Metric Learning in Optimal Transport for Domain Adaptation

Tanguy Kerdoncuff, Rémi Emonet, Marc Sebban

To cite this version:

Tanguy Kerdoncuff, Rémi Emonet, Marc Sebban. Metric Learning in Optimal Transport for Domain Adaptation. International Joint Conference on Artificial Intelligence, Jan 2021, Kyoto, Japan. ujm-02611800v3

HAL Id: ujm-02611800

https://hal-ujm.archives-ouvertes.fr/ujm-02611800v3

Submitted on 15 Jul 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Metric Learning in Optimal Transport for Domain Adaptation

Tanguy Kerdoncuff, Rémi Emonet and Marc Sebban
Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien
UMR 5516, F-42023, SAINT-ETIENNE, France
{tanguy.kerdoncuff, remi.emonet, marc.sebban}@univ-st-etienne.fr

Abstract

Domain Adaptation aims at benefiting from a labeled dataset drawn from a source distribution to learn a model from examples generated according to a different but related target distribution. Creating a domain-invariant representation between the two source and target domains is the most widely technique used. A simple and robust way to perform this task consists in (i) representing the two domains by subspaces described by their respective eigenvectors and (ii) seeking a mapping function which aligns them. In this paper, we propose to use Optimal Transport (OT) and its associated Wasserstein distance to perform this alignment. While the idea of using OT in domain adaptation is not new, the original contribution of this paper is two-fold: (i) we derive a generalization bound on the target error involving several Wasserstein distances. This prompts us to optimize the ground metric of OT to reduce the target risk. (ii) From this theoretical analysis, we design an algorithm (MLOT) which optimizes a Mahalanobis distance leading to a transportation plan that adapts better. Experiments demonstrate the effectiveness of this original approach.

1 Introduction

Domain adaptation (DA) has been shown to be very effective in many real world applications, e.g., in computer vision, medical diagnosis, or recommender systems, to cite a few. The main idea is to use labeled data of a source domain to improve the performance of a classifier deployed on a related target domain which suffers from a lack of labeled examples. In this paper, we address a complex setting, called unsupervised DA, where there is only unlabeled data available from the target distribution.

Different approaches have been proposed to tackle this problem, some of them coming with theoretical guarantees (see, e.g. the survey [Redko et al., 2019b]). One classical way is to learn a common latent space where the distribution shift is smaller. For instance, the Subspace Alignment algorithm (SA) [Fernando et al., 2013] learns a classifier in a subspace obtained after a linear alignment of the source and target eigenvectors. In a similar manner, the Correlation Alignment (CORAL) [Sun et al., 2016] uses the covariance of the source and target distributions to reduce the shift, while Transfer Component Analysis (TCA) [Pan et al., 2011] looks for common features between the two domains. Some other works directly learn the target labels but do not gather the two distributions in a common feature space. This is the case of MEDA [Wang et al., 2018] which learns a domain-invariant classifier in Grassman manifold. On the other hand, DA with deep learning has received much attention during the past decade from the computer vision community leading to a substantial amount of research to address visual tasks for which a large amount of training data or a pre-trained model is available (see, e.g. the survey [Wang and Deng, 2018]).

More recently, Optimal Transport (OT) has been shown to be a very promising tool to perform DA tasks. OT consists in mapping two source and target probability measures with a minimal cost of transportation associated to the so-called Wasserstein distance. Beyond its use in deep learning to solve visual DA tasks (see, e.g. [Sun and Saenko, 2016; Bhushan Damodaran et al., 2018]), this idea of reducing the shift by OT has been exploited in a more generic DA setting by the algorithm OTDA [Courty et al., 2017b]. OTDA modifies the original Kantorovich optimization problem by resorting to a regularization preventing the transportation plan from moving two source points of different labels onto the same target example. Then, a classifier can be learned from the labeled source data and deployed over the target distribution. Based on this work, [Courty et al., 2017a] ensures that the final classifier is coherent with the transportation plan.

