



**HAL**  
open science

# Interpretable Domain Adaptation Using Unsupervised Feature Selection on Pre-trained Source Models

Luxin Zhang, Pascal Germain, Yacine Kessaci, Christophe Biernacki

► **To cite this version:**

Luxin Zhang, Pascal Germain, Yacine Kessaci, Christophe Biernacki. Interpretable Domain Adaptation Using Unsupervised Feature Selection on Pre-trained Source Models. 2021. hal-03325509v1

**HAL Id: hal-03325509**

**<https://hal.science/hal-03325509v1>**

Preprint submitted on 24 Aug 2021 (v1), last revised 20 Sep 2022 (v2)

**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.

# Interpretable Domain Adaptation Using Unsupervised Feature Selection on Pre-trained Source Models

Luxin Zhang<sup>\*1</sup>, Pascal Germain<sup>2</sup>, Yacine Kessaci<sup>3</sup>, and Christophe  
Biernacki<sup>4</sup>

<sup>1</sup>*Worldline Company and Inria, Lille, France*

<sup>2</sup>*Université Laval, Québec, Canada.*

<sup>3</sup>*Worldline Company, Lille, France*

<sup>4</sup>*Inria, Lille, France*

## Abstract

We study a realistic domain adaptation setting where one has access to an already existing “black-box” machine learning model. Indeed, in real-life scenarios, an efficient pre-trained source domain predictive model is often available and required to be preserved. Our work extends a method that has been recently proposed to tackle this specific problem, yet providing an interpretable *target to source* transformation, by seeking a coordinate-wise adaptation of the feature space. However, this method requires partially labeled target data to select the features to be adapted. In contrast, we address the more challenging unsupervised version of this domain adaptation scenario. We propose a new pseudo-label estimator over unlabeled target examples, based on the rank-stability in regards to the source model prediction. Such estimated “labels” are further used in a feature selection process to assess whether each feature needs to be transformed to achieve adaptation. We provide theoretical foundations of our method as well as an efficient implementation. Numerical experiments on real datasets show particularly encouraging results since approaching the supervised case, where one has access to labeled target samples.

## 1 Introduction

Domain Adaptation (DA) methods deal with a scenario where training data, the so-called source domain data, and test data, the so-called target domain data, are drawn from different distributions Pan & Yang (2009); Torralba & Efros (2011).

---

<sup>\*</sup>Email: luxin.zhang@worldline.com

This is a ubiquitous challenge in industrial machine learning applications. For example, to expand companies’ businesses, a payment fraud detection system trained in one geographical localization, say one country (source domain), may be used to detect fraudsters in another country (target domain), where people have different payment habits. Directly training such a predictive model on a new country is not desirable since one has scarce labeled target domain data. However, abundant unlabeled data are usually easy to get. Classical DA methods address this problem often by seeking a latent space where input distributions of target and source domains coincide Pan et al. (2010); Baktashmotlagh et al. (2013) or by transforming source domain data to match distributions of the target domain Sugiyama et al. (2008); Sun et al. (2017). Current approaches based on deep neural networks rely on adversarial learning to generate domain invariant features Ganin et al. (2016); Tzeng et al. (2017). However, such methods require retraining a predictive model after or during the adaptation, which is undesirable in a real-life industrial setting. Indeed, in many real-life scenarios, a well-performing pre-trained source domain predictive model is often given. The pre-trained predictive model can be from various model types: neural networks, decision trees, expert rules, to name a few. Retraining such an aggregation of models requires tedious hyper-parameter fine-tuning and can be even unfeasible due to no longer accessible expertise. As a result, it is natural to consider this pre-trained model as a “black-box” and reuse it directly. Furthermore, the automatically generated features by deep neural networks are generally not interpretable; hence provide no insights to understand drifts between source and target domains. Alternatively, we propose to stand in a *target to source* DA scenario: one transforms target data into the source ones and predicts target domain labels using the pre-trained source model directly without supplementary retraining. Moreover, we focus on the adaptation problem of tabular data where the input space contains categorical attributes as well as numerical ones, thus complexifying the analysis. All these constraints require the adaptation method to be predictive *model-agnostic*, *retraining-free*, and *feature-type independent*.

To address such a challenging *target to source* DA scenario, we propose to use *coordinate-wise* optimal transports for adaptation (note that we initiated this line of research in an early conference paper Zhang et al. (2021)). As we show further in experiments, transformation functions of *coordinate-wise* optimal transports are easily interpretable and can seamlessly adapt numerical features as well as categorical ones. Moreover, leveraging a weakly-supervised feature selection process, we enhanced the interpretability and prediction performance of our method by providing a sparse and ordered transformation function. More precisely, we ranked features of input space by their contributions to DA tasks and focused on the adaptation of features that have the most contributions. From a business point of view, the selected small subset of features can reveal the source of gaps between source and target domains and provides business experts, with or without machine learning backgrounds, interpretability to gain more insights for better understanding different domains. Although our previous proposed method achieved empirically state-of-the-art performances over several real-life DA tasks, a small subset of labeled target data were required during

Table 1: Comparison of different DA methods. (i) *classical adaptation method*: SA Fernando et al. (2013), (ii) *deep adaptation methods*: DANN Ganin et al. (2016), (iii) *classical adaptation method* with feature selection: OT Feature Selection Gautheron et al. (2018).

Advantages	SA	DANN	OT Feature Selection	<b>Ours</b>
Model-Agnostic	✓	✗	✓	✓
Retraining-Free	✗	✗	✗	✓
Feature-type Free	✗	✓	✗	✓
Huge Dataset	✗	✓	✗	✓
Interpretable	✓	✗	✓	✓

the feature selection process; thus, we were in a weakly-supervised DA scenario. However, in practice, target domain labels are often missing, hence limiting the spectrum of use of the previous method.

We extend this DA method, typically the idea of feature selection over pre-trained source models, to an unsupervised DA setting through the following contributions: (i) We propose a general pseudo-label estimator relying on the stability of ranks of predictions to counterbalance the absence of labels in target domains. (ii) We leverage such estimated pseudo-labels to propose an unsupervised objective function to select and adapt the features that contribute the most to the tasks. (iii) We provide theoretical foundations of the method and give an efficient implementation. (vi) We represent our method over challenging DA tasks via interpretable illustrations empirically.

