Target to Source Coordinate-wise Adaptation of Pre-trained Models

Luxin Zhang, Pascal Germain, Yacine Kessaci, Christophe Biernacki

Target to Source Coordinate-wise Adaptation of Pre-trained Models

Luxin Zhang[1,2]✉, Pascal Germain[2,3], Yacine Kessaci[1], and Christophe Biernacki[2]

[1] Worldline, France
{luxin.zhang, yacine.kessaci}@worldline.com
[2] MODAL Team, Inria, Lille, France
{luxin.zhang, germain.pascal, christophe.biernacki}@inria.fr
[3] Université Laval, Québec, Canada
pascal.germain@ift.ulaval.ca

Abstract. Domain adaptation aims to alleviate the gap between source and target data drawn from different distributions. Most of the related works seek either for a latent space where source and target data share the same distribution, or for a transformation of the source distribution to match the target one. In this paper, we introduce an original scenario where the former trained source model is directly reused on target data, requiring only finding a transformation from the target domain to the source domain. As a first approach to tackle this problem, we propose a greedy coordinate-wise transformation leveraging on optimal transport. Beyond being fully independent of the model initially learned on the source data, the achieved transformation has the following three assets: scalability, interpretability and feature-type free (continuous and/or categorical). Our procedure is numerically evaluated on various real datasets, including domain adaptation benchmarks and also a challenging fraud detection dataset with very imbalanced classes. Interestingly, we observe that transforming a small subset of the target features leads to accuracies competitive with "classical" domain adaptation methods.

Keywords: Domain Adaptation · Optimal Transport · Feature Selection.

1 Introduction

Traditional supervised machine learning algorithms assume the estimated model to be used on the data having the same underlying distribution as the training one. However, this assumption is not always valid. For example, a predictive model may have been trained on a dataset of users living in a specific country, and be used afterward on a dataset of users living in another geographical region. A common domain adaptation strategy used to mitigate these differences is to align the training (source) data and the test (target) data. However, most of these methods rely on training a predictor on the transformed source dataset.

Related Works. To mitigate the gap between source and target domains, former "classical" domain adaptation methods seek to minimize a discrepancy term between the source and target distributions. The minimization of Kullback-Leibler Divergence [14] and Maximum Mean Discrepancy [11] are among the most widely used techniques. The former re-weights source samples in the target domain, while the latter aligns source and target distributions in a latent space. Besides, a recently proposed correlation alignment method [15] aims to align the source domain and target domain second-order statistics. Nevertheless, one still needs to re-estimate a model on the transformed source data.

Deep learning approaches have also been proposed to tackle the domain adaptation problem. Methods like Deep Adaptation Networks [9] and Domain Adversarial Neural Networks [6] have achieved state-of-the-art results in computer vision tasks. However, it is known that these methods are not interpretable and requires careful tuning of hyperparameters.

Finally, prior works that leverage on optimal transport [4,3] focus mainly on the source to target transformation and are not scalable to a huge dataset. Hence, to the best of our knowledge, we are the first to tackle the target to source adaptation problem that is scalable and requires no retraining.

Contributions. In this paper, we propose a new domain adaptation perspective by transforming the target data into the source one, and applying directly the pre-trained model on the adapted data. We argue that this strategy is more suited to some real-life scenarios. Indeed, in an industrial context, a prediction system might have been developed and trained by employees or contractors that are no more available. This system being used as a "black box" predictor, it cannot be retrained on a new dataset. However, it is often mandatory to adapt such a system to different contexts, given that the industry wants to expand their business, or just because the data distribution naturally "drift" as time goes.

Therefore, we introduce an original scenario where the former trained model is directly reused on target data, requiring only finding a transformation from the target domain to the source domain. As a first approach to tackle this problem, we propose a greedy coordinate-wise transformation leveraging on optimal transport. Beyond being fully independent of the model initially learned on the source data, we design our approach with the following three characteristics in mind, as we want our method to be appealing for users in an industrial context.

1. Scalability: We want our transformation to be computable even on very large datasets.
2. Interpretability: We want to provide the user some information that can help him to interpret the nature of the target to source transformation.
3. Feature-type free: We want our transformation to apply to a variety of data types. In this paper, we address specifically the case where the data is a mix of numerical and categorical attributes.

Our domain adaptation method is applicable without any label, but we show that one can use few target labels to select the features to adapt. We show empirically that this feature selection scheme leads to competitive results with state-of-the-art domain adaptation methods that necessitate a training phase. Our experiments are performed on three real-life datasets.

2 Domain Adaptation and Optimal Transport

2.1 The Studied Domain Adaptation Framework

Let denote $(x, y) \in \mathcal{X} \times \mathcal{Y}$ a data point, where $\mathcal{X}$ and $\mathcal{Y}$ refer respectively to the input space and the output space. In this paper, $\mathcal{X}$ encompasses vectors of categorical and numerical types, possibly mixed together. Moreover, we focus on classification problems, with binary labels $\mathcal{Y} = \{0, 1\}$, but our method naturally extends to multilabel classification. In the supervised learning framework, one observes a training dataset $\{(x_i, y_i)\}_{i=1}^m$, and each observed training sample (x_i, y_i) is viewed as a realization of a random variable pair (X, Y) obeying a joint probability $P(X, Y)$. Thus, to learn a classifier, a supervised learning algorithm aims to model $P(Y|X)$ from the training dataset.