Inspired from both SA and OTDA, our contribution aims at using OT for domain adaptation by aligning the source and target subspaces. The main conjecture we formulate in this paper is that the Euclidean distance usually used as the cost matrix in OT may not be the best metric to perform the adaptation. While learning a better metric (especially a Mahalanobis distance) in OT has been recently studied [Deshpande et al., 2019; Genevay et al., 2018; Paty and Cuturi, 2019; Cuturi and Avis, 2014], optimizing such a ground metric to address DA tasks has not receive attention yet. We fill this gap from both a theoretical and an algorithmic perspective. First, we formally establish a relation between the target error and the magnitude of different Wasserstein distances. This prompts us to see the Wasserstein distance as a parameterized
metric that might be minimized, leading to a better transportation plan for DA. We also formally make a link between a PCA and the minimization of the Wasserstein distance. Based on this theoretical analysis, we propose MLOT, an algorithm which optimizes a Mahalanobis distance that improves the Optimal Transport between the source and target subspaces generated by a PCA. Unlike OTDA which does not change the feature space, MLOT jointly optimizes (i) the dimensionality reduction of the source domain, (ii) the transportation plan between the source and the target and (iii) the underlying metric used in the transportation. The intuition behind MLOT is illustrated in Figure 1. The original source and target examples are represented on the left. The second figure gives evidence on the advantage of performing a PCA before learning jointly the ground metric and the transportation plan.

The rest of the paper is organized as follows: Section 2 reminds the main principles of OT, DA and Metric Learning. Section 3 is dedicated to the theoretical contribution of our paper which leads to the design of our MLOT algorithm in Section 4. An extensive experimental study is presented in Section 5 before our conclusion in Section 6.

2 Optimal Transport and Metric Learning

In this section, we briefly introduce Optimal Transport, its use in OTDA, and Metric Learning.

2.1 Optimal Transport and OTDA

Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a feature space. We consider here a source distribution $\mu_s$ and a target distribution $\mu_t$ both defined over $\mathcal{X}$. OT [Villani, 2008] aims at moving $\mu_s$ on the top of $\mu_t$ with a transportation plan of minimal cost. We will use in this paper the formulation proposed by Kantorovich [1942] which gives the Wasserstein distance between $\mu_s$ and $\mu_t$. Let $\Pi(\mu_s, \mu_t)$ be the collection of all joint probability measures on $\mathcal{X} \times \mathcal{X}$ with marginals $\mu_s$ and $\mu_t$ respectively. Let suppose that $\mu_s$ and $\mu_t$ have a finite $p$-order moment with $p \geq 1$. One can define the $p$-Wasserstein distance as

$$W_p^p(\mu_s, \mu_t) = \min_{\gamma \in \Pi(\mu_s, \mu_t)} \int_{\mathcal{X} \times \mathcal{X}} c(x_s, x_t)^p \, d\gamma(x_s, x_t),$$

where $c : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a cost function. The minimizer of the previous problem $\gamma^*$ corresponds to the Optimal Transport plan. In practice, we deal with the empirical measures $\hat{\mu}_s$ and $\hat{\mu}_t$ supported on $m_s$ and $m_t$ examples respectively. In such a case, $\gamma^*$ can be represented by a matrix of size $m_s \times m_t$ where $\gamma^*_{i,j}$ corresponds to the mass that should be sent from the point $x^*_s$ to the point $x^*_t$. Let $\Pi(\hat{\mu}_s, \hat{\mu}_t) = \{ \gamma \in \mathbb{R}^{m_s \times m_t} | \gamma 1_{m_s} = \hat{\mu}_s, \gamma^T 1_{m_t} = \hat{\mu}_t \}$. Considering the Euclidean distance as the cost function, we denote by $C^p$ the $m_s \times m_t$ matrix composed of the costs $C^p_{ij} = \|x^*_s - x^*_t\|^2_2$. The $p$-Wasserstein distance can be reformulated as follows:

$$W_p^p(\hat{\mu}_s, \hat{\mu}_t) = \min_{\gamma \in \Pi(\hat{\mu}_s, \hat{\mu}_t)} \sum_{i=1}^{m_s} \sum_{j=1}^{m_t} \|x^*_s - x^*_t\|^p_2 \gamma_{ij},$$

(2)

It is worth noting that (i) Problem (2) is a linear program, (ii) the solution is not stable and (iii) the algorithmic complexity is supercubic. To change it into a strongly convex problem with a unique solution, we usually add a regularization term in the form of the classic Shannon entropy [Cuturi, 2013]. Let $\lambda_c \in \mathbb{R}_+$ be a small regularization parameter, we get

$$W_p^p(\hat{\mu}_s, \hat{\mu}_t) \approx \min_{\gamma \in \Pi(\hat{\mu}_s, \hat{\mu}_t)} \left\{ \arg\min_{\gamma} \langle \gamma, C^p \rangle - \lambda_c \Omega_c(\gamma), C^p \right\},$$

(3)

with $\Omega_c(\gamma) = -\sum_{i=1}^{m_s} \sum_{j=1}^{m_t} \gamma_{ij} \log(\gamma_{ij})$. Problem (3) can be efficiently solved by resorting to the Sinkhorn-Knopp algorithm [Cuturi, 2013]. The mass of a source point will now be spread across several target points preventing the algorithm from overfitting and leading to smoother solutions.