We organize the paper as follows. Section 2 reviews some DA methods in a classical setting and some pseudo-labeling techniques. Section 3 gives the formalization of the *target to source* DA problem and introduce details of the *coordinate-wise* adaptation method. We propose our unsupervised feature selection method in Section 4 in addition to its theoretical foundations. Section 5 provides an efficient implementation of our method. In Section 6, we empirically evaluate our proposition on the Amazon review benchmark and two challenging real-life fraud detection tasks. Finally, we conclude and give some future perspectives in Section 7.

## 2 Related Work

**Domain Adaptation (DA).** According to classes of transformation functions, DA methods can be roughly categorized as *deep adaptation methods* that rely on deep neural networks, and *classical adaptation methods* that do not use neural networks to adapt data.

Deep neural networks are shown to be capable of extracting transferable

features between different tasks Yosinski et al. (2014). Recent *deep adaptation methods* enhance this transferability by plugging into neural networks an adaptation layer Long et al. (2015); Gong et al. (2016); Long et al. (2017) to minimize Maximum Mean Discrepancy (MMD) or other statistical moments of different orders Zellinger et al. (2017); Chen et al. (2020) between source and target distributions. Another popular paradigm relies on adversarial learning Goodfellow et al. (2014) to generate features that are domain invariant Ganin et al. (2016); Tzeng et al. (2017); Long et al. (2018); Saito et al. (2018). Such methods have achieved state-of-the-art adaptation performances, especially in image datasets. However, all these methods project source and target data into a common latent space and require retraining of the predictive model. These approaches are not desirable when a “black-box” pre-trained model is given and cannot be retrained due to the manual defined expert rules. Moreover, DA methods that rely on characteristics of image data for DA may not be easily applied to tabular ones. Besides, several recent works Liang et al. (2020); Kurmi et al. (2021); Yeh et al. (2021) that also leverage a pre-trained source model for DA stand in a setting where source domain data are not available while assuming that pre-trained models are neural networks. However, such methods do not address our problem, as we focus on the adaptation of tabular data and have no restriction over the family of predictive models. In our case, source domain models can be of any type (*e.g.*, decision trees or expert rules); thus, one should have access to source domain data so that our adaptation method can apply to generic “black-box” predictive models. Moreover, deep neural networks, especially adversarial networks, are challenging to train and need lots of manual tuning to find the optimal hyper-parameter and trade-off between classification and adversarial losses or regularization terms. Such hyper-parameter searching is not straightforward in an unsupervised DA scenario You et al. (2019).

The *classical adaptation methods* are more suited for tabular data than image data. Some address the DA problem by minimizing measures like Kullback-Leibler Divergence Shimodaira (2000); Sugiyama et al. (2008) or MMD Pan et al. (2010); Baktashmotlagh et al. (2013); Long et al. (2013) between source and target domain distributions. Others align target and source domain correlation matrices Sun et al. (2017) or principal axes Fernando et al. (2013). Recent research works that leverage optimal transport theory Monge (1781); Kantorovich (1958) and advances in computational optimal transport Cuturi (2013), consider the DA problem as finding the optimal transport plan between source and target domains Perrot et al. (2016); Courty et al. (2016, 2017). However, most of these methods are not scalable to transform massive datasets or fail to adapt categorical features.

**Feature selection and DA.** To take into account both interpretability and a huge dataset of mixed feature-types, our proposition leverages a *coordinate-wise* transformation Zhang et al. (2021) and feature selection DA methods Satpal & Sarawagi (2007); Uguroglu & Carbonell (2011); Gautheron et al. (2018). Classical feature selection DA methods predict target labels relying solely on selected domain invariant features. However, label-relevant features could be discarded and decrease prediction performance. Alternatively, we keep all features such that one can directly use pre-trained source models to predict target labels. In

addition, features that contribute to DA are adapted to mitigate gaps between source and target domains. Besides, the selected features by the unsupervised method offer an immediate overview of the source of drifts between domains. Table 1 highlights the added values of our proposition compared to some typical adaptation methods.

**Pseudo-labeling and DA.** The pseudo-labeling method is first used to guess and annotate unlabeled data in semi-supervised learning Zhu (2005). Namely, one iteratively trains a model with real and pseudo-labels and updates pseudo-labels using the most confident predictions. In an unsupervised DA scenario, one annotates unlabeled target data using the most confident predictions of source models Lee (2013). Leveraging this technique, Long et al. Long et al. (2018) propose to align target and source domain conditional distributions for DAs. Some other works Chen et al. (2011); Saito et al. (2017) combine co-training or tri-training techniques to get estimations of pseudo-labels of target domains. In French et al. (2018) and Shu et al. (2018), the authors use a teacher network to predict target labels and refine decision boundaries. Inspired by the success of pseudo-labeling methods, we propose a new pseudo-label estimator relying on rank stability of predictions that helps selecting features that contribute the most to DA tasks. Different from the aforementioned methods that use discrete class labels with the most confident predictions as pseudo-labels, our approach gives probability scores that unlabeled examples belong to one class. In many real-life applications, probability scores represent more precisely inter-example relationships than just class labels, especially when classes are highly unbalanced.

### 3 Target to Source Domain Adaptation

We first introduce the formalization of *target to source* DA scenarios and some basic notations. Of note, the current section essentially recaps the framework we previously introduced for the weakly-supervised setting Zhang et al. (2021), on which we build the contribution of forthcoming Section 4 for unsupervised feature selection.

The objective of *target to source* DA consists of finding a predictive model independent transformation  $\mathcal{G} : \mathcal{X} \rightarrow \mathcal{X}$  that aligns input marginal distributions and output conditional distributions between source and target domains, that is,

$$\begin{cases} P(X^s) = P(\mathcal{G}(X^t)), \\ P(Y^s|X^s) = P(Y^t|\mathcal{G}(X^t)), \end{cases} \quad (1)$$

where  $\mathcal{X}$  is the input space that contains numerical and categorical dimensions.  $X^s \in \mathcal{X}$  (resp.  $X^t \in \mathcal{X}$ ) refers to the input variable of source domain (resp. target domain), and  $Y^s \in \mathcal{Y}$  (resp.  $Y^t \in \mathcal{Y}$ ) refers to the output variable of source domain (resp. target domain), where  $\mathcal{Y}$  is the output space. We also denote by  $\mathbb{X}^t = \{x_i \in \mathcal{X} | i = 1, \dots, n_t\}$ , a set of target inputs drawn from  $P(X^t)$  with  $n_t$  examples, and by  $\mathbb{X}^s = \{x_j \in \mathcal{X} | j = 1, \dots, n_s\}$ , a set of source inputs drawn from  $P(X^s)$ . We study the binary classification problem

where  $\mathcal{Y} = \{0, 1\}$ , and we assume that we are given a pre-trained source model  $h_s : \mathcal{X} \rightarrow [0, 1]$  corresponding to the source domain optimal Bayes predictor, that is  $h_s(x) = P(Y^s=1|X^s=x)$ .