The domain adaptation framework differs from the standard supervised learning one by the fact that two distinct joint probabilities over the input-output space $\mathcal{X}\times\mathcal{Y}$ are considered, referred to as the *source domain* probability $P(X^s, Y^s)$ and the *target domain* probability $P(X^t, Y^t)$. The domain adaptation classification problem is to infer the target predictive model $P(Y^t|X^t)$ in the situation where most observed learning data are instances of (X^s, Y^s). More precisely, we stand hereafter in the semi-supervised domain adaptation setting where no target label is provided to the learner. We denote the source training points as $\mathbb{X}^s = \{x_1^s, \ldots, x_i^s, \ldots, x_m^s\} \in \mathcal{X}^m$, $\mathbb{Y}^s = \{y_1^s, \ldots, y_i^s, \ldots, y_m^s\} \in \mathcal{Y}^m$, and the target training points as $\mathbb{X}^t = \{x_1^t, \ldots, x_j^t, \ldots, x_n^t\} \in \mathcal{X}^n$. Of course, the challenging task of learning $P(Y^t|X^t)$ without target labels[4] is only achievable under the assumption that, despite $P(X^s, Y^s) \neq P(X^t, Y^t)$, both joint probabilities are "similar" to each other. Any rigorous domain adaptation study must characterize the underlying source-target similarity assumptions. In the domain literature, one common assumption is the so-called *covariate shift* setting [13], which describes the case where the source domain and target domain output conditional distributions coincide, that is $P(Y^s|X^s) = P(Y^t|X^t)$ whereas the input marginal distributions $P(X^s)$ and $P(X^t)$ differ. Alternatively, Courty et al. [4] assume that there exists a *mapping function* $\mathcal{T} : \mathcal{X} \to \mathcal{X}$ that models the domain drift from the source to the target, such that $P(Y^s|\mathcal{T}(X^s)) = P(Y^t|X^t)$. Our work builds on a sibling assumption, that is the existence of a mapping function $\mathcal{G} : \mathcal{X} \to \mathcal{X}$ that models the *domain drift* from the target to the source:

$$P(Y^s|X^s) = P(Y^t|\mathcal{G}(X^t))\,.$$

One could also look for a bijective mapping function such that $\mathcal{G}^{-1} = \mathcal{T}$, but this is not required by our analysis. Inspired by Courty et al. [4], we choose to leverage on *optimal transport* methods to empirically estimate $\mathcal{G}$ from the training

[4] In the experiment section, we consider few target labels in order to perform model selection.

dataset. Nonetheless, the source to target method of Courty et al. [4] requires computing the transportation map from $\mathbb{X}^s$ to $\mathbb{X}^t$, and training a learning model on the training dataset $\{(\mathcal{T}(x_i^s), y_i^s)\}_{i=1}^m$ afterward, while our method is based on the idea of using a pre-trained source model.

2.2 Optimal Transport for Domain Adaptation

The optimal transport problem was first introduced by Monge in the 18th century [10] and further developed by Kantorovich in the mid-20th [8]. Intuitively, the original Monge-Kantorovich problem looks for minimal effort to move masses of dirt to fill a given collection of pits. Optimal transport has been revisited over the past years to solve a variety of computational problems [12], including many machine learning ones [5]. It is naturally suited for domain adaptation problems [4], and it offers a principled method to transform a source distribution into a target one.

Let us now present the important optimal transport notions we rely upon. Even if we tailor the nomenclature to the context of our domain adaptation problem (*e.g.* we invoke "target" and "source" densities), the remaining part of this section is shared by the general optimal transport literature.

The training sets $\mathbb{X}^s$ and $\mathbb{X}^t$ provide discrete estimations of the input domain densities $P(X^s)$ and $P(X^t)$. Classically, we consider the empirical distributions as additions of Dirac functions. Denoting by δ_x the Dirac measure on $x \in \mathcal{X}$, we define the empirical estimation of the source domain distribution and the target domain distribution as

$$\hat{P}(X^s) = \sum_{i=1}^{m} w_i^s \delta_{x_i^s}\,, \text{ and } \hat{P}(X^t) = \sum_{j=1}^{n} w_j^t \delta_{x_j^t}\,,$$

where $\mathbb{W}^s = \{w_1^s, ..., w_i^s, ..., w_m^s\}$ and $\mathbb{W}^t = \{w_1^t, ..., w_j^t, ..., w_n^t\}$ are weights over the training points. Typically, we consider that the mass is uniformly distributed among each point, *i.e.* $w_i^s = \frac{1}{m}$ and $w_j^t = \frac{1}{n}$, but the framework allows reweighing the samples, such that

$$\sum_{i=1}^{m} w_i^s = \sum_{j=1}^{n} w_j^t = 1\,; \quad w_i^s, w_j^t \geq 0\,.$$

For simplicity, we assume that every x_i^s appears only once in $\mathbb{X}^s$, respectively x_j^t in $\mathbb{X}^t$. We then write $\hat{P}(X^s{=}x_i^s) = w_i^s$ and $\hat{P}(X^t{=}x_j^t) = w_j^t$.

Central to optimal transport methods is the notion of a *cost function* between a source point and a target point, denoted by

$$c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}\,. \tag{1}$$

Moreover, $C \in R^{m\times n}$ denotes the *cost matrix* between source and target training points such that $C_{i,j} = c(x_i^s, x_j^t)$ corresponds to the cost of moving weight from $x_j^t \in \mathbb{X}^t$ to $x_i^s \in \mathbb{X}^s$. Based on these concepts, we present below the Kantorovich [8] formulation of the optimal transport problem in the discrete case.