OTDA (Optimal Transport for Domain Adaptation) [Courty et al., 2017b] was the first attempt to use Optimal Transport for Domain Adaptation. While Problem (3) is totally unsupervised, OTDA takes into account the labels of the transported points. Let us suppose that a discrete label $y_s \in \mathcal{Y} = [1, c]$ is associated to each source example, with...
c the number of classes. OTDA adds a penalty term to prevent two source points of different labels from being sent to the same target location. This takes the form of the following optimization problem:

$$\min_{\gamma \in \Omega(\hat{\mu}, \tilde{\mu}^*)} \langle \gamma, CP \rangle - \lambda_c \Omega_c(\gamma) + \lambda_r \Omega_r(\gamma),$$

where

$$\Omega_c(\gamma) = \sum_{j=1}^{m} \sum_{i=1}^{c} \| \gamma(J_{cl, j}) \|^2$$

and $\gamma(J_{cl, j})$ is the column $j$ of matrix $\gamma$ with only the rows corresponding to samples of class $cl$.

2.2 Metric Learning

As said before, the cost matrix $C$ used in OT is usually set to the Euclidean distance. Though this choice seems natural, it has a direct impact on the quality of the transportation. We suggest here to optimize this matrix by learning a metric that allows us to better match in a DA setting the source and target distributions, hopefully in a smaller feature space. Metric Learning (ML), and especially Mahalanobis distance learning, has been widely studied in the literature during the past decade (see, e.g., the survey [Bellet et al., 2015]). It typically boils down to optimizing the shape and the orientation of an ellipsoid rather than using the Euclidean ball. More formally, let $L \in \mathbb{R}^{k \times n}$ with $k \in [1, n]$ and $M$ be a PSD matrix such as $M = L^T L$. For all $(i, j) \in [1, \ldots, m]^2$, the squared Mahalanobis distance $D^2_M$ parameterized by $M$ is defined as

$$D^2_M(x^i, x^j) = (x^i - x^j)^T M (x^i - x^j) = ||L(x^i - x^j)||^2.$$  \hspace{1cm} (5)

Notice that $L$ defines a unique $M$ but there is more than one Cholesky decomposition of $M$. The goal of Metric Learning is to learn either the matrix $L$ or $M$ under semantic constraints, which typically aim to bring closer examples of the same class while pushing away data of different labels (see, e.g. LMNN [Weinberger and Saul, 2009] or ITML [Davis et al., 2007]). The problem is often convex in $M$ but the PSD constraint makes the optimization more complicated. The minimization in $L$ is not convex but is simpler and gives good results in practice. In the rest of this paper, we will denote by $\Omega_t(L)$ the underlying objective function of the metric learning problem. Note that learning a Mahalanobis distance for OT has been recently studied [Deshpande et al., 2019; Genevay et al., 2018; Paty and Cuturi, 2019; Cuturi and Avis, 2014]. Our objective in the rest of this paper is to show how to optimize such a ground metric when OT is used to address domain adaptation tasks by the alignment of the source and target subspaces.

3 Theoretical Analysis of DA with OT

In this section, we derive two theoretical results. First, we establish a strong relation between a PCA and the minimization of the Wasserstein distance. Then, we derive a generalization bound on the target error whose terms depend on several Wasserstein distances. By changing the Euclidean costs by a Mahalanobis distance, we can see $W^2_p(\hat{\mu}, \tilde{\mu})$ as a parameterized distance that might be optimized from training data. This leads to the design of a new Domain Adaptation algorithm, called MLAT (see Section 4), which learns the ground metric allowing us to optimize the transportation plan while performing the adaptation.

