E}}\|_F}{\|\nabla\mathcal{E}\|_F}$. As most of the time the real gradient cannot be computed in a reasonable time, we use the value found with 10^6 samples as a reference. Figure 2 shows a clear correlation (with log-log axis) between the number of samples and the estimate of the gradient, with the exception of the very noisy first points. Some points are even omitted if they correspond to a gradient of 0. The latter may happen when M is too small and for high order tensors as the tensors X and Y become sparser. Thus, the 1 sample approximation proposed by Kerdoncuff, Emonet, and Sebban (2021) is no longer possible for $D > 2$, as it often corresponds to a null gradient.

From a time complexity point of view, the main bottleneck in terms of efficiency (excluding the gradient approximation) is the Sinkhorn algorithm with a complexity $O(SN^2)$, independent of D, where S is the number of Sinkhorn iterations which is usually of the order of 100 or 1000. Thus, since the gradient approximation step has a complexity of $O(MN^2)$, M can be chosen of the same order as S to avoid any increase in terms of time complexity. As shown in Figure 2, this is acceptable for $D = 3$ (as in the experiments bellow) but might not be enough for $D = 4$. This difference between all three orders is expected as there is a sum of 10^4 matrices in the gradient for $D = 2$, 10^8 for $D = 3$ and 10^{12} for $D = 4$. On both figures, to have the same relative error of 0.5 on the gradient, it requires approximately 10^5 samples for $D = 4$, 5×10^3 samples for $D = 3$ and 2×10^2 for $D = 2$.

5.2 Domain Adaptation (DA)

Hyperparameters This section will describe the key hyperparameters used by all the methods for reproducibility purpose. Except for the Kullback-Leibler regularization ϵ and class regularization η, we keep the default parameters provided with the code of each algorithm. Note that the code is also attached to this supplementary material.

S-GWL

- Loss: squared euclidean distance
- Outer iteration: 4000
- Max iteration for the barycenter: 4
- We use the automatic update of the marginal p proposed by S-GWL. Similar results were obtained without any automatic update of the marginal.

OTT

- Loss: squared euclidean distance
- Number of samples M: 1000
- Number of iterations S: 1000
- Number of outer iterations of OTDA (Courty et al. 2016): 10
- Number of inner iterations of OTDA (Courty et al. 2016): 200

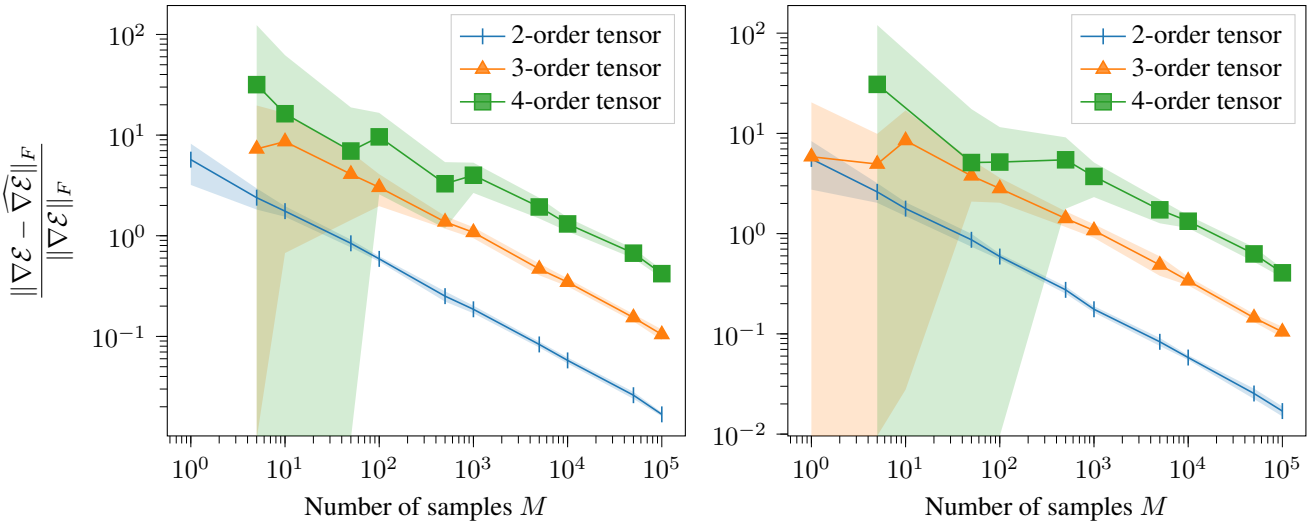

Figure 2: Relative error of the gradient approximation for an increasing number of samples M. The gradient was approximated after 100 (left) and 1000 (right) iterations of the algorithm OTT. The mean and standard deviation over 10 runs are displayed. The points corresponding to a gradient of 0 are omitted. This may happen with a small number of samples M and for high order tensors as the tensors X and Y become sparser.

Co-OT

- Loss: squared euclidean distance
- Number of iterations S: 10
- Number of outer iterations of OTDA (Courty et al. 2016): 10
- Number of inner iterations of OTDA (Courty et al. 2016): 200

GW

- Loss: squared euclidean distance
- Number of iterations S: 1000
- Number of outer iterations of OTDA (Courty et al. 2016): 10
- Number of inner iterations of OTDA (Courty et al. 2016): 200

SVM

- Square L2 penality C: 1.0 (Any value will give the same result has there is only 1 point per class)

Figures for each parameter and datasets In this section we present the Figures evoked in Section 6.1 for each dataset instead of the average. We also display in Figures 4 to 10 the values of the various computed distances rescaled between 0 and 1 for every transport method to more clearly demonstrate the correlation between the distance and the target accuracy. This supports the choice of using the distance to select the hyperparameters on this Domain Adaptation task. In addition we also plot similar figures related to the class regularization η and quickly analyze the performance for an increasing number of samples for the gradient estimation.

Increasing number of samples Figure 3 shows that having a higher number of samples M for the estimation of the gradient improves the performances of OTT.

Kullback-Leibler and classes regularization We show on Figures 4 to 10 the impact of the Kullback-Leibler and classes regularizations.

Increasing the supervision We show in Figures 11 to 17 the impact of an increase in terms of supervision with respect to the pairwise information that the same users rated old and new movies and the number of labels available for the target.

5.3 Comparison based clustering using OTT barycenter

Theoretical link between t-STE and OTT In this section, we show that t-STE (Van Der Maaten and Weinberger 2012) is just a OTT_{111} barycenter with a fixed transport plan. While in the closed form presented in Theorem 2 we minimized directly

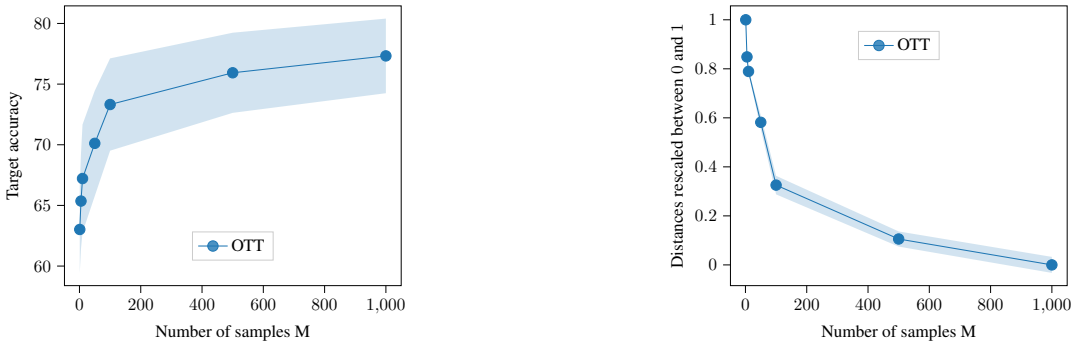

Figure 3: Average target accuracy and distance over all the dataset for an increasing number of samples for the estimation of the gradient.