Definition 1 (Kantorovich's discrete optimal transport problem). *The relationship between source and target examples is encoded as a joint probability* coupling matrix $\gamma \in \mathbb{R}_+^{m\times n}$*, where* $\gamma_{i,j}$ *corresponds to the* weight *to be* moved *from* $x_j^t \in \mathbb{X}^t$ *to* $x_i^s \in \mathbb{X}^s$*. The set of admissible coupling matrices is given by*

$$\varGamma = \left\{ \gamma \in \mathbb{R}_+^{m\times n} \;\middle|\; w_{i'}^s = \sum_{j=1}^{n} \gamma_{i',j} \;\; \text{and} \;\; w_{j'}^t = \sum_{i=1}^{m} \gamma_{i,j'} \right\}.$$

Then, the optimal coupling matrix γ^* *is obtained by solving*

$$\gamma^* = \underset{\gamma\in\varGamma}{\operatorname{argmin}} \, \langle C, \gamma \rangle = \underset{\gamma\in\varGamma}{\operatorname{argmin}} \sum_{i=1}^{m}\sum_{j=1}^{n} C_{i,j}\gamma_{i,j} \,. \tag{2}$$

In turns the transformation function $\mathcal{G}$ *is given by*

$$\mathcal{G}(x_j^t) = \underset{x\in\mathcal{X}}{\operatorname{argmin}} \sum_{i=1}^{m} \gamma_{i,j}^* c(x, x_j^t) \,. \tag{3}$$

The solution $x \in \mathcal{X}$ *of Equation* (3) *minimization problem is commonly referred to as the* barycenter *in the optimal transport literature.*

Equation (2) is a linear optimization problem, and algorithms such as the network simplex or dual ascent methods can be used to compute the solution [12]. The obtained joint probability γ^* reveals the allocation of mass from one domain to the other.

However, the computational complexity of this linear optimization problem is expensive; In the case where the number of target or source training points are equal ($m = n$), the computation time is $O(n^3)$. By adding a regularization term $H(\gamma) = \sum_{i,j} \gamma_{i,j} \log(\gamma_{i,j})$ to Equation (2), Cuturi [5] achieve to reduce the computation complexity. The regularized optimization problem is expressed as

$$\gamma_\eta^* = \underset{\gamma\in\varGamma}{\operatorname{argmin}} \, \langle C, \gamma \rangle + \eta H(\gamma) \,, \tag{4}$$

where $\eta > 0$ is a hyperparameter to be fixed. Altschuler et al. [1] show that, using the Sinkhorn iteration method [5], solving the regularized optimal transport of Equation (4) requires about $O(n^2 \log(n))$ operations. This is still too expensive to apply to large learning problems.[5] Also, as mentioned before, such domain adaptation methods imply training a new model from the transformed source dataset.

To overcome these limitations of existing domain adaptation methods, we propose in the following section a coordinate-wise target to source domain adaptation method.

[5] The number of transactions in fraud detection datasets as the ones used in the experiments of Section 4 is around ten million.

3 Target to Source Coordinate-wise Domain Adaptation

In this section, we formalize our target to source domain adaptation problem and propose our adaptation method.

3.1 Formalization of Target to Source Transformation

The originality of our domain adaptation approach is to rely solely on a target to source transformation. That is, we consider that we have access to a source training dataset $\mathbb{X}^s$, a target dataset $\mathbb{X}^t$, and a pre-trained source predictor

$$h^s : \mathcal{X} \to \mathcal{Y} . \tag{5}$$

This predictor h^s might have been trained on the available source training dataset $\mathbb{X}^s$, but it could also originate from another dataset that is no more available. Also, no assumption is made on the nature of the predictor. For instance, it could be a neural network, a support vector machine, a decision tree, *etc.* Hence, we consider h^s as a "black box" predictor, and the goal is to predict the label of the samples in $\mathbb{X}^t$.

We consider that h^s has been trained to minimize a loss function $l\colon \mathcal{Y}\times\mathcal{Y}\to\mathbb{R}_+$. The loss on the source distribution is $R_s^l(h^s) = \mathbf{E}\, l\big(h^s(X^s), Y^s\big)$. In the absence of labeled data, we cannot assess the quality of this estimation. Nonetheless, we want to be able to provide a similar performance on the target distribution, according to the same metric. That is, we would like to find a target to source mapping $\mathcal{G} : \mathcal{X} \to \mathcal{X}$ such that h^s minimizes the loss on the transformed samples from the target distribution: $R_t^l(h^s \circ \mathcal{G}) = \mathbf{E}\, l\big(h^s(\mathcal{G}(X^t)), Y^t\big)$. The following proposition states sufficient conditions (Equations 6 and 7) for which our goal (Equation 8) is achieved.

Proposition 1. *Under the assumption that*

$$P(Y^s) \;=\; P(Y^t) \,, \tag{6}$$

if we can find a transformation $\mathcal{G} : \mathcal{X} \to \mathcal{X}$ *such that*

$$\forall x \in \mathcal{X}\,; \forall y \in \mathcal{Y} : \quad P(X^s = x | Y^s = y) = P(X^t = \mathcal{G}(x) | Y^t = y) \,, \tag{7}$$

then for any $h^s : \mathcal{X} \to \mathcal{Y}$*, we have*

$$R_t^l(h^s \circ \mathcal{G}) = R_s^l(h^s) \,. \tag{8}$$

Proof. The proof is straightforward by noticing that Equations (6) and (7) imply

$$\forall x \in \mathcal{X}\,; \forall y \in \mathcal{Y} : \quad P(X^s = x, Y^s = y) = P(X^t = \mathcal{G}(x), Y^t = y) \,.$$

Since the $\mathcal{G}$-mapped target joint probability equals the source joint probability, $P(\mathcal{G}(X^t), Y^t) = P(X^s, Y^s)$, Equation (8) is obtained by a change of variable. □

Albeit simple, this training-free target to source perspective on domain adaptation is—up to our knowledge—an unexplored problem. The landscape of possible methods to address this problem is certainly vast. In the remaining part of the paper, we propose and evaluate a variant of optimal transport methods.