![Figure 2: Standard PCA projecting a discrete distribution \(\hat{\mu}\) (data in orange) onto a one-dimensional subspace (in purple). This projection is equivalent to finding the optimal mapping function \(g\) that minimizes the Wasserstein distance \(W_2(\hat{\mu}, g\#\hat{\mu})\).](image-url)
\[ \min_{g \in G_d} \mathcal{W}_2^2(\mu_g, g \# \tilde{\mu}) = \frac{1}{m} \sum_i \|x^i - \tilde{g}^*(x^i)\|_2^2 \]
\\[ \geq \frac{1}{m} \sum_i \|x^i - V^TVx^i\|_2^2. \]

The last line comes from the PCA. It shows that \( V^TV \) associated with \( \frac{1}{m} I_m \), upper bounds the optimal solution \( g^* \). But since \( V^TV \in G_d \), it is actually the optimal solution. \( \square \)

Theorem 1 tells us that PCA is the best way to reduce the dimension in the sense of the Wasserstein distance. The next result provides a generalization bound on the target error where Wasserstein distances are the main terms to minimize.

### 3.2 Upper Bound on the Target Risk

Let \( \mathcal{H} = \{ h : \mathcal{X} \to \mathcal{Y} \} \) be the hypothesis space. We assume the existence of a deterministic ground-truth function \( f : \mathcal{X} \to \mathcal{Y} \) which gives the label associated to each point of \( \mathcal{X} \). For all \( (h, h') \in \mathcal{H}^2 \), we define \( \epsilon_t(h, h') = P_{x \sim \mu_s}(h(x) \neq h'(x)) \) and \( \epsilon_s(h, h') = P_{x \sim \mu_s}(h(x) \neq h'(x)) \) which represent the disagreement between two classifiers.

The goal of DA is to find the best \( h \in \mathcal{H} \) which minimizes the target risk \( \epsilon_t(h, f) = P_{x \sim \mu_s}(h(x) \neq f(x)) \). In the same way, \( \epsilon_s(h, f) = P_{x \sim \mu_s}(h(x) \neq f(x)) \) is the source risk.

To simplify the notations, let us use \( \epsilon_t(h, f) = \epsilon_t(h) \) and \( \epsilon_s(h, f) = \epsilon_s(h) \).

**Lemma 1** (Generalization bound [Shen et al., 2017]). Let \( \mu_s, \mu_t \) be two probability measures on \( \mathcal{X} \). Assume the hypotheses \( h \in \mathcal{H} \) are all \( K \)-Lipschitz continuous for some \( K \in \mathbb{R}^+ \). Then, \( \forall (h, h') \in \mathcal{H}^2 \), the following holds:

\[ \epsilon_t(h, h') \leq \epsilon_s(h, h') + 2KW_2(\mu_t, \mu_s). \] (8)

Building on the proof of Redko et al. [2017] based on the seminal works of Ben-David et al. [2007] and Mansour et al. [2009], we derive the following bound.

**Theorem 2.** Let \( g_s : \mathcal{X} \to \mathcal{X} \) and \( g_t : \mathcal{X} \to \mathcal{X} \). Under the assumption of Lemma 1, \( \forall h \in \mathcal{H} \), the following holds:

\[ \epsilon_t(h) \leq \epsilon_s(h) + 2K\left[ W_2(g_s \# \mu_s, g_t \# \mu_t) \right] + 2K\left[ W_2(\mu_s, g_s \# \mu_s) + W_2(\mu_t, g_t \# \mu_t) \right] + \lambda \] (9)

where \( \lambda \) is the combined error of the ideal hypothesis \( h^* \) that minimizes the combined error \( \epsilon_s(h^*) + \epsilon_t(h^*) \).

**Proof.** We have:

\[ \epsilon_t(h) \leq \epsilon_t(h^*) + \epsilon_t(h^*), h^* \]
\[ = \epsilon_t(h^*) + \epsilon_s(h^*, h^*) + \epsilon_t(h^*, h) - \epsilon_s(h, h^*) \]
\[ \leq \epsilon_t(h^*) + \epsilon_s(h^*, h^*) + 2KW_2(\mu_s, \mu_t) \]
\[ \leq \epsilon_t(h^*) + \epsilon_t(h) + \epsilon_s(h^*) + 2KW_2(\mu_s, \mu_t) \]
\[ \leq \epsilon_t(h^*) + 2KW_2(\mu_s, \mu_t) + W_2(\mu_s, g_t \# \mu_s) + W_2(\mu_t, g_t \# \mu_t) \]
\[ \leq \epsilon_s(h) + \lambda \]
\[ + 2KW_2(\mu_s, \mu_t) + W_2(\mu_t, g_s \# \mu_s) + W_2(\mu_s, g_t \# \mu_t) \]
\[ \leq \epsilon_s(h) + 2KW_2(\mu_s, g_s \# \mu_s) + W_2(\mu_t, g_t \# \mu_t) + W_2(\mu_s, g_t \# \mu_t) \]
\[ \leq \epsilon_s(h) + 2KW_2(\mu_s, g_s \# \mu_s) + W_2(\mu_t, g_t \# \mu_t) + \lambda. \]