We also assume the existence of an unknown target domain optimal predictor, denoted by  $h_t(x) = P(Y^t=1|X^t=x)$ . Accordingly, the second condition of Equation (1) can be reformulated as  $h_t(x) = h_s \circ \mathcal{G}(x)$ . Notice that  $P(Y^s) = P(Y^t)$  is a necessary condition that Equation (1) has a solution, and we can further infer that the following equality holds in this case:

$$P(h_s(X^s)) = P(h_t(X^t)). \quad (2)$$

Consequently, our proposed DA approach consists of three steps: (i) To handle the case where  $P(Y^s) \neq P(Y^t)$ , we first calibrate outputs of pre-trained source models in order to apply the same adaptation process as the case where  $P(Y^s) = P(Y^t)$ , relying on the works of Saerens et al. Saerens et al. (2002) and Lin et al. Lin et al. (2002); (ii) To transform target domain distributions to the source ones, we stand in an unsupervised setting and seek an optimal transport function  $\mathcal{G}$  that verifies  $P(X^s) = P(\mathcal{G}(X^t))$ . (iii) To take into account weakly-labeled (pseudo-labeled) target data, we use a feature selection process to find features that contribute the most to DA tasks. In addition, we encourage the sparsity of transformation  $\mathcal{G}$  so that only the selected features are transformed. Besides, the selected order intuitively reveals the contribution of each feature.

### 3.1 Output Calibration

Although it is a common assumption in DA to consider  $P(h_s(X^s)) = P(h_t(X^t))$ , this condition may be violated in practice. A particular case of violation is the so-called *label shift* Japkowicz & Stephen (2002), where  $P(Y^s) \neq P(Y^t)$ . In this paper, we stand in the unsupervised case where neither the target predictor  $h_t$  nor the target labels  $Y^t$  are given, while the target domain output marginal distribution  $P(Y^t)$  is accessible. For example, in a fraud detection system, despite the lack of labels in target domains, one may have the proportion of fraud estimated by experts. Furthermore, we suppose that, for a binary classification problem, if

$$P(Y^s) = P(Y^t), \quad (3)$$

then we have

$$P(h_s(X^s)) = P(h_t(X^t)). \quad (4)$$

As input-output pairs  $(X^s, Y^s)$  of source domains are given, and target domain output marginal distribution  $P(Y^t)$  is also known, then, one can get Equation (3) by re-weighting source examples by classes. Although re-weighted source data have no *label shift* compared to target ones, the pre-trained “black-box” model  $h_s$  will no longer be optimal in the re-weighted source domain. Thus, in light of the works Saerens et al. (2002) and Lin et al. (2002), we propose to

calibrate  $h_s$  such that it gives optimal predictions for examples in the re-weighted source domain. For the simplicity of analysis, we note respectively  $X^p$  and  $Y^p$  the input and output variables of re-weighted source domain. By definition, we have

$$P(Y^p) = P(Y^t) \text{ and } P(X^p|Y^p) = P(X^s|Y^s).$$

**Proposition 1** (Source Model Calibration). Let  $h_s$  be the pre-trained optimal binary classifier in the source domain. The optimal predictor  $h_p(x) = P(Y^p = 1|X^p = x)$  in the re-weighted source domain is obtained by:

$$h_p(x) = \frac{h_s(x)w(1)}{h_s(x)w(1) + (1 - h_s(x))w(0)}, \quad (5)$$

where

$$w(y) = \frac{P(Y^t = y)}{P(Y^s = y)}.$$

The details of proof are given in Appendix. The proposition suggests that the difference between binary output marginal distributions of source and target domains can be mitigated by calibrating outputs of the pre-trained “black-box” model. For the sake of simplicity, hereafter, we abuse notations to use  $X^s$  and  $Y^s$  instead of  $X^p$  and  $Y^p$  to express the source domain where data are already re-weighted to match the target domain output distribution, and  $h_s$  refers to the calibrated optimal predictive model in the re-weighted domain.

### 3.2 Optimal Transport for Coordinate-wise Domain Adaptation

The optimal transport problem was first introduced by Monge in the 18th century Monge (1781) and further developed by Kantorovich in the mid-20th Kantorovich (1958). Intuitively, the original Monge-Kantorovich problem looks for minimal effort to move masses of dirt to fill a given collection of pits. It is naturally suited for DA problems Courty et al. (2016) of tabular data, as it offers a principled method to transform seamlessly numerical and categorical target distributions to source ones. Central to optimal transport methods is the notion of a *cost function* between a source point and a target point, denoted by

$$c : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}. \quad (6)$$

Moreover,  $C \in \mathbb{R}^{n_t \times n_s}$  denotes the *cost matrix* between source and target training points such that  $C_{i,j} = c(x_i, x_j)$  corresponds to the cost of moving weight from  $x_i \in \mathbb{X}^t$  to  $x_j \in \mathbb{X}^s$ . As discuss towards the end of this section, the cost may be defined both for categorical and numerical features.

Based on these concepts, we present below the Kantorovich Kantorovich (1958) formulation of the multidimensional 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*  $\gamma \in \mathbb{R}_+^{n_t \times n_s}$ , where  $\gamma_{i,j}$  corresponds to the *weight* to be moved from  $x_i \in \mathbb{X}^t$  to  $x_j \in \mathbb{X}^s$ . The set of admissible coupling matrices is given by

$$\Gamma = \left\{ \gamma \in \mathbb{R}_+^{n_t \times n_s} \mid w_i^t = \sum_{j'=1}^{n_s} \gamma_{i,j'} \text{ and } w_j^s = \sum_{i'=1}^{n_t} \gamma_{i',j} \right\},$$

where  $w_i^t$  (Resp.  $w_j^s$ ) is the weight of  $x_i \in \mathbb{X}^t$  (Resp.  $x_j \in \mathbb{X}^s$ ). Typically, we consider that the mass is uniformly distributed among each point, *i.e.*  $w_i^t = 1/n_t$  and  $w_j^s = 1/n_s$ , but the framework allows reweighing the samples, such that

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

Then, the optimal coupling matrix  $\gamma^*$  is obtained by solving

$$\gamma^* = \underset{\gamma \in \Gamma}{\operatorname{argmin}} \langle C, \gamma \rangle = \underset{\gamma \in \Gamma}{\operatorname{argmin}} \sum_{i=1}^{n_t} \sum_{j=1}^{n_s} C_{i,j} \gamma_{i,j}. \quad (7)$$

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

$$\mathcal{G}(x_i) = \underset{x' \in \mathcal{X}}{\operatorname{argmin}} \sum_{j=1}^{n_s} \gamma_{i,j}^* c(x', x_j). \quad (8)$$

The solution  $x' \in \mathcal{X}$  of Equation (8) minimization problem is commonly referred to as the *barycenter mapping* in the optimal transport literature. For unseen target examples  $x$  drawn from  $P(X^t)$  while  $x \notin \mathbb{X}^t$ , we project  $x$  to its nearest  $x_i$  according to  $c(x_i, x)$  and then get its source DA using Equation (8).