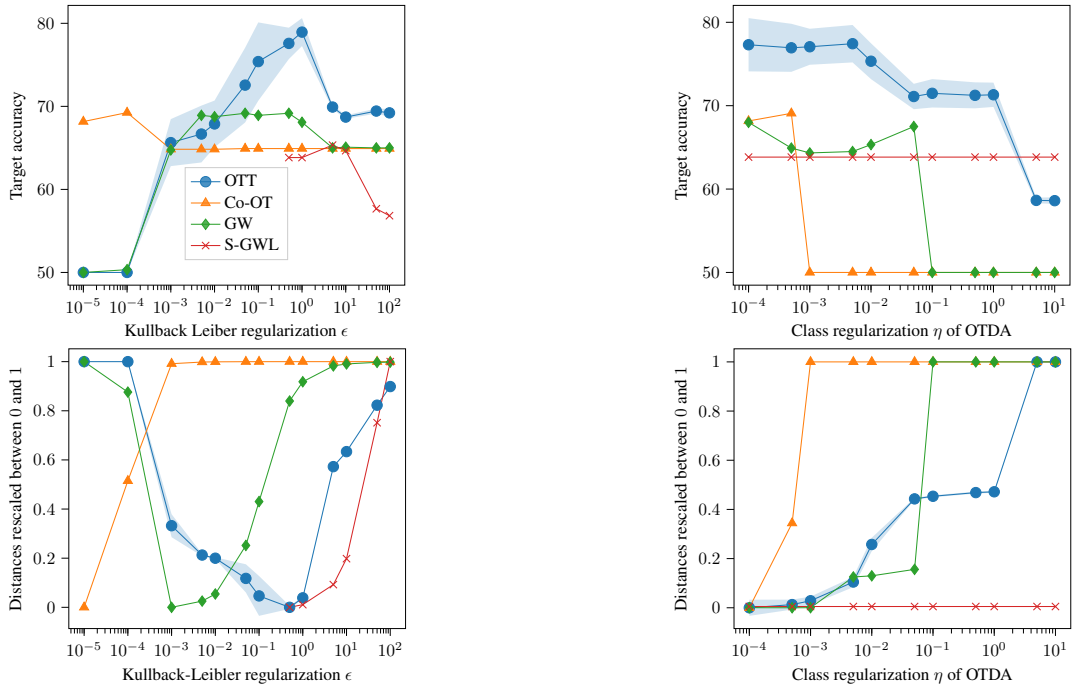

Figure 4: (**Top row**) Average target accuracy for an increasing Kullback-Leibler and classes regularization values. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values. The distances have been re-scaled between 0 and 1.

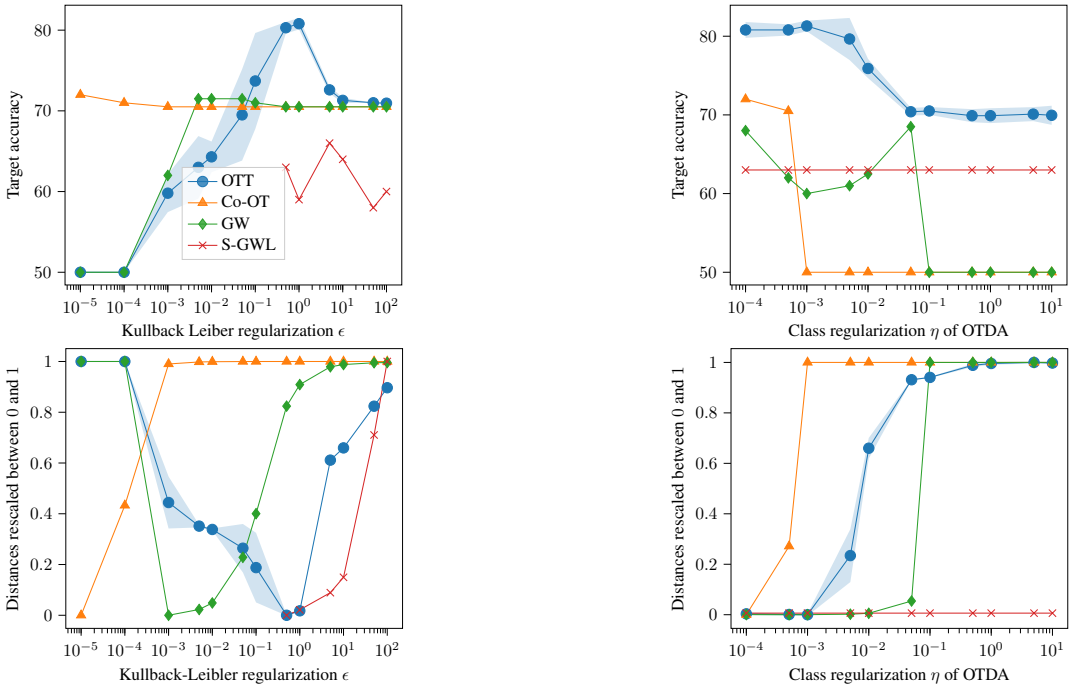

Figure 5: (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1.