**Algorithm 1 MLOT**

**Input:** \( \eta \) (gradient step), \( X_s, X_t, Y_s \)

1. \( V_s = PCA(X_s), V_t = PCA(X_t) \)
2. \( L_s = V_s^TV_s, L_t = V_t^TV_t \)
3. for \( i = 1 \) to \( N \) do
   4. \( \gamma = \min_{\gamma \in \Omega(\mu_s, \mu_t)} (\gamma, C^2(L_s, L_t) - \lambda_s \Omega_s(\gamma) + \lambda_t \Omega_t(L_s)) \)
   5. \( L_s = L_s - \eta \nabla L_s(\gamma, C^2(L_s, L_t) + \lambda_t \Omega_t(L_s)) \)
4. end for
5. \( \hat{X}_s = \gamma L_t X_t \)
6. classifier = classifier method(\( \hat{X}_s, Y_s \))
7. return \( \hat{Y}_t \)

Note that if \( g_s \) and \( g_t \) are linear, \( g_s \# \mu_s \) and \( g_t \# \mu_t \) can be seen as linear projections of the source and target examples taking the form of two matrices that can be learned by any standard metric learning algorithm. Therefore, rather than discussing about its tightness, it is worth noticing that the previous bound is the first one which jointly relates (i) the target risk in domain adaptation, (ii) the minimization of the Wasserstein distance and (iii) the metrics that can be learned to get a better transportation plan. We can use Theorem 1 to minimize both \( W_2(\mu_s, g_s \# \mu_s) \) and \( W_2(\mu_t, g_t \# \mu_t) \). Note that \( W_2(\mu_s, \mu_t) \) and \( W_2(\mu_s, \mu_t) \) can be bounded under some assumptions using Theorem 2.1 in [Bolley et al., 2007]. Moreover, \( \lambda \) is supposed to be small to allow the adaptation. A theoretical analysis about \( \lambda \) is available in [Redko et al., 2019a].

The last term of interest that has to be minimized in the bound of Theorem 2 is \( W_2(g_s \# \mu_s, g_t \# \mu_t) \). We address this problem from an algorithmic perspective thanks to our algorithm MLOT presented in the next Section.

### 4 MLOT: Metric Learning in OT for DA

Inspired from OTDA [Courty et al., 2017b], our algorithm MLOT leverages our previous theoretical analysis and resorts to an additional term \( \Omega_t(L_s) \) dedicated to optimize a metric allowing us to get a better transportation plan. Let us consider the cost function \( C^2(L_s, L_t)_{ij} = \|L_s x^i - L_t x^j\|_2^2 \). MLOT takes the form of the following joint optimization problem:

\[ \min_{L_s \in \mathbb{R}^{n \times n}, \gamma \in \Omega(\mu_s, \mu_t)} (\gamma, C^2(L_s, L_t) - \lambda_s \Omega_s(\gamma) + \lambda_t \Omega_t(L_s) + \lambda_t \Omega_t(L_s). \]

Note that MLOT only learns the matrix \( L_s \) associated to the source data. The reason is twofold. First, it prevents the algorithm from leading to a trivial minimal solution \( W_2(L_s \# \mu_s, L_t \# \mu_t) \) where both matrices \( L_s \) and \( L_t \) are null. By this way, MLOT tends to provide two different matrices \( L_s \) which is (learned) and \( L_t \) (set to \( V_t^TV_t \) according to Theorem 1) which better capture the peculiarities of the two distributions. Second, labels - that are required to learn a metric - are available only in the source domain. To find the solution of Problem (10), we minimize the objective function w.r.t. \( \gamma \) and \( L_s \) alternately. From a practical point of view, we apply one step of gradient descent over \( L_s \) and then completely
compute the optimal \( \gamma \). Since the problem is not convex, the initialization of \( L_s \) is key. According to Theorem 1, we use a PCA to set \( L_s = V^T_s V_s \). At the initialization, \( L_s \) is a \( n \times n \) matrix of rank \( d \). But note that along the iterations, this rank can increase if it allows a better adaptation. The pseudo-code of MLOT is described in Algo. 1, where \( X_s \) and \( X_t \) are the source and target datasets, and \( Y_s \) is the set of source labels.