However, Equation (7) is a linear optimization problem, when  $n_s = n_t = n$ , the computational complexity is  $O(n^3)$  which is not scalable to huge datasets.

Therefore, we restrict the class of transformations  $\mathcal{G}$  by considering all features as independent in transfer tasks and then propose to use one-dimensional optimal transport individually on each attribute to transform target data to the source domain, which is the so-called *coordinate-wise* adaptation Zhang et al. (2021). The transformation  $\mathcal{G}$  is decomposed by feature-wise transformations  $\mathcal{G}_k$ , such that  $\mathcal{G} = [\mathcal{G}_1, \dots, \mathcal{G}_k, \dots, \mathcal{G}_d]$ , where  $d$  is the number of features of the input space. Each transformation  $\mathcal{G}_k$  solves the adaptation problem of the  $k$ -th feature by aligning marginally input distributions of this feature between target and source domains, that is,  $P(X_k^s) = P(\mathcal{G}_k(X_k^t))$ .

**Numerical Attribute.** In the case where the  $k$ -th feature is a numerical one,  $\mathcal{X}_k = \mathbb{R}$ , and the cost function of this dimension is defined as

$$\forall x_i^k, x_{i'}^k \in \mathcal{X}_k, \quad c_{\text{num}}^p(x_i^k, x_{i'}^k) = |x_i^k - x_{i'}^k|^p,$$

where  $x_i^k$  and  $x_{i'}^k$  stand for the  $k$ -th dimension of inputs  $x_i$  and  $x_{i'}$ . Instead of solving  $\mathcal{G}_k$  relying on Equation (7) and Equation (8), there is a closed-form solution of the Kantorovich optimization problem Peyré & Cuturi (2019):

$$\mathcal{G}_k(x^k) = (F_k^{s-1} \circ F_k^t)(x^k), \quad (9)$$

where  $F_k^s$  and  $F_k^t$  are respectively cumulative distribution functions of  $P(X_k^s)$  and  $P(X_k^t)$ . This solution is also known as increasing arrangement.

**Categorical Attribute.** In contrast, if the  $k$ -th feature is categorical, we have  $\mathcal{X}_k = D_k$ , where  $D_k = \{e_1^k, \dots, e_{n_k}^k\}$  is the (non-ordered) set of values taken by the  $k$ -th categorical feature, and  $n_k$  is the number of unique values in  $D_k$ .

We use a generic strategy that can be applied to any categorical features, by defining the cost in terms of the occurrence frequency Jones (1972):

$$\begin{aligned} \forall e_l^k, e_r^k \in D_k, c_{\text{cate}}(e_l^k, e_r^k) &= C_{l,r}^k \\ &= \begin{cases} 0 & \text{if } e_l^k = e_r^k, \\ 1 - \frac{1}{1 + \log(\frac{1}{v_l^k}) \log(\frac{1}{v_r^k})} & \text{otherwise,} \end{cases} \end{aligned} \quad (10)$$

where  $v_l^k \in (0, 1]$  (Resp.  $v_r^k \in (0, 1]$ ) is the frequency of occurrences of the value  $e_l^k$  (Resp.  $e_r^k$ ) for the  $k$ -th feature. In Equation (10), 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 (7):

$$\gamma^k = \operatorname{argmin}_{\gamma \in \Gamma^k} \langle C^k, \gamma \rangle = \operatorname{argmin}_{\gamma \in \Gamma^k} \sum_{l=1}^{n_k} \sum_{r=1}^{n_k} C_{l,r}^k \gamma_{l,r}, \quad (11)$$

with

$$\Gamma^k = \left\{ \gamma \in \mathbb{R}_+^{n_k \times n_k} \mid \frac{|\{i \mid x_i^k = e_l^k\}|}{n_t} = \sum_{j=1}^{n_k} \gamma_{l,j} \right. \\ \left. \text{and } \frac{|\{j \mid x_j^k = e_r^k\}|}{n_s} = \sum_{i=1}^{n_k} \gamma_{i,r} \right\},$$

where  $x_i^k$  (Resp.  $x_j^k$ ) is the  $k$ -th dimension of  $x_i \in \mathbb{X}^t$  (Resp.  $x_j \in \mathbb{X}^s$ ). Therefore, we perform the optimal transport on the  $n_k$  categorical values  $e^k$  instead on the  $n_t$  target (and  $n_s$  source) examples. Typically,  $n_k \ll n_t$ , and the computation is thus less expensive than the original problem. However, unlike numerical features where we can compute a barycenter thanks to Equation (8), the barycenter of categorical features is difficult to define. Consequently, we propose a *stochastic mapping* strategy to tackle this problem. The probability of transforming one value  $e_l^k$  to  $e_r^k$  is

$$P(\mathcal{G}_k(e_l^k) = e_r^k) = \frac{\gamma_{l,r}^k}{\sum_{j=1}^{n_k} \gamma_{l,j}^k}. \quad (12)$$

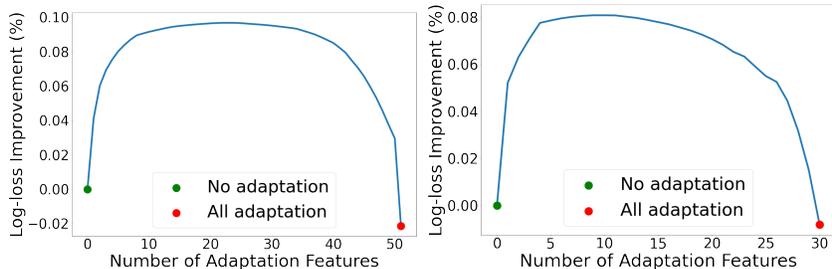


Figure 1: Evolution of log-loss improvements according to the number of adapted features. Left: The Kaggle fraud detection dataset with a neural network pre-trained model. Right: The real fraud detection dataset with a tree-based pre-trained model.