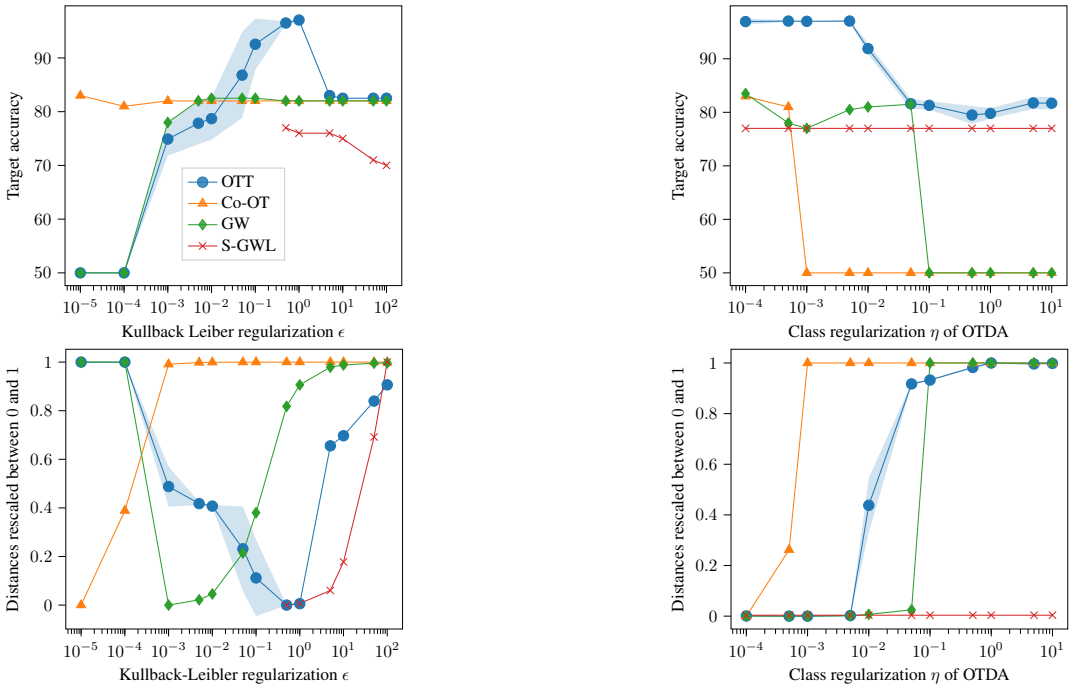

Figure 6: (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1.

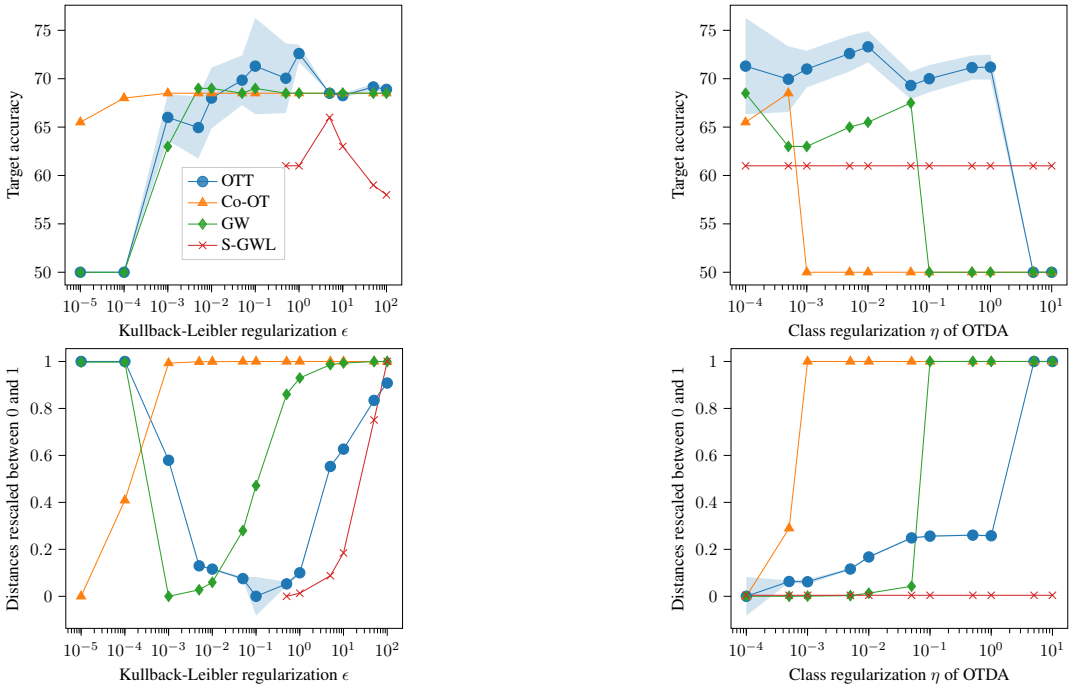

Figure 7: (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. The distances have been re-scaled between 0 and 1.

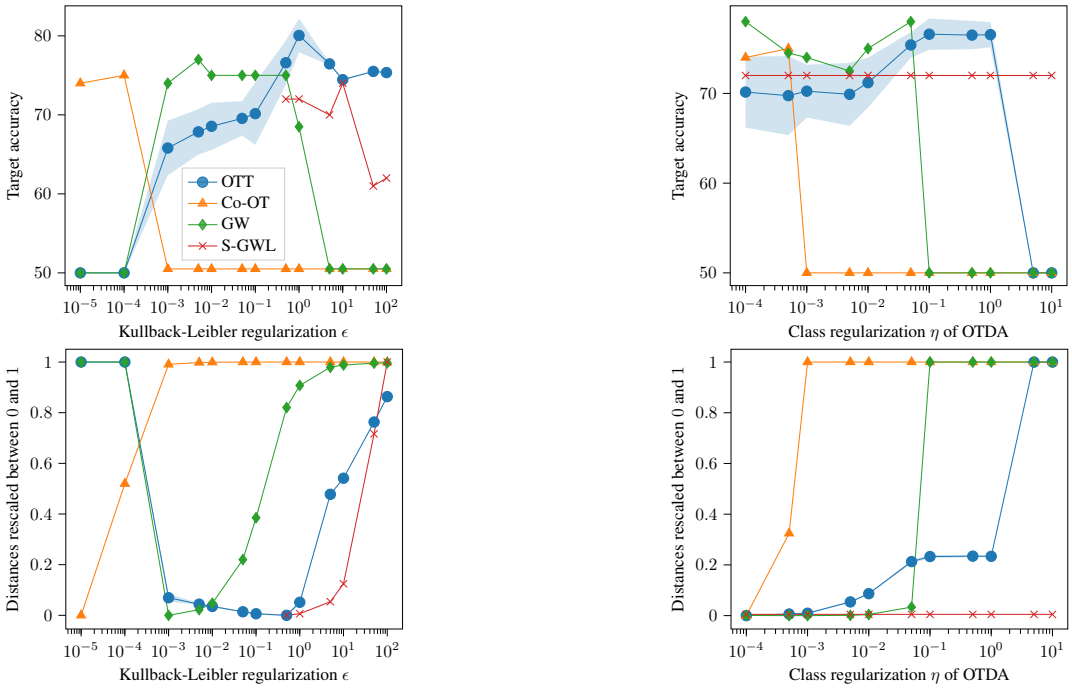

Figure 8: (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. The distances have been re-scaled between 0 and 1.