Note that any gradient descent-based metric learning algorithm can be used to learn \( L_s \) via the term \( \lambda l_{1}(L_s) \). Note also that the computation of the barycenter of the transported points \( \bar{X}_s = \gamma L_t X_t \) makes sense only if the 2-Wasserstein is used. Figure 3 summarizes the workflow of MLOT.

5 Experiments

In this section, we perform experiments and demonstrate the effectiveness of MLOT compared to OTDA and other baselines on various datasets and types of features.

5.1 Datasets

We use the Office-Caltech dataset [Gong et al., 2012] which is a classic benchmark on visual DA. We study the effect of using different features such as SURF features [Bay et al., 2006] and DeCAF Deep Learning features [Donahue et al., 2014]. The Office-Caltech dataset is composed of 4 different subsets (Amazon, Caltech, DSLR, Webcam) that are combined in a pairwise manner, to create 12 DA subproblems. The notation \( A \rightarrow C \) means that Amazon is used as the source and Caltech as the target. There are the same 10 classes in each dataset containing from 157 to 1,123 images.

We also use the Office31 dataset [Saenko et al., 2010] with features extracted from the 7th layer of DeCAF Deep Learning network. The dataset is composed of 4,110 images in 3 subsets (Amazon, DSLR, Webcam) with 31 classes.

5.2 Setup and Cross-validation

We compare 10 different methods\(^1\) that are able to handle arbitrary features. We exclude deep learning methods as they require having the original images and fine tuning a network. We compare: NA (No Adaptation). The classifier is learned on the source dataset and directly applied on the target. LMNN: Large Margin Nearest Neighbor [Weinberger and Saul, 2009]. SA: Subspace Alignment [Fernando et al., 2013]. CORAL: CORrelation ALignment [Sun et al., 2016]. TCA: Transfer Component Analysis [Pan et al., 2011]. OT: Optimal Transport with entropy [Cuturi, 2013]. OTDA: Optimal Transport with entropy and class regularization [Courty et al., 2017b]. OTDA\(_p\): OTDA after a PCA. JDOT: Joint Distribution Optimal Transportation [Courty et al., 2017a]. MLOT: our method. Following [Courty et al., 2017b] the final classification is done with a 1-Nearest Neighbor (1NN).

In unsupervised DA, there is no target label and it is impossible to use the classical cross-validation procedure to choose the best hyper-parameters. To fairly compare methods, we take inspiration from the work of [Zhong et al., 2010] and apply the following strategy for all methods. We first assign pseudo-labels to the target points (using the considered method) and then use these target labels to re-assign labels to the source data, using a basis DA algorithm. Here, we choose SA [Fernando et al., 2013] which has been shown to be one of the most robust DA method. We can then compare the actual source labels with the predicted source ones. We take the set of hyper-parameters that gives the best accuracy over 48 hours, limited to 1000 iterations. This back-and-forth adaptation is done independently for each pair of datasets. MLOT is parameterized by 5 hyper-parameters: the three regularization parameters \( (\lambda_s, \lambda_c, \lambda_t) \) which control the trade-off between each term in Eq. (10), the number of dimensions kept by the PCA \( d \) and the number of iterations \( N \). Note that in these experiments, we used arbitrarily LMNN [Weinberger and Saul, 2009] to learn \( L_s \) in the term \( l_{1}(L_s) \). Therefore, an additional parameter has to be tuned corresponding to the margin used in this metric learning algorithm. Note that SA and MLOT resort to a PCA. To speed-up the process, we used a "randomized"-PCA [Halko et al., 2011] and run 10 iterations. This explains why the variance is indicated for these three methods in the reported results in Table 1.

The code of the 10 methods is available\(^2\), together with the datasets, the code for the cross-validation that recreates Table 1, and the code that produces automatically Figures 1 and 4.

5.3 Analysis of the Results

The results are reported in Table 1. For the SURF features, MLOT outperforms, on average, all the other methods. MLOT outperforms OTDA 8 times over the 12 DA subproblems and yields impressive improvements for some cases (e.g. \( W \rightarrow A \) and \( C \rightarrow D \)) and a clear gain on average (1.5 points). The results on Office-Caltech DeCAF6 features show the effectiveness of our method on deep learning features. MLOT outperforms by 1.8 the second best method (here OT). On the Office31 dataset, the best results are obtained by SA. MLOT is still very competitive and outperforms OTDA by 0.9 point on average.