The final prediction score is averaged over each possible transformation weighted by Equation (12).

Therefore, the global computational complexity of the proposed *coordinate-wise* DA is

$$d_{\text{num}} \times (n_s \log(n_s) + n_t \log(n_t)) + \sum_k^{d_{\text{cate}}} (n_k)^3,$$

where  $d_{\text{num}}$  and  $d_{\text{cate}}$  respectively refer to the number of numerical and categorical attributes. Although this *coordinate-wise* approach does not take into account correlations between features, it is appealing for the simplicity and the transformations of huge tabular datasets.

### 3.3 Supervised Feature Selection

We have noticed in various experiments of different tasks that features contribute differently to DAs. Figure 1 illustrates decreasing percentages of log-loss (log-loss improvement) in a feature selection process. At initialization, no feature is adapted. Then, at each step of the process, we transform one more feature to the source domain that has the minimal value of log-loss over target data. We stop when all features are adapted. Note that we use target labels to select the feature to transform at each step only for illustration, whereas they are not accessible in practice. Interestingly, instead of adapting all features, the adaptation of a well-selected subset of features has better performance (larger value of log-loss improvement). Therefore, in the *target to source* DA scenario where a “black-box” source model  $h_s$  is available, we aim to seek a subset of features  $\alpha \in \mathcal{A}$  to adapt, where  $\mathcal{A}$  contains all possible subsets of features of the input space  $\mathcal{X}$ .

The selected features are adapted one-by-one using *coordinate-wise* optimal transport mapping functions, while other features remain identical without being excluded from the dataset. Consequently, we can use the source model directly on adapted target data to predict labels. The resulting predictive model of

target domain is expressed by  $h_t^\alpha = h_s \circ \mathcal{G}^\alpha$ , where  $\mathcal{G}^\alpha$  is the transformation function that adapts the feature subset  $\alpha \in \mathcal{A}$ . We show further in experiments that  $\alpha$  generally contains just a few features. Thus, the transformation  $\mathcal{G}^\alpha$  is very sparse. Let  $\mathcal{G}^*$  be the transformation that verifies Equation (1), then the optimal target predictor is expressed by  $h_t = h_s \circ \mathcal{G}^*$ .

In a supervised setting, one tackles this feature selection problem by leveraging on labeled data in target domains to find the optimal subset of features that minimizes the expected risk, that is,

$$\alpha^* = \operatorname{argmin}_{\alpha \in \mathcal{A}} \mathbb{E}_{(x,y) \sim P(X^t, Y^t)} \left[ |h_t^\alpha(x) - y| \right], \quad (13)$$

where  $P(X^t, Y^t)$  is the joint distribution of  $(X^t, Y^t)$ , and the solution  $\alpha^*$  is the optimal subset of features to adapt. One may note that  $\mathcal{G}^{\alpha^*}$  could be different from  $\mathcal{G}^*$ , as  $\mathcal{G}^{\alpha^*}$  is restricted to the class of *coordinate-wise* transformations, whereas  $\mathcal{G}^*$  refers to the optimal transformation among all possible adaptation functions.

However, in typical DA problems,  $P(X^t, Y^t)$  is unknown, thus directly minimizing Equation (13) is not feasible. When few labeled target data are available, *i.e.*, in a weakly supervised setting, we proposed Zhang et al. (2021) to seek the subset of features that minimizes the following term:

$$\hat{\alpha}^* = \operatorname{argmin}_{\alpha \in \mathcal{A}} \frac{1}{n_q} \sum_{(x,y) \in \mathbb{Q}} \left| h_t^\alpha(x) - y \right|, \quad (14)$$

where  $\mathbb{Q} = \{(x_i, y_i) | i = 1, \dots, n_q\}$  contains  $n_q$  labeled target examples, and  $n_q$  is very small. Despite the promising performance obtained by weakly-supervised methods, one still needs labeled data in the target domain.

## 4 Unsupervised Feature Selection Based on Stable Pseudo-Labeling

In this paper, we address a more challenging unsupervised DA setting where target labels, even few labeled examples, are not available ( $n_q = 0$ ). However, the set of target inputs  $\mathbb{X}^t$  is given. Intuitively, if one gets an estimator  $\hat{h}$  to annotate some specific target examples  $x \in \mathbb{X}^t$  approximately, one can solve the feature selection problem by injecting  $\hat{h}$  into Equation (14) to replace  $y$ . Since  $\hat{h}$  does not generalize to new target examples, one cannot directly use it as the target domain predictor. Nevertheless, it can serve as an adequate “anchor” for the unsupervised feature selection process. Such approximate annotations are the so-called pseudo-labels.

One of the most well-known strategies estimates target pseudo-labels using predictions of source models directly, assuming that high-confidence predictions are correct Zhu (2005); Chen et al. (2011); Saito et al. (2017). However, we illustrate further on a toy example (Figure 2) that this approach could be unstable

and gives incorrect pseudo-labels in some cases. In contrast, instead of pseudo-labeling target examples with confident predictions, we estimate pseudo-labels of examples that have rank-stable predictions under different transformation functions.

## 4.1 Rank Stability

In this section, we define a notion of *stable inputs* suited for our DA task, we propose a pseudo-label estimator, and we prove that our method gives pseudo-labels equal predictions of the optimal *coordinate-wise* adaptation function, making it legitimate to be applied to the unsupervised feature selection.

**Definition 2** (Stable Inputs). A target input example  $x_i \in \mathbb{X}^t$  is called stable over  $\mathcal{A}$  if its rank of prediction remains unchanged after being adapted by *coordinate-wise* transformations over all different feature-subsets from  $\mathcal{A}$ , that is,

$$\forall x_{i'} \in \mathbb{X}^t, x_{i'} \neq x_i, \forall \alpha, \beta \in \mathcal{A}, \\ h_t^\alpha(x_i) > h_t^\alpha(x_{i'}) \iff h_t^\beta(x_i) > h_t^\beta(x_{i'}). \quad (15)$$

Accordingly, we denote by  $\mathbb{X}^{\mathcal{A}}$  a set of all such target examples.

Furthermore, since  $\alpha^*$  is the optimal subset of features to adapt, we suppose that predictions of  $h_t^{\alpha^*}$  and  $h_t$  on target domain data have the same distribution, that is,

$$P(h_t^{\alpha^*}(X^t)) = P(h_t(X^t)). \quad (16)$$

Under this mild assumption, the following proposition is verified.