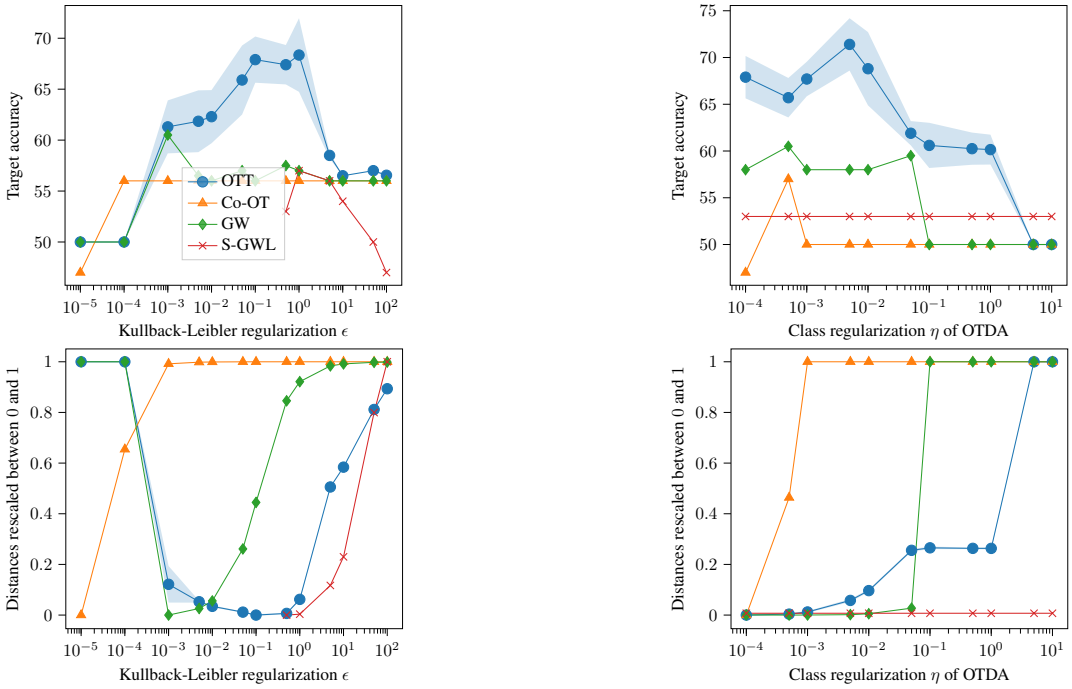

Figure 9: (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. The distances have been re-scaled between 0 and 1.

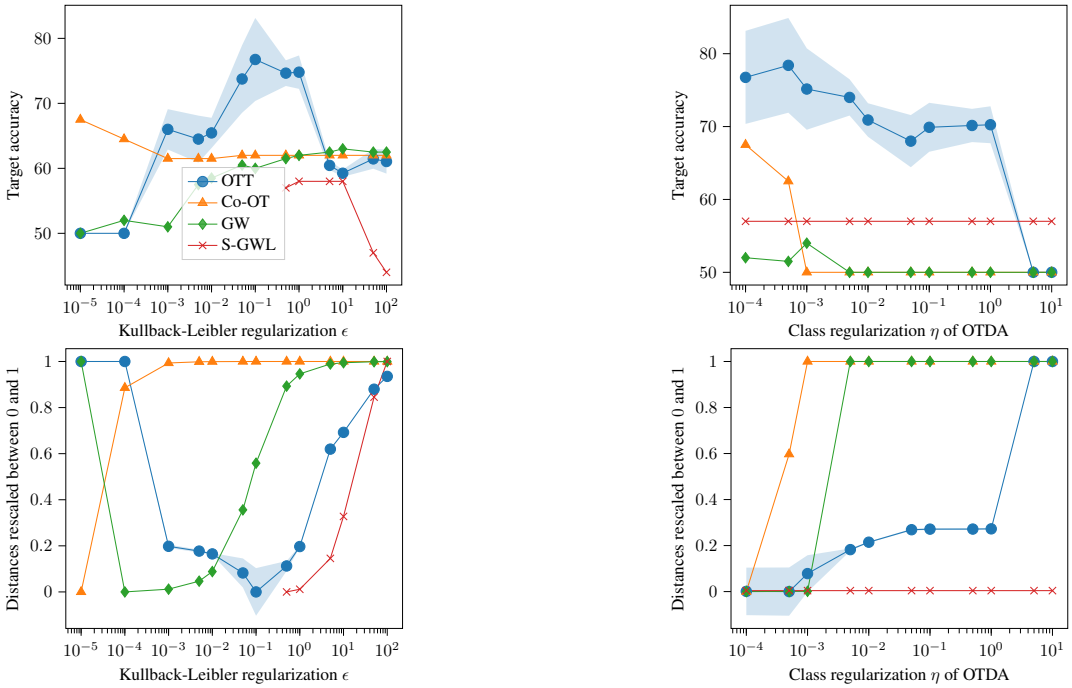

Figure 10: (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *War/Western*. The distances have been re-scaled between 0 and 1.

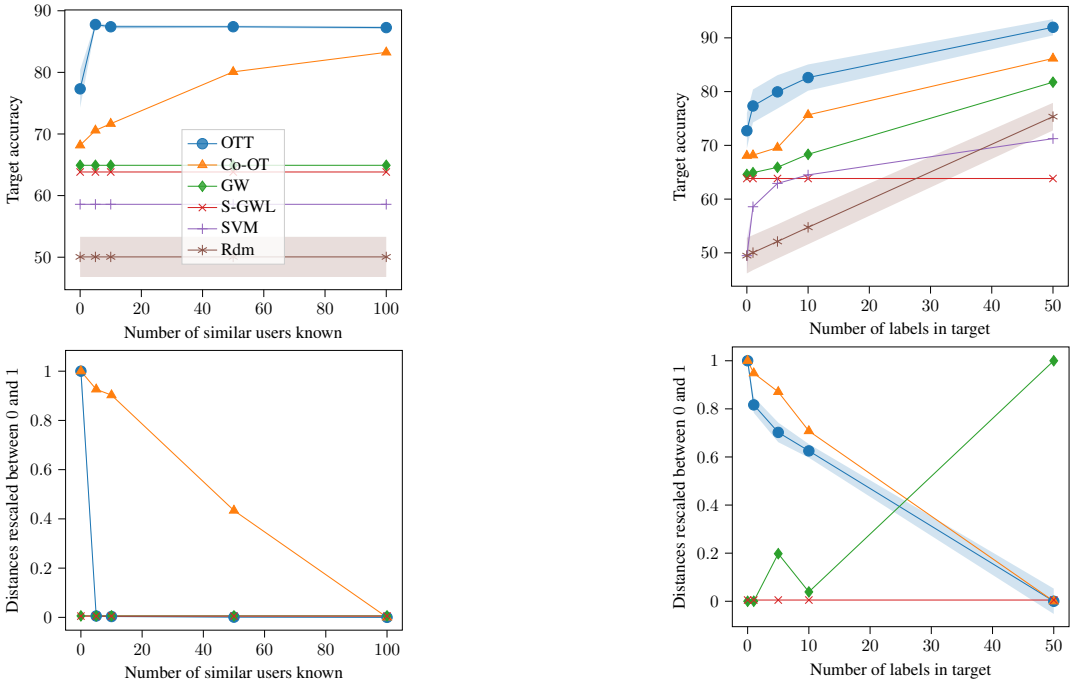

Figure 11: (**Top row**) Average target accuracy for an increasing number of users known and label available in the target domain. (**Bottom row**) Distances of the different methods for an increasing number of users known and label available in the target domain. The distances have been re-scaled between 0 and 1.