3.2 Marginal Coordinate-wise Optimal Transport

The target to source transformation explored in the current paper assumes that the output marginal distributions are equivalent, as stated by Equation (6) of Proposition 1, and is motivated by the goal to find a mapping $\mathcal{G}$ complying Equation (7). However, due to the lack of labeled data, it is not possible to directly estimate $\mathcal{G}$; we cannot properly estimate the conditional of the inputs (X^s or X^t) given the outputs (Y^s or Y^t). In these conditions, we relax the requirement of Equation (7), and we seek for a function $\mathcal{G}$ belonging to the family of transformations that aligns the input marginal distributions:

$$P(X^s = x) = P(X^t = \mathcal{G}(x))\,. \tag{9}$$

Inspired by the previous works on optimal transport for domain adaptation introduced in Section 2.2, we choose $\mathcal{G}$ to minimize a transportation cost. Although it would be possible to define a cost function (see Equation 1) between every $x_i^s \in \mathbb{X}^s$ and $x_j^t \in \mathbb{X}^t$ in order to find the Kantorovich discrete optimal transport problem (Definition 1), it would cause some issues that we aim to overcome:

- Solving Kantorovich's optimization problem (Equation 2), or even its regularized version (Equation 4), is computationally expensive on large datasets;
- In the case where the input space $\mathcal{X}$ contains mixed attributes, such a mix of numerical and categorical values, defining a cost function might be difficult;
- Even in the case where the input space contains exclusively numerical attributes (*e.g.* $\mathcal{X} \subseteq \mathbb{R}^d$), multidimensional distance metrics like Euclidean distance is not able to deal properly with the different scaling of each coordinate.
- As performing multidimensional optimal transport addresses the dependence across attributes, the solution is a large variance estimator of the optimal transformation in the case where the available data is not sufficiently abundant.

The proposed domain adaptation method is then performed by solving a sequence of one-dimensional optimal transport method. Doing so, we decompose the transformation $\mathcal{G}$ by feature-wise transformations $\mathcal{G}_k$:

$$\mathcal{G} = [\mathcal{G}_1, ..., \mathcal{G}_k, ..., \mathcal{G}_d]\,, \tag{10}$$

where d is less or equal to the number of features of the input space $\mathcal{X}$.

Each elementary transformation $\mathcal{G}_k$ solves the Kantorovich optimization problem (Equation 2) on one feature only, thus the total computation is generally less expensive compared than the relaxed optimal transport problem (Equation 4).[6] The distance measure can also be easily defined for each specific feature, especially when each of them has a different significance. Note that this feature by feature transformation is also robust to variation of scaling.

[6] See Sections 3.3 and 3.4 for details.

We denote by d' the number of transformations of numerical features. Without loss of generality, we refer the one-dimensional transformations of numerical features as $\mathcal{G}_1 ... \mathcal{G}_{d'}$, and to the transformations of categorical features as $\mathcal{G}_{d'+1} ... \mathcal{G}_d$. The next two subsections detail how we process the numerical and the categorical features.

3.3 One-Dimensional Mapping of Numerical Features

The 1-D optimal transport on the real line has a closed-form solution [12] provided that the cost of moving one point to another is defined with respect to an ℓ^p norm:

$$\forall x^s, x^t \in \mathbb{R}\,,\ c^p_{num}(x^s, x^t) = |x^s - x^t|^p\,.$$

Different from the resolution of multidimensional optimal transport, in our 1-D scenario, we first need to sort the numerical features values in ascending order. We denote the obtained real-valued vector as

$$\forall k \in \{1, \ldots, d'\}\,,\ \mathbb{X}^s_k = \left(x^s_{k,1}, \ldots, x^s_{k,i}, \ldots, x^s_{k,m}\right) \in \mathbb{R}^m\,,$$
$$\forall k \in \{1, \ldots, d'\}\,,\ \mathbb{X}^t_k = \left(x^t_{k,1}, \ldots, x^t_{k,j}, \ldots, x^t_{k,n}\right) \in \mathbb{R}^n\,,$$

where $x^s_{k,i}$ is the k-th feature of a training point in $\mathbb{X}^s$, ranked at position i according to the sorted order: $x^s_{k,i} \leq x^s_{k,i+1}$ for all $i \in \{1, \ldots, m-1\}$. Then, the transformation function $\mathcal{G}_k$ is obtained by the following formula:

$$\mathcal{G}_k(x^t_{k,j}) = (F_s^{-1} \circ F_t)(x^t_{k,j})\,, \tag{11}$$

where $F_t(x^t_{k,j})$ is the cumulative distribution function of $\hat{P}(X^t)$, obtained by counting the elements of vector $\mathbb{X}^t_k$ with values lesser or equal to $x^t_{k,j}$. This solution is also known as increasing arrangement. Note that the transformation $\mathcal{G}_k$ is a mapping that "moves" the target smallest value to the source smallest value ($\mathcal{G}_k(x^t_{k,1}) = x^s_{k,1}$), and the target largest value to the source largest value ($\mathcal{G}_k(x^t_{k,n}) = x^s_{k,m}$). For the intermediate target points ($1 < k < m$), the mapping is given by a barycenter in the source domain, according to Equation (3) applied with the choice of the cost function c^p.