**Proposition 2** (Property of Stable Inputs). Given that  $\mathcal{A}$  contains the optimal subset of features  $\alpha^*$ , we have

$$\forall x \in \mathbb{X}^{\mathcal{A}}, \forall \beta \in \mathcal{A}, H_s^{-1} \circ H_\beta(h_t^\beta(x)) = h_t^{\alpha^*}(x), \quad (17)$$

where  $H_s$  and  $H_\beta$  are respectively cumulative distribution functions of  $h_s(X^s)$  and  $h_t^\beta(X^t)$ .

*Proof.* As ranks of predictions can be naturally expressed by cumulative distribution functions, given Equation (15), and  $\alpha^*, \beta \in \mathcal{A}$ , we have

$$\forall x \in \mathbb{X}^{\mathcal{A}}, H_{\alpha^*}(h_t^{\alpha^*}(x)) = H_\beta(h_t^\beta(x)), \quad (18)$$

where  $H_{\alpha^*}$  refers to the cumulative distribution function of  $h_t^{\alpha^*}(X^t)$ . According to Equations (2) and (16), we have  $H_{\alpha^*} = H_t = H_s$ . Analogously,  $H_t$  is the cumulative distribution functions of  $h_t(X^t)$ . Replacing  $H_{\alpha^*}$  to  $H_s$  in Equation (18), we get

$$H_s(h_t^{\alpha^*}(x)) = H_\beta(h_t^\beta(x)).$$

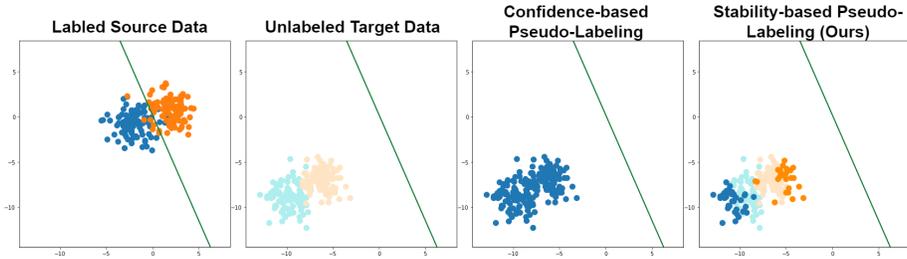


Figure 2: Left: labeled source domain data. Middle left: unlabeled target domain data. Middle right: pseudo-labels given by confidence-based methods. Right: pseudo-labels provided by our proposition over stable target examples. The ground truth of target data is shown in light colors, and pseudo-labels of the target domain are shown in deep colors. The green line in the sub-figures is the pre-trained source domain predictor.

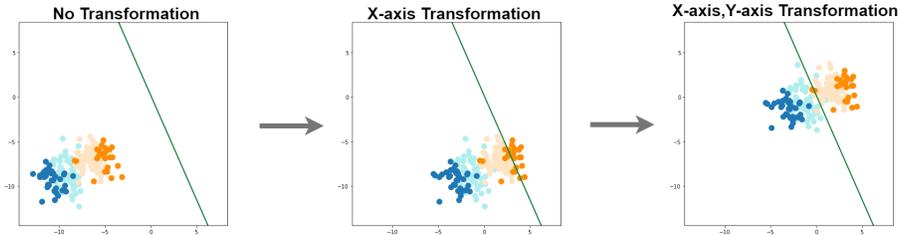


Figure 3: Steps of *coordinate-wise* transformations using stability-based pseudo-labels.

As  $H_s$  is invertible, we have

$$h_t^{\alpha^*}(x) = H_s^{-1} \circ H_\beta(h_t^\beta(x)),$$

which proves Equation (17).  $\square$

Note that,  $h_t^{\alpha^*}$  is the optimal *coordinate-wise* adaptation function that we expect to get. Therefore, we define the pseudo-label estimator  $\hat{h}^\beta(x)$  by the following formula:

**Definition 3** (Rank-stable Based Pseudo-label Estimator).

$$\forall x \in \mathbb{X}^{\mathcal{A}}, \forall \beta \in \mathcal{A},$$

$$\hat{h}^\beta(x) = H_s^{-1} \circ H_\beta(h_t^\beta(x)) = h_t^{\alpha^*}(x). \quad (19)$$

An example is illustrated in Figure 2, where we compare two different pseudo-labeling techniques on a toy dataset. In this example, we first identify stable target examples over  $\mathcal{A}$ , where  $\mathcal{A}$  contains all feature subsets of the two-dimensional space. Then we estimate pseudo-labels using  $\hat{h}^\beta$ . Stable examples

$x$  with  $\hat{h}^\beta(x) > 0.5$  are colored as blue and the others as orange. Note that according to Equation (19), the choice of  $\beta$  does not affect the pseudo-labels for all  $x \in \mathbb{X}^A$ . The pseudo-labeled examples are further used to adapt target data in order to fit the pre-trained source model (Figure 3). We provide details of this process in Section 5. In contrast, confidence-based pseudo-labeling methods (Figure 2 middle right) consider predictions of target examples far from the decision boundary as correct; thus, all examples are pseudo-labeled as blue and provide no information to help DAs. One may note that pseudo-labels given by our method can be close to the decision boundary of the two classes. Indeed, our method is agnostic to the prediction value but relies only on the rank-stability over  $\mathcal{A}$ .

## 4.2 Relaxation of Rank Stability

The method described in the last subsection is ineffective when the number of examples in  $\mathbb{X}^A$  is scarce, as it needs enough *stable* elements to reach a diversity that faithfully expresses the global distribution of  $X^t$ . Therefore, we introduce a relaxation of Definition 2 to compensate for this scarcity.

**Definition 4** ( $\delta$ -stable Inputs). A target input example  $x \in \mathbb{X}^t$  is called  $\delta$ -stable over  $\mathcal{A}$  if

$$\mathcal{B}(x) = \max_{\mu, \nu \in \mathcal{A}} (|\hat{h}^\mu(x) - \hat{h}^\nu(x)|) \leq \delta. \quad (20)$$

Accordingly, we denote by  $\mathbb{X}_\delta^A$  an input set that contains all such target examples. By setting  $\delta = 0$ , one can retrieve Definition 2.