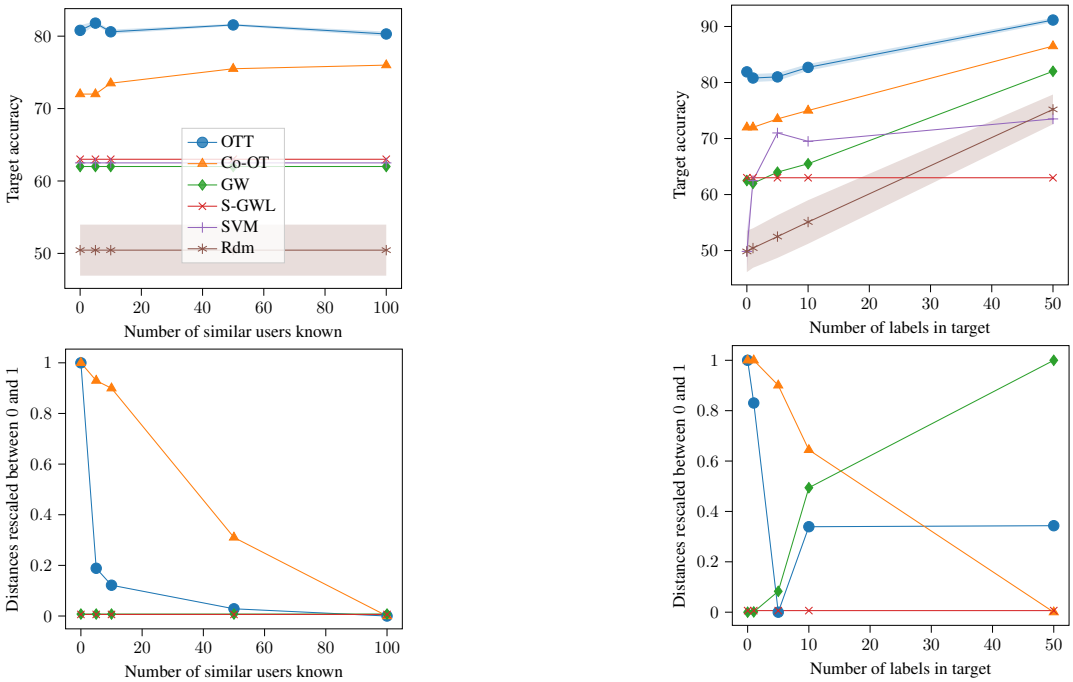

Figure 12: (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1.

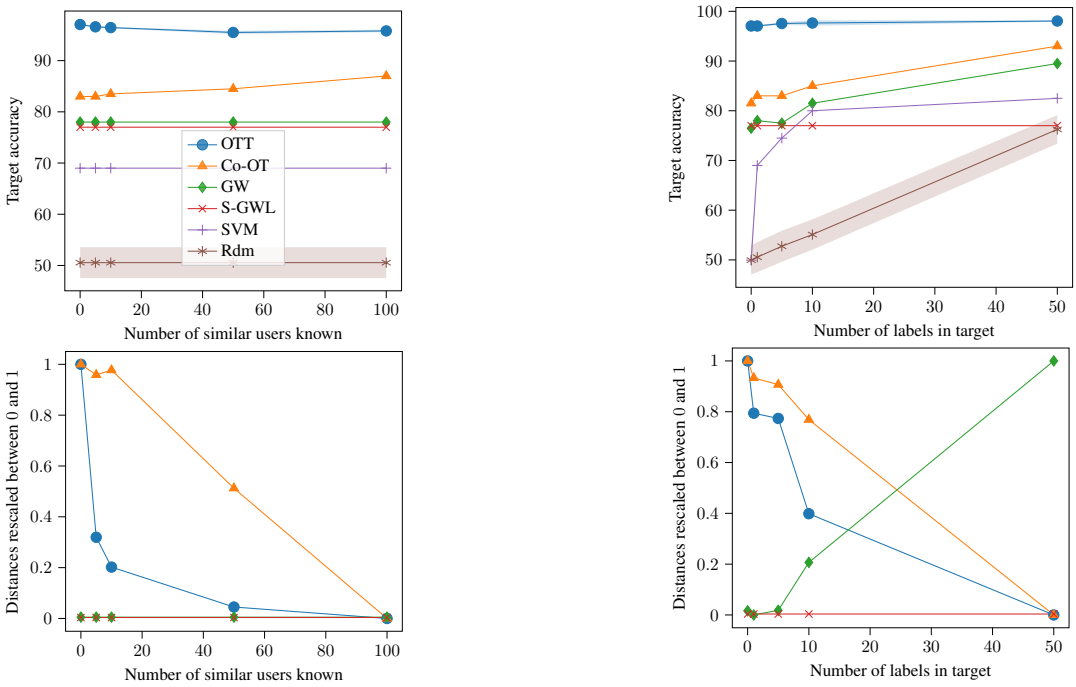

Figure 13: (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1.

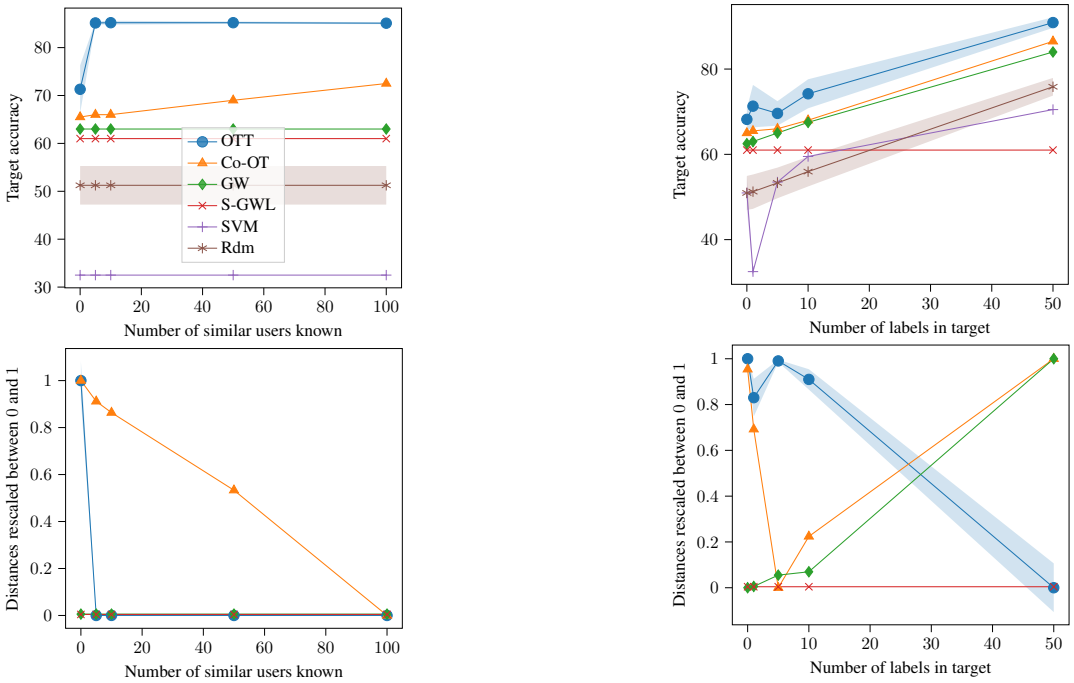

Figure 14: (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. The distances have been re-scaled between 0 and 1.