Recall that sorting a vector of n elements requires $O(n \log n)$ steps. For a specific attribute k, once vectors $\mathbb{X}^s_k$ and $\mathbb{X}^t_k$ are sorted, computing $\mathcal{G}(x^t_{k,j})$ for every $x^t_{k,j} \in \mathbb{X}^t_k$ requires a single pass over both vectors that is $O(n)$ steps (provided $n \geq m$). Given that the number of numerical features to process is typically small compared to the number of instances ($d' \ll n$), the required $O(d' n \log n)$ computational time of our method is favorably compared to the typical $O(n^2 \log n)$ time of regularized optimal transport.

3.4 One-Dimensional Mapping of Categorical Features

Let $D_k = \{e^k_1, \ldots, e^k_{n_k}\}$ be the (non-ordered) set of values taken by a categorical feature, where $k \in \{d'+1, \ldots, d\}$ is the feature index, and n_k is the number of unique values in D_k. We denote the set of source and target values for categorical features by

$$\forall k \in \{d'{+}1, \ldots, d\}\,,\ \mathbb{X}_k^s = (x_{k,1}^s, \ldots, x_{k,i}^s, \ldots, x_{k,m}^s) \in D_k^m\,,$$
$$\forall k \in \{d'{+}1, \ldots, d\}\,,\ \mathbb{X}_k^t = (x_{k,1}^t, \ldots, x_{k,j}^t, \ldots, x_{k,n}^t) \in D_k^n\,.$$

As for numerical features, we propose to rely on Kantorovich's optimal transport to transform the target features into the source feature, but we cannot rely on a ℓ^p norm cost function as in Section 3.3. Instead, we need to define a cost function between the categorical values.

One could tailor this cost metric to the specificity of the learning problem at hand. In our experiments, we use a generic strategy that can be applied to any categorical features, by defining the cost in terms of the occurrence frequency [7]:

$$\forall e_l^k, e_r^k \in D_k\,,\ c_{cate}(e_l^k, e_r^k) = C_{l,r}^k = \begin{cases} 1 & \text{if } e_l^k = e_r^k\,, \\ \dfrac{1}{1 + \log(\frac{1}{v_l^k})\log(\frac{1}{v_r^k})} & \text{otherwise,} \end{cases} \tag{12}$$

where $v_l^k \in (0,1]$ is the frequency of occurrences of the value e_l^k for the k-th feature in $\mathbb{X}_k^s \cup \mathbb{X}_k^t$ (respectively $v_r^k \in (0,1]$ is the frequency of the value e_r^k). In Equation (12), we write $C_{l,r}^k$ for the entry of the cost matrix $C^k \in \mathbb{R}^{n_k \times n_k}$. Then, we state our optimal transport problem on a categorical feature in terms of the following coupling matrix $\gamma^k \in \mathbb{R}_+^{n_k \times n_k}$ in place of Equation (2):

$$\gamma^k = \operatorname*{argmin}_{\gamma \in \mathit{\Gamma}^k} \left\langle C^k, \gamma \right\rangle = \operatorname*{argmin}_{\gamma \in \mathit{\Gamma}^k} \sum_{l=1}^{n_k} \sum_{r=1}^{n_k} C_{l,r}^k \gamma_{l,r}\,, \tag{13}$$

with

$$\mathit{\Gamma}^k = \left\{ \gamma \in \mathbb{R}_+^{n_k \times n_k} \;\middle|\; \frac{\left|\{i \mid x_{k,i}^s{=}e_l^k\}\right|}{m} = \sum_{j=1}^{n_k} \gamma_{l,j} \text{ and } \frac{\left|\{j \mid x_{k,j}^t{=}e_r^k\}\right|}{n} = \sum_{i=1}^{n_k} \gamma_{i,r} \right\}.$$

That is, we perform the optimal transport on the n_k categorical values instead of on the n source (and m target examples). Typically, $n_k \ll n$, and the computation is thus less expensive than the original problem. However, unlike numerical features where we can compute a barycenter thanks to Equation (3), the barycenter for categorical features is difficult to define. We distinguish two strategies.

Numerical embedding. In some case, a categorical feature has a numerical representation $\phi_k : D_k \to \mathbb{R}^{d_k}$ (for example, a real vector embedding like the common "Word2Vec" representation). In such cases, we use the barycenter of numerical representations as the adapted value:

$$\mathcal{G}_k(e_r^k) = \operatorname*{argmin}_{x \in \mathbb{R}^{d_k}} \sum_{l=1}^{n_k} \gamma_{l,r}^k \, c_{num}^p(x\,, \phi_k(e_r^k))\,. \tag{14}$$

Stochastic mapping. More generally, we can define a stochastic transformation of categorical features. The probability of transforming one point $x^t_{k,j}$ to $x^s_{k,i}$ is

$$P(\mathcal{G}_k(e^k_r) = e^k_l) = \frac{\gamma^k_{l,r}}{\sum_{i=1}^{n_k} \gamma^k_{i,r}} \,. \tag{15}$$

Based on this method, to predict on a target example $x^t_j \in \mathbb{X}^t_k$ with the source predictor h^s (Equation 5), we perform a Monte Carlo estimation by sampling the value of every categorical feature thanks to Equation (15), and performing an average of the outputs predicted by h^s on these stochastic transformations.

3.5 Weakly Supervised Feature Selection

We have noticed in various experiments on different domain adaptation tasks that some features contribute more to domain adaptation than others. Instead of adapting all the features, the adaptation of well-selected features has better performance. In the scenario where few labeled target data are available, we propose to use them in order to select the features for which a transformation is beneficial.

The proposed feature selection scheme is a greedy algorithm. At initialization, no feature is adapted. Then, at each step of the process, we transform one feature of the target set. The selected feature is the one allowing the greatest accuracy increase on the small set of labeled target samples. The process is stopped when the accuracy improvement is no more significant according to the following criteria: we perform bootstrap sampling on the target label set, and we stop if more than one half of improvements are negative. Therefore, the remaining unselected features are unchanged when the target to source mapping is applied.