Although one can still use  $\hat{h}^\beta$  to estimate pseudo-labels for  $x \in \mathbb{X}_\delta^A$ , it is uncertain that Proposition 2 is verified. Intuitively, a larger  $\delta$  results in a richer  $\mathbb{X}_\delta^A$  but with a higher risk of violating Proposition 2. In the remainder of this section, we formally analyse the effects of this relaxation over the feature selection process and propose the corresponding unsupervised objective function.

In their seminal DA analysis, Ben-David et al. (2007) proposed to upper bound the expected target domain risk by a sum of three terms: (i) the source domain risk, (ii) the  $\mathcal{H}$ -divergence defined as

$$d(P(X^t), P(X^s)) = 2 \sup_{h \in \mathcal{H}} \left| \mathbb{E}_{x \sim P(X^t)} [h(x) \neq 1] - \mathbb{E}_{x \sim P(X^s)} [h(x) \neq 1] \right|$$

to measure the discrepancy between source ( $P(X^s)$ ) and target ( $P(X^t)$ ) input marginal distributions, and (iii) an intrinsic error between true labeling functions of two domains. We denote by

$$\mathcal{E}(\alpha) = \mathbb{E}_{x \sim P(X^t)} \left[ |h_t^\alpha(x) - h_t^{\alpha^*}(x)| \right] \quad (21)$$

the target domain risk between a label predictor  $h_t^\alpha$  and the optimal one  $h_t^{\alpha^*}$ . We notice that the drift between  $\mathbb{X}_\delta^A$  and  $\mathbb{X}^t$  is known as sample selection bias Heckman (1979); thus, we can upper bound  $\mathcal{E}(\alpha)$  by considering  $\delta$ -stable examples  $x \in \mathbb{X}_\delta^A$  as the “source” domain.

**Theorem 1.** Given a subset of features  $\alpha \in \mathcal{A}$ , for all  $\beta \in \mathcal{A}$ ,  $\delta \in [0, 1]$ , the following inequality holds:

$$\mathcal{E}(\alpha) \leq \mathcal{L}(\alpha, \beta, \delta) + \mathcal{D}(\delta), \quad (22)$$

where

$$\mathcal{L}(\alpha, \beta, \delta) = \mathbb{E}_{x \sim P(X_\delta^A)} \left[ |h_t^\alpha(x) - \hat{h}^\beta(x)| \right], \quad (23)$$

$$\mathcal{D}(\delta) = \mathbb{E}_{x \sim P(X_\delta^A)} \left[ \mathcal{B}(x) \right] + \frac{1}{2} d(P(X^t), P(X_\delta^A)), \quad (24)$$

$P(X_\delta^A)$  referring to the distribution of  $\delta$ -stable target inputs.

*Proof.* According to the Theorem 1 of Ben-David et al. (2007), we have

$$\mathcal{E}(\alpha) \leq \mathbb{E}_{x \sim P(X_\delta^A)} \left[ |h_t^\alpha(x) - h_t^{\alpha^*}(x)| \right] + \frac{1}{2} d(P(X^t), P(X_\delta^A)) + C.$$

As examples in  $\mathbb{X}^t$  and in  $\mathbb{X}_\delta^A$  have the same true labeling function, the constant term  $C = 0$ . We use the triangle inequality on the expectation term of the upper bound and we get

$$\begin{aligned} \mathcal{E}(\alpha) &\leq \mathcal{L}(\alpha, \beta, \delta) \\ &+ \mathbb{E}_{x \sim P(X_\delta^A)} \left[ |\hat{h}^\beta(x) - h_t^{\alpha^*}(x)| \right] + \frac{1}{2} d(P(X^t), P(X_\delta^A)). \end{aligned}$$

Since  $H_{\alpha^*} = H_s$  according to Equation (2) and Equation (16), and relying on Equation (17), we get

$$h_t^{\alpha^*}(x) = H_s^{-1} \circ H_{\alpha^*}(h_t^{\alpha^*}(x)) = \hat{h}^{\alpha^*}(x).$$

As  $\beta, \alpha^*$  are subsets of  $\mathcal{A}$ , by replacing  $h_t^{\alpha^*}(x)$  by  $\hat{h}^{\alpha^*}(x)$ , and relying on Equation (20), we have

$$\begin{aligned} |\hat{h}^\beta(x) - h_t^{\alpha^*}(x)| &= |\hat{h}^\beta(x) - \hat{h}^{\alpha^*}(x)| \leq \mathcal{B}(x) \\ \implies \mathbb{E}_{x \sim P(X_\delta^A)} \left[ |\hat{h}^\beta(x) - h_t^{\alpha^*}(x)| \right] &\leq \mathbb{E}_{x \sim P(X_\delta^A)} \left[ \mathcal{B}(x) \right]. \end{aligned}$$

Theorem 1 is proved.  $\square$

In this bound,  $\mathcal{L}(\alpha, \beta, \delta)$  refers to the feature selection risk over  $\delta$ -stable target examples.  $\mathcal{D}(\delta)$  encompasses the risk related to the stable inputs relaxation, and the discrepancy between  $P(X^t)$  and  $P(X_\delta^A)$ . All elements in this upper bound can be computed without target domain labels. Therefore, we define the unsupervised objective function as

$$\tilde{\alpha}^* = \operatorname{argmin}_{\alpha \in \mathcal{A}} \min_{\beta \in \mathcal{A}} \min_{\delta \in [0, 1]} \left( \mathcal{L}(\alpha, \beta, \delta) + \mathcal{D}(\delta) \right). \quad (25)$$

## 5 Greedy Algorithm for Feature Selection

Inspired by greedy solvers of classical feature selection methods, we propose the following process to solve Equation (25), and we name our proposed adaptation method: *Stability-based feature selection for Coordinate-wise Domain Adaptation (SCDA)*.

### 5.1 Optimization Process

**Set the value of  $\delta$ .** Experimental results show that the optimal  $\delta$  that minimizes the sum of  $\mathcal{L}(\alpha, \delta, \beta) + \mathcal{D}(\delta)$  is close to the minimizer of  $\mathcal{D}(\delta)$ . Therefore, we can rely first on  $\mathcal{D}(\delta)$  to determinate the optimal value of  $\delta$ , and then find the tuple  $(\alpha, \beta)$  that minimizes  $\mathcal{L}(\alpha, \beta, \delta)$ . This approach reduces the complexity by simplifying the combination problem of triplets to the combination problem of tuples. In practice, we use a grid search algorithm to discretize the value space of  $\delta$ , and estimate empirically  $\mathcal{H}$ -divergence by training a classifier to distinguish examples between  $\mathbb{X}^t$  and  $\mathbb{X}_\delta^A$ .