As already mentioned, cross-validating the hyperparameters in unsupervised DA is key and is still an open problem, since we do not have access to labels from the target domain. We performed an experimental comparison to show how the cross-validation method used in this paper behaves when compared to a scenario where we would use the actual labels of the target examples. Table 2 reports the gap between the optimal hyperparameters and those obtained by our method inspired from [Zhong et al., 2010]. We can see

\(^1\)We could not run MEDA [Wang et al., 2018] because the implementation requires the use of a proprietary software.

\(^2\)https://github.com/Hv0nnus/MLOT
Table 1: Accuracy of all the methods on 3 different types of features. The best method for each dataset is in bold.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NA</th>
<th>LMNN</th>
<th>SA</th>
<th>CORAL</th>
<th>TCA</th>
<th>OT</th>
<th>OTDA</th>
<th>OTDA_p</th>
<th>JDOT</th>
<th>MLOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SURF</td>
<td>A→C</td>
<td>26.0</td>
<td>40.3</td>
<td>40.2±0.2</td>
<td>25.4</td>
<td>40.0</td>
<td>33.9</td>
<td>40.2</td>
<td>39.4±0.5</td>
<td>39.9</td>
</tr>
<tr>
<td></td>
<td>A→D</td>
<td>25.5</td>
<td>36.9</td>
<td>39.5±0.5</td>
<td>26.8</td>
<td>31.8</td>
<td>30.6</td>
<td>40.1</td>
<td>39.6±1.1</td>
<td>37.6</td>
</tr>
<tr>
<td></td>
<td>A→W</td>
<td>29.8</td>
<td>38.0</td>
<td>39.9±1.3</td>
<td>26.8</td>
<td>41.7</td>
<td>32.5</td>
<td>37.3</td>
<td>39.8±0.9</td>
<td>38.0</td>
</tr>
<tr>
<td>C→A</td>
<td>23.7</td>
<td>46.0</td>
<td>41.3±1.1</td>
<td>23.6</td>
<td>39.8</td>
<td>41.0</td>
<td>41.0</td>
<td>45.27</td>
<td>48.5±0.7</td>
<td>48.1</td>
</tr>
<tr>
<td>C→D</td>
<td>25.5</td>
<td>45.9</td>
<td>45.4±1.2</td>
<td>26.1</td>
<td>44.6</td>
<td>36.9</td>
<td>47.9</td>
<td>51.4±1.4</td>
<td>49.7</td>
<td><strong>52.2±1.3</strong></td>
</tr>
<tr>
<td>C→W</td>
<td>25.8</td>
<td>41.7</td>
<td>36.6±1.1</td>
<td>23.7</td>
<td>36.9</td>
<td>28.1</td>
<td><strong>46.4</strong></td>
<td>45.8±1.4</td>
<td>43.4</td>
<td>45.9±0.8</td>
</tr>
<tr>
<td>D→A</td>
<td>28.5</td>
<td>31.1</td>
<td>35.4±1.0</td>
<td>28.8</td>
<td>32.9</td>
<td>29.3</td>
<td>32.4</td>
<td>37.8±1.0</td>
<td>32.8</td>
<td><strong>37.8±0.7</strong></td>
</tr>
<tr>
<td>D→C</td>
<td>26.3</td>
<td>30.7</td>
<td>32.3±0.6</td>
<td>30.0</td>
<td>31.5</td>
<td>31.7</td>
<td>32.0</td>
<td>33.5±0.7</td>
<td>31.7</td>
<td><strong>34.4±0.5</strong></td>
</tr>
<tr>
<td>D→W</td>
<td>63.4</td>
<td>77.3</td>
<td>88.5±1.1</td>
<td>84.4</td>
<td>84.7</td>
<td><strong>88.8</strong></td>
<td><strong>88.8</strong></td>
<td>87.5±1.2</td>
<td>82.7</td>
<td>87.8±0.7</td>
</tr>
<tr>
<td>W→A</td>
<td>23.0</td>
<td>32.3</td>
<td>32.6±0.5</td>
<td>26.2</td>
<td>29.4</td>
<td>34.1</td>
<td>33.7</td>
<td>37.6±0.6</td>
<td>37.6</td>
<td><strong>38.0±0.8</strong></td>
</tr>
<tr>
<td>W→C</td>
<td>19.9</td>
<td>30.4</td>
<td>29.0±0.6</td>
<td>22.6</td>
<td>29.2</td>
<td>30.1</td>
<td><strong>34.1</strong></td>
<td>33.3±0.5</td>
<td>33.1</td>
<td>33.2±0.6</td>
</tr>
<tr>
<td>W→D</td>
<td>59.2</td>
<td>86.6</td>
<td>89.5±1.0</td>
<td>84.1</td>
<td>91.7</td>
<td>89.2</td>
<td><strong>92.4</strong></td>
<td>91.8±1.2</td>
<td>89.8</td>
<td>90.8±0.8</td>
</tr>
<tr>
<td>AVG</td>
<td>31.4</td>
<td>44.8</td>
<td>45.8±1.0</td>
<td>35.7</td>
<td>44.5</td>
<td>42.2</td>
<td>48.2</td>
<td>48.8±0.9</td>
<td>47.0</td>
<td><strong>49.7±0.8</strong></td>
</tr>
</tbody>
</table>