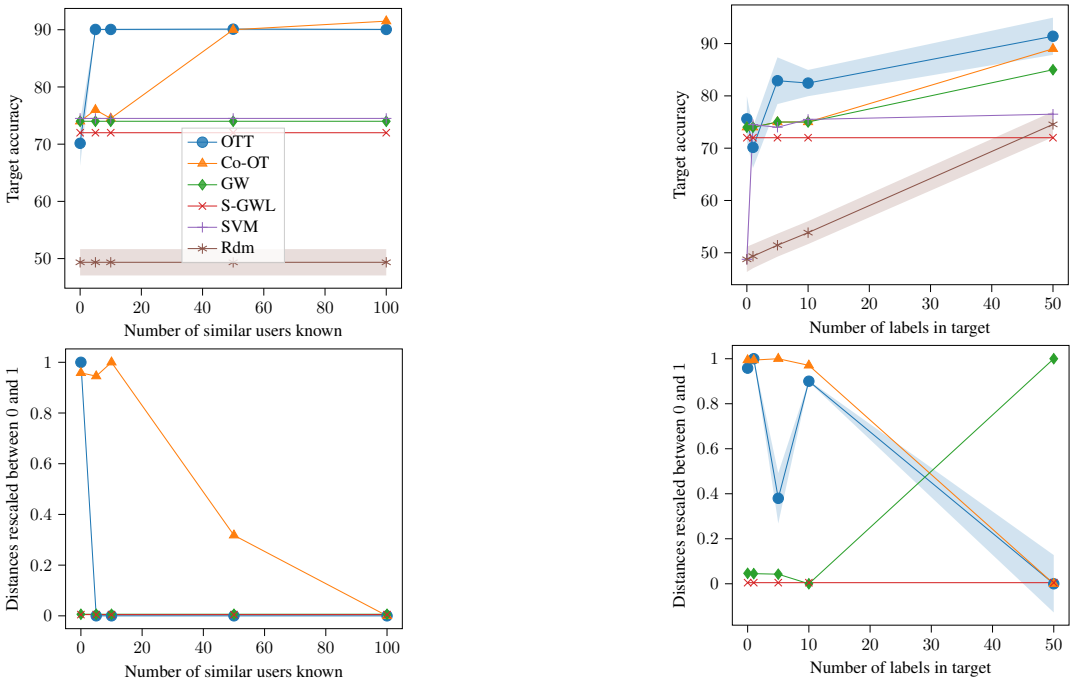

Figure 15: (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. The distances have been re-scaled between 0 and 1.

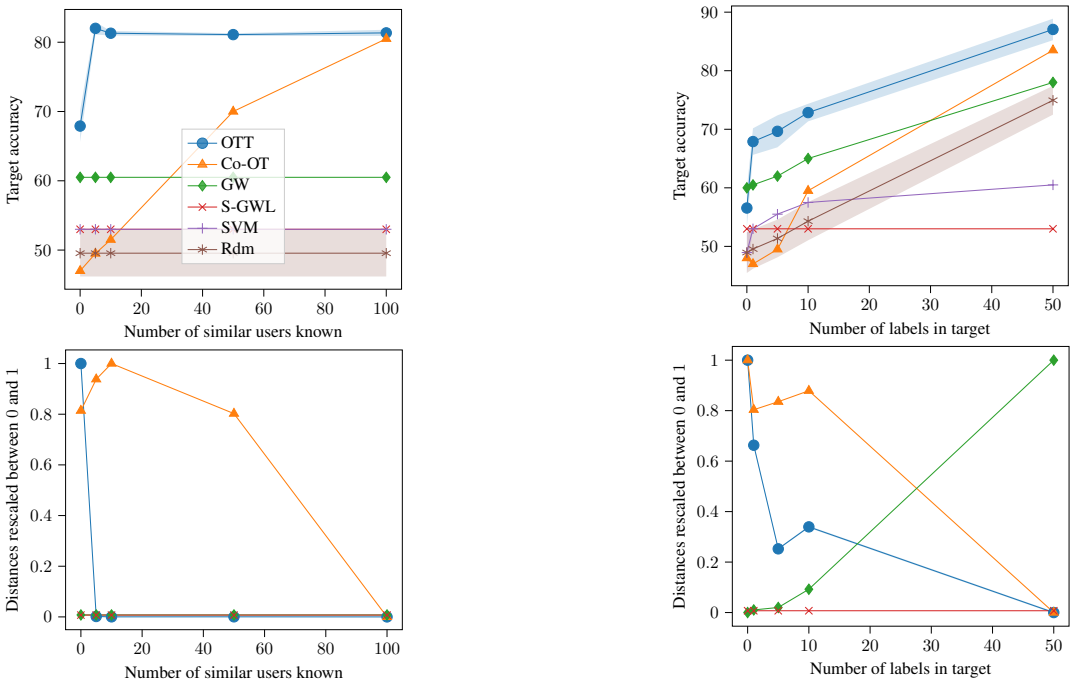

Figure 16: (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. The distances have been re-scaled between 0 and 1.

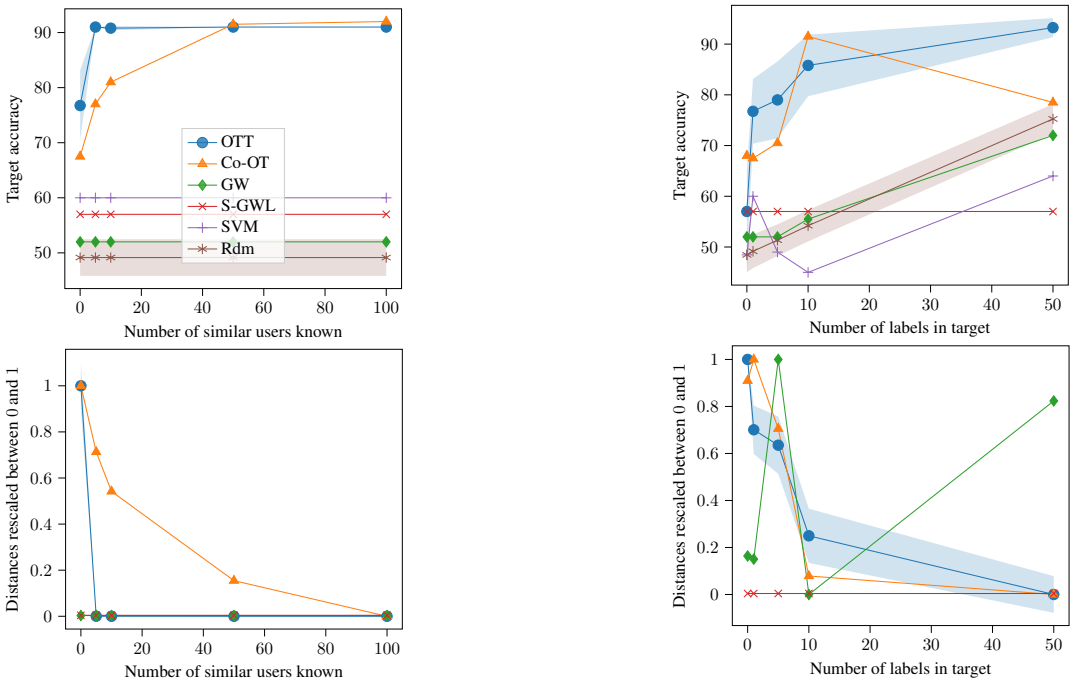

Figure 17: (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Children's/Animation* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Children's/Animation* and *War/Western*. The distances have been re-scaled between 0 and 1.

the value of the tensor X, we could also minimize another related variable, such as points in a vector space x which generate $X(x)$. This is the principle of many triplet embedding methods which are looking for points x in a vector space that respect as closely as possible the triplets provided in X^1. Theorem 3 shows that a widely used method for triplet embedding, t-STE (Van Der Maaten and Weinberger 2012), is a particular case of OTT barycenter.