4 Experiments

In this section, we evaluate our methods on three datasets. The first one is the well-known sentiment classification task on Amazon review datasets. The second one is a fraud detection dataset that we collect from a Kaggle competition. The third one is the real-life industrial fraud detection dataset collected from a company which is one of the leaders of payment systems. Note that the methods that we have compared to are CORAL [15] which aligns the correlation matrices between domains, Deep Adaptation Networks (DAN) [9] which projects the source domain into a latent space and Domain Adversarial Neural Networks (DANN) [6] which generates the features that are both discriminating and invariant to the change of domains. Extended details concerning the experimental framework, additional results and related works are provided as supplementary material.[7]

[7] The supplementary material, the code and data for the first two tasks are available on Github: https://github.com/marrvolo/CDA. Due to confidential reasons, the real-life fraud dataset is not shared.

4.1 Datasets

Amazon Reviews. This dataset contains reviews across different product categories from Amazon. Every review is a short text with associated score. According to the scores, reviews are labeled as positive or negative. Using supervised learning, one can build a sentiment classification model to estimate the different points of view of buyers. However, the model trained on one category does not generalize perfectly to another category. Some words appear frequently in reviews of one category but not the others.

The dataset we use here is the same as [2] and [6]. It is a class-balanced small dataset with 4 domains: Books (B), DVDs (D), Electronics (E) and Kitchen appliances (K). Each domain has 2000 training examples and around 4000 test examples, and features are bags-of-words. We generate a new feature representation using the mSDA unsupervised auto-encoder [2]. We generate two mSDA representations, one with all 5000 words dimensions, and another with only the most frequent 400 words dimensions. Similar to Chen et al. [2] we use a mSDA transformation with 5 layers. However, instead of stacking all hidden representations, we take only the hidden representation of the last layer.

Kaggle Fraud Detection. This is a public dataset from Kaggle IEEE-CIS Fraud Detection competition.[8] The objective is to predict the probability that an online transaction is fraudulent. The dataset contains transactions issued from different devices. We consider the mobile device as the source domain and the desktop device as the target domain. The dimension of the raw dataset is over 400 and contains missing values. Since the paper does not focus on the transformation of features with missing values, we remove all dimensions with more than 1% missing values and all transactions with missing values. The dimension of the dataset after preprocessing is 120 and the proportion of fraud is around 8%. We train the source models in a 4-folds cross-validation way and name the models from Model1 to Model4.

Real Fraud Detection. This real-life fraud detection dataset consists of two domains: Belgian dataset and German dataset. The two datasets are real anonymous clients' transactions in production environments. They have the same mixture types of features, the number of categorical features is 8 and the number of numerical features is 23, numerical features are nearly all generated manually and are normalized between 0 and 1. We use 3 months of data, from July 2018 to October 2018. There are 180 million transactions for Belgian dataset and 90 million for German dataset.

4.2 Results

Amazon Reviews.[9] We first test our method on SVM model and compare our proposed method with two other methods that can be easily applied to a target to source adaptation problem. CORAL [15] aligns the correlations between

[8] https://www.kaggle.com/c/ieee-fraud-detection

[9] See supplementary material for the results in the 400 dimensional dataset.

Table 1. Amazon reviews results (prediction accuracy) using SVM on 5000 dimensions.

		No Retrain		
Domains	**Source**	**CORAL**	**OT**	**1D OT**
$B \leftarrow D$	.8407	.8229	.7420	**.8438**
$B \leftarrow E$	.7692	.7283	.6856	**.8010**
$B \leftarrow K$	**.8445**	.7527	.7098	.8403
$D \leftarrow B$	**.8358**	.8132	.7433	.8351
$D \leftarrow E$	.8002	.7511	.7150	**.8355**
$D \leftarrow K$	**.8704**	.7806	.7248	.8639
$E \leftarrow B$	.7695	.7162	.6864	**.7854**
$E \leftarrow D$	.8084	.7406	.6979	**.8109**
$E \leftarrow K$	.8889	.8724	.7900	**.8970**
$K \leftarrow B$	.7870	.7357	.7025	**.8002**
$K \leftarrow D$	.8042	.7501	.7180	**.8145**
$K \leftarrow E$	**.8794**	.8560	.7863	.8785

domains and OT [4] is the optimal transport method in multidimensional space without any regularization. Here we use Euclidean distance as the cost metric. As shown in Table 1 (the left arrow shows the target to source direction of adaptation), our proposed 1D optimal transport outperforms the two others for this sentiment analysis task. Furthermore, we notice that the multidimensional optimal transport and CORAL consistently perform negative transfer. The potential reason is that the dataset size is not sufficiently large to accurately capture the multidimensional relations between attributes.

Table 2 reports experimental results using a neural network predictor. Our method is compared with CORAL, OT and with two deep adaptation methods DAN [9] and DANN [6]. For a given dataset, the adaptation of CORAL and OT are deterministic, so no standard deviation is reported. The DANN and DAN models are trained with 30 different random states, and we report the average accuracy and associated standard deviation. On this 5000 dimensional dataset, the 1D optimal transport gets overall results comparable to the best adaptation method, while CORAL and OT still perform negative transfer.

Kaggle Fraud Detection.[10] As shown in Table 3, we evaluate our coordinate-wise domain adaptation method by transforming all features (1D OT), only numerical features (1D OT NUM) and only categorical features (1D OT CATE) in an unsupervised setting. We also compare the weakly supervised adaptation method by selecting significant features using few labeled target data to the retraining supervised (marked by **ws**) model. The column named "%n" shows the percentage of labeled target data that are available. The column "*d*" shows the number of adapted features on average. The reported metric is the area under the precision-recall curve (PR-AUC). Similar to experiment settings of Amazon

[10] See supplementary material for the results on Model3 and Model4.