Table 2: Accuracy comparison on Office-Caltech (SURF features) between a cross-validation method that uses the true target labels (first line) and the cross-validation method used in this paper that exploits pseudo-labels in the unsupervised DA setting (second line). CORAL and NA are excluded as they do not have hyperparameter.

<table>
<thead>
<tr>
<th>OT</th>
<th>TCA</th>
<th>LMNN</th>
<th>SA</th>
<th>JDOT</th>
<th>OTDA</th>
<th>OTDA_p</th>
<th>MLOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG</td>
<td>71.0</td>
<td>79.4</td>
<td>83.7±0.5</td>
<td>77.2</td>
<td>83.4</td>
<td>83.9</td>
<td>83.2</td>
</tr>
<tr>
<td>AVG</td>
<td>64.3</td>
<td>64.7</td>
<td><strong>66.5±0.2</strong></td>
<td>64.1</td>
<td>64.1</td>
<td>65.3</td>
<td>65.3</td>
</tr>
<tr>
<td>AVG</td>
<td>53.8</td>
<td>62.6</td>
<td>65.1±0.6</td>
<td>58.0</td>
<td>64.0</td>
<td>63.5</td>
<td>63.6</td>
</tr>
</tbody>
</table>

Figure 4: Absolute difference between the mean accuracy of MLOT and OTDA (on the left). The rows (resp. columns) correspond to the results with different values of the regularization parameter $\lambda_e$ (resp. $\lambda_c$) on the entire Office-Caltech dataset with SURF features. On the right, accuracy of MLOT for each pair of parameters.

that the ranking of the methods is preserved, even though this experiment shows that there is still room for improving the way we may tune the parameters in unsupervised DA. Note that the other datasets show similar behaviors. To show the specific gain brought by MLOT compared to OTDA, we performed a last experiment where we set the hyper-parameters of MLOT as follows: $\lambda_1 = 1$, $N = 10$, $margin_{LMNN}$ = 10, $d =$ 70; and we tune $\lambda_e$ and $\lambda_c$. The results are reported in Figure 4. It is worth noticing that whatever the set of parameters, MLOT always yields better accuracy, which confirms the interest of learning a metric and using the PCA to initialize $L_s$ and $L_t$. Note that when the entropy term has more importance (last column), the difference between the two methods is smaller because $\gamma$ tends to be uniform. When the class regularization is set to 0 (first row), OTDA becomes similar to OT (Sinkhorn algorithm). This shows the effectiveness of MLOT even without the class regularization. However, the performances drop without this supervised information which tends to show that the metric learned and the class regularization are complementary. Notice that the results of MLOT are quite good for many values of the entropy and class regularizations (Figure 4 on the right). The best performance is 50.9 which is better than the result obtained by the cross-validation method. Once again, this is an evidence about the difficulty of tuning parameters in unsupervised DA.

6 Conclusion

We proposed in this paper a new Domain Adaptation (DA) method, called MLOT, benefiting from both Metric Learning (ML) and Optimal Transport (OT). Dedicated to address problems in the complex unsupervised DA setting, MLOT jointly learns a good metric and the optimal transportation plan. A theoretical study has driven the design of MLOT. We derived a bound on the target error which prompts us to learn the ground metric involved in the Wasserstein distance. The experimental study has shown very competitive results and a significant improvement compared to OTDA, the first method coupling OT and DA. Although deep learning is not at the core of this paper, note that we designed a differentiable version of MLOT using the PyTorch framework. While the first results, with pre-extracted features, did not bring any performance boost, this implementations opens the door to a full end-to-end model for Wasserstein-based domain adaptation.

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