Theorem 3. *We suppose that $\mathcal{T}$ is a list of triplets which can also be represented with a cubic 3-order tensor X^1 of size (I_1, I_1, I_1) with, at the position i_1, i_2, i_3 the number of occurrences of the triplet (i_1, i_2, i_3) in $\mathcal{T}$. Let $x = (x_i)_{i \in [\![1, I_1]\!]}$ be I_1 points in a vector space $\mathbb{R}^q$. We can then set the tensor X to the t-STE or the STE formula as given in (Van Der Maaten and Weinberger 2012), for STE: $X_{i_1,i_2,i_3} = \frac{exp(-\|x_{i_1} - x_{i_2}\|^2)}{exp(-\|x_{i_1} - x_{i_2}\|^2) + exp(-\|x_{i_1} - x_{i_3}\|^2)}$. If $\mathcal{L}$ is the cross-entropy and f is the constant function (all the 3 transport plans are similar), then STE is a particular case of OTT with the identity matrix* $\mathbf{Id}$ *(divided by I_1) of size I_1 as the transport plan,*

$$\max_{x \in \mathbb{R}^{I_1 \times q}} \sum_{(i_1,i_2,i_3) \in \mathcal{T}} \log(X_{i_1 i_2 i_3}(x)) = I_1^3 \min_{x \in \mathbb{R}^{I_1 \times q}} \sum_{b=1}^{1} \mathcal{E}\left(X(x), X^1, \frac{\mathbf{Id}}{I_1}\right). \tag{46}$$

Proof. We start from the STE formulation and reformulate the problem,

$$\max_{x\in\mathbb{R}^{I_1\times q}} \sum_{(i_1,i_2,i_3)\in\mathcal{T}} \log(X_{i_1,i_2,i_3}(x)) \tag{47}$$

$$= \max_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} X^1_{i_1,i_2,i_3} \log(X_{i_1,i_2,i_3}(x)) \tag{48}$$

$$= \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} -X^1_{i_1,i_2,i_3} \log(X_{i_1,i_2,i_3}(x)) \tag{49}$$

$$= \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \sum_{k_1,k_2,k_3=1}^{I_1,I_1,I_1} -X^1_{i_1,i_2,i_3} \log(X_{k_1,k_2,k_3}(x)) \mathbf{Id}_{i_1,k_1}\mathbf{Id}_{i_2,k_2}\mathbf{Id}_{i_3,k_3} \tag{50}$$

$$= \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \sum_{k_1,k_2,k_3=1}^{I_1,I_1,I_1} \mathcal{L}\left(X^1_{i_1,i_2,i_3}, X_{k_1,k_2,k_3}(x)\right) \mathbf{Id}_{i_1,k_1}\mathbf{Id}_{i_2,k_2}\mathbf{Id}_{i_3,k_3} \tag{51}$$

$$= I_1^3 \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \sum_{k_1,k_2,k_3=1}^{I_1,I_1,I_1} \mathcal{L}\left(X^1_{i_1,i_2,i_3}, X_{k_1,k_2,k_3}(x)\right) \frac{\mathbf{Id}_{i_1,k_1}}{I_1}\frac{\mathbf{Id}_{i_2,k_2}}{I_1}\frac{\mathbf{Id}_{i_3,k_3}}{I_1}. \tag{52}$$

Note that, any permutation are equivalent to the identity as x_i and x_j can be exchanged. In addition, the identity matrix is not necessarily the optimal value, thus the OTT barycenter might lead to a better optimum, but loses the pairwise connection that can be useful for interpretation. □

Interestingly, this new interpretation of t-STE in the light of Theorem 3 gives a good theoretical justification of the choice of the log function in t-STE which is nothing more than a cross entropy between 3D-tensor. One specificity of OTT barycenter, compared to t-STE, is that the size of the barycenter X is not necessarily the size of X^1. Thus, it is notably possible to use a small size for X which will aggregate similar points from X^1. This idea has been used advantageously in the experiment to obtain a direct clustering of a triplet dataset.

Hyperparameters used In this section we details the hyperparameters used in the experiment.

OTT

- Loss type: squared euclidean distance
- Number of samples M: 100
- Number of iterations S: 500
- Kullback-Leibler regularization: 0.1

AddS3

- Number of iterations of the k-means: 300

t-STE

- Degrees of freedom in student T kernel: 0
- Number of iterations of k-means: 300
- Maximum number of iterations: 1000
- L2 regularization constant: 0

Detailed tables for the experiment We display the ARI for comparison based clustering in Table 1 which is similar to the one provided in the paper, without averaging the different classes. This table shows a similar behaviour, OTT is very often better than AddS3$_s$ while being comparable to t-STE$_s$ on most datasets.

Additionally, we present several experiments in the balanced case in Table 2 where the proportion of classes are similar. In this case, OTT is slightly worse than the two other baselines while still being competitive. This is not surprising since OTT, contrary to AddS3 and t-STE, was not specifically designed to handle triplet comparisons. Instead, it is a general purpose Optimal Transport formulation between tensors of potentially high order that can be used to solve multiple kind of tasks.

Table 1: ARI for unbalanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 runs.