Table 2. Amazon reviews results (prediction accuracy) using neural networks on 5000 dimensions.

		Retrain		**No Retrain**		
Domains	**Source**	**DANN**	**DAN**	**CORAL**	**OT**	**1D OT**
$B \leftarrow D$	.8276	.8382 ±.0041	**.8421 ±.0019**	.8179	.7200	.8374
$B \leftarrow E$	.7410	.7845 ±.0194	**.8256 ±.0020**	.7319	.6912	.8033
$B \leftarrow K$	.8267	.8361 ±.0049	**.8546 ±.0027**	.7589	.7222	.8412
$D \leftarrow B$	.8221	**.8353 ±.0021**	.8320 ±.0041	.7867	.7381	.8181
$D \leftarrow E$	.7919	.8073 ±.0119	**.8468 ±.0012**	.7268	.7081	.8185
$D \leftarrow K$	.8501	.8666 ±.0057	**.8691 ±.0016**	.7513	.7169	.8433
$E \leftarrow B$	.7711	.7743 ±.0078	**.8091 ±.0050**	.7146	.6860	.7905
$E \leftarrow D$	.8095	.7987 ±.0050	**.8148 ±.0052**	.7322	.6943	.8064
$E \leftarrow K$	.8904	.8853 ±.0057	.8911 ±.0012	.8723	.7991	**.8925**
$K \leftarrow B$	.7899	**.8022 ±.0019**	.7995 ±.0033	.7357	.7034	.7957
$K \leftarrow D$	.8095	.8213 ±.0018	**.8259 ±.0016**	.7481	.7052	.8139
$K \leftarrow E$	.8748	.8745 ±.0013	**.8749 ±.0010**	.8563	.7882	.8744

reviews task, we repeat 30 times for no deterministic experiments (DANN and DAN) and report their standard deviations.

Table 3 first presents results using Gradient Boosting Decision Tree (GBDT) models. The coordinate-wise adaptation method on categorical features achieve the best performance among all unsupervised adaptation methods without retraining. The CORAL has a significant negative transfer on this dataset. Both weakly supervised methods (marked by **ws**) have improved the performance compared to the adaptation of all features. However, in the situation where 10% of target labels are available, retraining a GBDT model gets the best performance. Compared to weakly supervised methods with 10% of labels, the method using only 1% target labels has comparable performance and less adapted features.

Regarding the neural networks source models, the coordinate-wise adaptation method on all features has the best performance among all unsupervised adaptation methods. In the case where few labeled target examples are available, the weakly supervised adaptation methods have selected the most significant features for domain adaptation and improved the prediction performance. Interestingly, the value of the standard deviation of selected features is smaller in the case of 1% of target labels in most of the experiments. This may due to the fact that selected features is far less than the ones in the case of 10% of target labels.

Fig. 1 reveals the progression of the accuracy metric of our coordinate-wise domain adaptation method using a greedy search feature selection approach. These graphs are obtained using all target labels, in order to show that in an ideal case few feature transformations are required to achieve a good adaptation, and that adapting all features might hurt.

Real Fraud Detection. Similar performance can be observed on the real-life fraud detection adaptation task from Table 4. The models are evaluated in 3 different

Table 3. PR-AUC scores for domain adaptation models and non adaptative models (annotated by a † mark) on Kaggle Fraud Detection dataset.

GBDT model

Retrain	**Method**	$\%n$	**Model1**	d	**Model2**	d
YES	Train on target†	1	.5604 ±.0316	-	.5646 ±.0361	-
	Train on target†	10	**.7332 ±.0191**	-	**.7299 ±.0185**	-
NO	Source model†	0	.6712	-	.6625	-
	CORAL NUM	0	.6041	112	.5660	112
	1D OT NUM	0	.6497	112	.6615	112
	1D OT CATE	0	.6986	8	.7011	8
	1D OT	0	.6748	120	.6898	120
	1D OT (ws)	1	.6848 ±.0064	9 ±3	.7075 ±.0089	13 ±4
	1D OT (ws)	10	.7061 ±.0024	36 ±4	.7159 ±.0017	42 ±6

Neural Network model

Retrain	**Method**	$\%n$	**Model1**	d	**Model2**	d
YES	Train on target†	1	.3660 ±.0745	-	.3960 ±.0584	-
	Train on target†	10	.5893 ±.0595	-	.5941 ±.0501	-
	DAN	0	.6489	120	.6372	120
	DANN	0	.6398	120	.6079	120
NO	Source model†	0	.5912	-	.5990	-
	CORAL	0	.5490	120	.5840	120
	1D OT NUM	0	.5872	112	.6008	112
	1D OT CATE	0	.6218	8	.6288	8
	1D OT	0	.6314	120	.6329	120
	1D OT (ws)	1	.6232 ±.0178	16 ±6	.6202 ±.0218	15 ±5
	1D OT (ws)	10	**.6561 ±.0100**	40 ±8	**.6548 ±.0101**	36 ±10

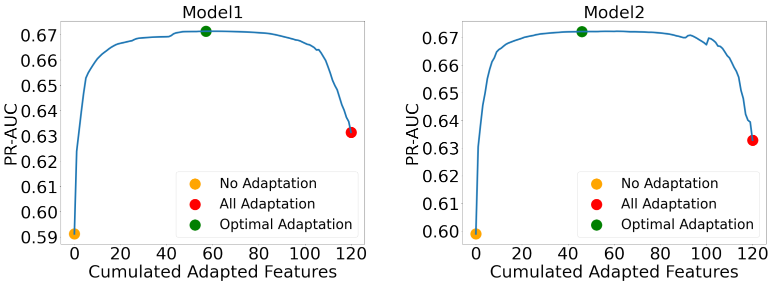

Fig. 1. Feature selection greedy algorithm with neural networks source model on Kaggle fraud detection task (idealized scenario involving all target labels).

periods. Notice that in all adaptation tasks except the month of September, at least one of our proposed adaptation methods outperforms the source model.