nb. examples per class—classes	AddS3	AddS3$_s$	t-STE	t-STE$_s$	OTT
200,20,20—0,1,2	0.35±0.02	0.89±0.19	0.36±0.02	**0.97**±0.02	0.96±0.02
200,20,20—1,3,4	0.35±0.02	0.92±0.03	0.34±0.01	0.94±0.03	**0.94**±0.03
200,20,20—1,9,4	0.35±0.02	0.92±0.03	0.34±0.01	0.94±0.03	**0.94**±0.03
200,20,20—2,9,8	0.32±0.03	0.8±0.05	0.72±0.21	**0.82**±0.06	0.8±0.06
200,20,20—3,4,0	0.72±0.32	0.52±0.25	**0.94**±0.02	0.88±0.18	0.87±0.06
200,20,20—3,4,9	0.38±0.06	0.83±0.18	0.39±0.16	**0.9**±0.05	0.9±0.04
200,20,20—5,7,0	0.57±0.34	0.76±0.35	0.93±0.04	**0.94**±0.04	0.93±0.05
200,20,20—6,4,8	0.35±0.03	0.89±0.2	0.33±0.01	0.96±0.02	**0.97**±0.02
200,20,20—6,8,5	0.36±0.03	0.81±0.24	0.4±0.18	0.93±0.04	**0.94**±0.03
200,20,20—7,1,9	0.53±0.35	0.69±0.22	**0.86**±0.05	0.8±0.06	0.8±0.05
30,3,1—0,1,2	0.32±0.08	0.8±0.33	0.35±0.17	0.93±0.12	**0.93**±0.12
30,3,1—1,3,4	0.28±0.08	0.85±0.33	0.37±0.21	**1.0**±0.01	0.96±0.09
30,3,1—1,9,4	0.27±0.1	0.96±0.09	0.3±0.03	**0.99**±0.01	0.95±0.09
30,3,1—2,9,8	0.29±0.08	0.87±0.15	0.35±0.17	**0.9**±0.14	0.84±0.15
30,3,1—3,4,0	0.2±0.1	0.61±0.24	**0.9**±0.12	0.85±0.15	0.84±0.21
30,3,1—3,4,9	0.29±0.06	0.84±0.21	0.33±0.17	**0.87**±0.15	0.84±0.21
30,3,1—5,7,0	0.25±0.06	0.7±0.01	0.85±0.22	0.88±0.15	**0.9**±0.15
30,3,1—6,4,8	0.31±0.03	0.94±0.12	0.29±0.03	**0.96**±0.09	0.9±0.13
30,3,1—6,8,5	0.31±0.05	0.9±0.14	0.29±0.03	**0.97**±0.1	0.9±0.14
30,3,1—7,1,9	0.27±0.23	0.76±0.12	0.84±0.23	0.81±0.15	**0.84**±0.16
30,3,3—0,1,2	0.4±0.11	0.85±0.19	0.38±0.11	**0.89**±0.12	0.83±0.19
30,3,3—1,3,4	0.37±0.05	0.85±0.24	0.35±0.03	**0.99**±0.01	0.93±0.18
30,3,3—1,9,4	0.37±0.05	0.85±0.25	0.36±0.03	**0.99**±0.01	0.92±0.18
30,3,3—2,9,8	0.34±0.06	0.8±0.24	0.38±0.17	**0.87**±0.12	0.81±0.24
30,3,3—3,4,0	0.47±0.38	0.5±0.25	0.94±0.07	**0.95**±0.09	0.8±0.17
30,3,3—3,4,9	0.38±0.1	0.83±0.21	0.37±0.17	**0.89**±0.12	0.85±0.19
30,3,3—5,7,0	0.27±0.21	0.73±0.29	0.91±0.09	**0.95**±0.09	0.89±0.19
30,3,3—6,4,8	0.36±0.04	0.89±0.19	0.36±0.03	**0.96**±0.07	0.92±0.18
30,3,3—6,8,5	0.36±0.04	0.89±0.19	0.36±0.03	**0.98**±0.07	0.93±0.18
30,3,3—7,1,9	0.38±0.27	0.59±0.23	**0.81**±0.19	0.77±0.17	0.8±0.2
300,30,10—0,1,2	0.3±0.02	0.96±0.02	0.3±0.01	0.96±0.03	**0.96**±0.02
300,30,10—1,3,4	0.29±0.01	0.93±0.02	0.28±0.0	0.89±0.1	**0.93**±0.03
300,30,10—1,9,4	0.29±0.01	**0.94**±0.03	0.28±0.01	0.93±0.03	0.93±0.04
300,30,10—2,9,8	0.28±0.03	**0.8**±0.05	0.42±0.26	0.77±0.04	0.73±0.06
300,30,10—3,4,0	0.25±0.04	0.59±0.23	**0.92**±0.03	0.91±0.06	0.85±0.06
300,30,10—3,4,9	0.3±0.04	0.85±0.1	0.4±0.22	0.9±0.04	**0.91**±0.03
300,30,10—5,7,0	0.3±0.19	0.68±0.02	0.9±0.04	**0.91**±0.04	0.88±0.06
300,30,10—6,4,8	0.3±0.02	0.93±0.09	0.28±0.01	0.92±0.09	**0.96**±0.03
300,30,10—6,8,5	0.3±0.02	0.9±0.1	0.34±0.19	**0.94**±0.03	0.93±0.03
300,30,10—7,1,9	0.25±0.03	0.69±0.04	0.65±0.27	0.76±0.05	**0.78**±0.05
AVG	0.34±0.09	0.81±0.16	0.51±0.1	**0.91**±0.08	0.89±0.1

Table 2: ARI for balanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 different combinations of classes, each run 10 times.

nb. examples per class	AddS3	AddS3$_s$	t-STE	t-STE$_s$	OTT
10,10,10	0.9±0.08	0.9±0.1	0.88±0.13	**0.93**±0.11	0.9±0.11
20,20,20	0.88±0.06	0.87±0.07	0.86±0.11	**0.91**±0.08	0.86±0.08
30,30,30	0.91±0.05	0.9±0.05	0.87±0.1	**0.92**±0.07	0.91±0.07
40,40,40	**0.92**±0.06	0.91±0.05	0.86±0.1	**0.92**±0.06	0.9±0.06
50,50,50	**0.92**±0.06	**0.92**±0.06	0.85±0.11	**0.92**±0.06	0.9±0.06
60,60,60	**0.92**±0.05	0.91±0.04	0.86±0.09	**0.92**±0.05	0.9±0.05
70,70,70	**0.92**±0.05	**0.92**±0.05	0.85±0.07	**0.92**±0.05	0.88±0.05
80,80,80	**0.92**±0.06	0.91±0.03	0.85±0.1	**0.92**±0.06	0.86±0.06
90,90,90	**0.92**±0.11	**0.92**±0.11	0.87±0.09	**0.92**±0.11	0.8±0.11
100,100,100	**0.92**±0.13	**0.92**±0.13	0.85±0.09	**0.92**±0.13	0.73±0.13
AVG	0.91±0.04	0.91±0.05	0.86±0.1	**0.92**±0.05	0.86±0.08

References

Beck, A.; and Teboulle, M. 2003. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*.

Berge, C. 1984. *Hypergraphs: combinatorics of finite sets*. Elsevier.

Chowdhury, S.; and Mémoli, F. 2019. The Gromov–Wasserstein distance between networks and stable network invariants. *Information and Inference: A Journal of the IMA*.

Courty, N.; Flamary, R.; Tuia, D.; and Rakotomamonjy, A. 2016. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9): 1853–1865.

Kerdoncuff, T.; Emonet, R.; and Sebban, M. 2021. Sampled Gromov Wasserstein. *Machine Learning*.

Peyré, G.; Cuturi, M.; and Solomon, J. 2016. Gromov-wasserstein averaging of kernel and distance matrices. In *International Conference on Machine Learning*.

Redko, I.; Vayer, T.; Flamary, R.; and Courty, N. 2020. CO-Optimal Transport. In *Advances in Neural Information Processing Systems*.

Van Der Maaten, L.; and Weinberger, K. 2012. Stochastic triplet embedding. In *2012 IEEE International Workshop on Machine Learning for Signal Processing*. IEEE.

Villani, C. 2008. *Optimal transport: old and new*. Springer Science & Business Media.

Xu, H.; Luo, D.; Zha, H.; and Duke, L. C. 2019. Gromov-Wasserstein Learning for Graph Matching and Node Embedding. In *International Conference on Machine Learning*.