The weakly supervised adaptation methods in GBDT and neural networks show that with only 1% of label information, the feature selection can achieve comparable performance as the one with 10% of label information and transform fewer features. Both weakly supervised adaptation methods improve the performance than the adaptation of all features. As expected, the standard deviation of adaptation performance with 1% of target labels slightly larger than that with 10% of target labels. Although, in the neural networks, the adaptation method like DAN and DANN have the best performance, they require the retraining of the model and the adaptation is performed in a latent space which is difficult to interpret. Moreover, these two methods are only for neural networks and our proposed methods are totally model independent.

Table 4. PR-AUC scores for domain adaptation models and non adaptative models (annotated by a † mark) on the Real Fraud Detection dataset.

GBDT model								
Retrain	**Method**	$\%n$	**July**	d	**August**	d	**September**	d
YES	Train on target†	1	.0441 ±.0348	-	.0170 ±.0126	-	.0698 ±.0395	-
	Train on target†	10	.2554 ±.0683	-	.0870 ±.0592	-	.2923 ±.0854	-
NO	Source model†	0	.2595	-	.1546	-	.3840	-
	CORAL NUM	0	.2296	23	.1709	23	.3245	23
	1D OT NUM	0	.2323	23	.2357	23	.3684	23
	1D OT CATE	0	**.2705**	8	.1738	8	**.3886**	8
	1D OT	0	.2383	31	.2199	31	.3695	31
	1D OT (ws)	1	.2630 ±.0071	8 ±5	.2097 ±.0235	9 ±4	.3830 ±.0107	10 ±5
	1D OT (ws)	10	.2697 ±.0043	14 ±2	**.2515 ±.0165**	14 ±3	.3837 ±.0112	15 ±1

Neural Network model								
Retrain	**Method**	$\%n$	**July**	d	**August**	d	**September**	d
YES	Train on target†	1	.0412 ±.0380	-	.0171 ±.0163	-	.0486 ±.0417	-
	Train on target†	10	.2325 ±.0679	-	.0918 ±.0456	-	.2637 ±.0626	-
	DAN	0	.3073	31	.2010	31	.2881	31
	DANN	0	**.2849**	31	.1966	31	**.2945**	31
NO	Source model†	0	.2351	-	.1852	-	.2607	-
	CORAL	0	.1548	31	.1796	31	.1039	31
	1D OT NUM	0	.2341	23	.1985	23	.2552	23
	1D OT CATE	0	.2459	8	.1718	8	.2247	8
	1D OT	0	.2392	31	.1965	31	.2371	31
	1D OT (ws)	1	.2511 ±.0163	11 ±4	.2009 ±.0154	10 ±4	.2554 ±.0127	9 ±4
	1D OT (ws)	10	.2599 ±.0160	14 ±3	**.2100 ±.0181**	14 ±3	.2591 ±.0153	14 ±3

5 Conclusion and Future Works

This article introduced a new target to source perspective for domain adaptation tasks. An unsupervised and a weakly supervised coordinate-wise transformation are proposed and achieve comparable results to the state-of-the-art methods. In addition, the proposed method is parameter-free and can be easily applied to various families of pre-trained models such as neural networks and decision trees. Although we have shown experimentally that transforming a small subset of target features leads to better predictions. As for future research, we aim to reveal the further relevance between these selected features and domain adaptation tasks.

Acknowledgements. This work was partially supported by the Canada CIFAR AI Chair Program.

References

1. Altschuler, J., Niles-Weed, J., Rigollet, P.: Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. In: NeurIPS (2017)
2. Chen, M., Xu, Z.E., Weinberger, K.Q., Sha, F.: Marginalized denoising autoencoders for domain adaptation. In: ICML (2012)
3. Courty, N., Flamary, R., Habrard, A., Rakotomamonjy, A.: Joint distribution optimal transportation for domain adaptation. In: NeurIPS (2017)
4. Courty, N., Flamary, R., Tuia, D., Rakotomamonjy, A.: Optimal transport for domain adaptation. IEEE transactions on pattern analysis and machine intelligence **39**(9), 1853–1865 (2016)
5. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. In: NIPS (2013)
6. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. JMLR **17**(1), 2096–2030 (2016)
7. Jones, K.S.: A statistical interpretation of term specificity and its application in retrieval. Journal of documentation (1972)
8. Kantorovich, L.: On the translocation of masses (1958)
9. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: ICML (2015)
10. Monge, G.: Mémoire sur la théorie des déblais et des remblais. Histoire de l'Académie Royale des Sciences de Paris (1781)
11. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks **22**(2), 199–210 (2010)
12. Peyré, G., Cuturi, M.: Computational optimal transport. Foundations and Trends in Machine Learning **11**(5-6), 355–607 (2019)
13. Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of statistical planning and inference **90**(2), 227–244 (2000)
14. Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P.V., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: NIPS (2008)
15. Sun, B., Feng, J., Saenko, K.: Correlation alignment for unsupervised domain adaptation. In: Domain Adaptation in Computer Vision Applications, pp. 153–171. Springer (2017)