**Find the optimal subset of features.** Another issue that we faced is the high cardinality of  $\mathcal{A}$ . As  $\mathcal{A}$  contains all possible combinations of subsets of features, and the number of subsets of features grows exponentially with the dimensionality of input data, directly using all possible combinations is sometimes not feasible. A similar problem also exists in the supervised feature selection scenario. We use a greedy search algorithm to tackle this issue. As shown in the forthcoming experiments, our method gives encouraging results in practice. Namely, we let  $\mathcal{A}_i$  refer to a subset of  $\mathcal{A}$ , where  $i$  is an index of the greedy search step. As shown in Algorithm 1, at initialization (steps 1-5), no feature is adapted,  $\mathcal{A}_0$  contains an empty feature subset and all singleton feature sets. We start by minimizing  $\mathcal{D}(\delta)$  to find and set the optimal value of  $\delta$  (step 5). The variable *Count* at the step 8 is a dictionary with the structure  $\{key : value\}$ . We use a bootstrap technique (steps 9-12) at each iteration of the greedy search to get the temporally optimal  $\tilde{\alpha}_{i+1}$  (step 13). Then we update  $\mathcal{A}_{i+1}$  and continue the process until  $\tilde{\alpha}_{i+1}$  remains unchanged or more than one-half bootstrap datasets are not in accordance with the optimal feature subset.

---

**Algorithm 1** Greedy Search Algorithm

---

- 1: Initialize  $i = 0$ .
- 2: Initialize  $\tilde{\alpha}_0 = \emptyset$ .
- 3: Initialize  $\mathcal{A}_0 = \{\emptyset\} \cup \{\{j\} | \forall j \in \mathcal{A}\}$ .
- 4: Initialize  $v_0 = 1$ .
- 5: Initialize  $\delta = \operatorname{argmin}_\delta \mathcal{D}(\delta)$ .
- 6: **repeat**
- 7:   Get  $X_\delta^{\mathcal{A}_i}$  using Definition 4 on  $\mathcal{A}_i$  and fixed  $\delta$ .
- 8:   Initialize  $\text{Count}[\alpha] = 0$  for all  $\alpha \in \mathcal{A}_i$ .
- 9:   **for**  $X^B$  **in** bootstraps of  $X_\delta^{\mathcal{A}_i}$  **do**
- 10:      $(\alpha, \beta) = \text{minimizer of } \mathcal{L}(\alpha, \beta, \delta) \text{ on } P_{X^B} \text{ and fixed } \delta$ .
- 11:      $\text{Count}[\alpha] = \text{Count}[\alpha] + 1$ .
- 12:   **end for**
- 13:    $\tilde{\alpha}_{i+1} = \operatorname{argmax}_\alpha \text{Count}[\alpha]$ .
- 14:    $\mathcal{A}_{i+1} = \{\tilde{\alpha}_{i+1}\} \cup \{\tilde{\alpha}_{i+1} \cup \{j\} | \forall j \in \mathcal{A} / \tilde{\alpha}_{i+1}\}$ .
- 15:    $v_{i+1} = \text{Count}[\tilde{\alpha}_{i+1}] / \sum_\alpha \text{Count}[\alpha]$ .
- 16: **until**  $\tilde{\alpha}_{i+1}$  unchanged or  $v_{i+1} < 0.5$ ;  $i = i + 1$ .
- 17: **return:**  $\tilde{\alpha}_i$

---

**Complexity of the greedy algorithm.** The computational complexity of the greedy feature selection for a target dataset with  $n_s$  examples and  $d$  dimensions is  $O(n_s d^2)$ . Although the feature selection process seems not scalable over high-dimensional data, we show further in experiments that, in practice, only very few features are selected for DA. Therefore, empirically, the number of operations is far less than  $O(n_s d^2)$ . Moreover, the minimization problem (step 10 in Algorithm 1) consists of finding the optimal tuple  $(\alpha, \beta)$  from  $\mathcal{A}_i \times \mathcal{A}_i$ . One can accelerate this optimization problem by partitioning  $\mathcal{A}_i \times \mathcal{A}_i$  into several smaller search spaces and parallelizing the searching process.

## 6 Experiments

In this section, we evaluate the performances of our adaptation method SCDA on 3 different datasets and 2 types of models: Gradient Boosting Decision Tree (GBDT) and neural networks (NN). We use respectively LightGBM Ke et al. (2017) and PyTorch Paszke et al. (2019) packages in Python to implement these models. The source code of experiments is available on Github: [www.github.com/marrvolo/SCDA](http://www.github.com/marrvolo/SCDA).

### 6.1 Setup Overview

**Kaggle Fraud Detection Dataset.**<sup>1</sup> The dataset contains payment transactions issued from mobile devices and desktop devices, and one aims to predict if an online transaction is fraudulent or not. The raw data dimension is over 400, while most features contain missing values and some are not discriminative. We discard features with more than 1% of missing values and all transactions

---

<sup>1</sup>[www.kaggle.com/c/ieee-fraud-detection](http://www.kaggle.com/c/ieee-fraud-detection)

containing missing values. To discard label-irrelevant features, we first train a predictive model in a supervised setting and predict test data where one feature’s values are randomly shuffled. The feature is considered label-irrelevant if the prediction performance remains nearly the same compared to the not-shuffled test dataset’s performance. After preprocessing, the dataset used in experiments has around 400,000 examples with 43 numerical features and 8 categorical ones. We consider the mobile device as the source domain and the desktop device as the target domain in our DA scenario. The proportions of fraud in each domain are respectively 10% and 7%. We correct *label shift* relying on Proposition 1. We perform a 4-fold validation by dividing mobile and desktop transactions into 4 parts and denoting each part respectively by M-1 to M-4 (mobile device) and D-1 to D-4 (desktop device). Following this setting, the *target to source* DA transforms data from D-i to M-i.

**Real Fraud Detection Dataset.**<sup>2</sup> This dataset consists of real anonymous clients’ transactions from July 2018 to September 2018 of two geographical domains: Belgium and Germany. Both datasets have 23 numerical attributes and 7 categorical ones. All features are generated by experts in payment and are thus discriminative. The number of examples in the Belgian dataset is over 30 million and around 15 million in the German dataset. The proportions of fraud are respectively 0.3% and 0.5% in the two countries. We correct *label shift* relying on Proposition 1. Note that classes of labels are highly unbalanced in Kaggle and real fraud detection datasets, thus completing DA tasks. Moreover, in the real fraud detection task, the data distribution naturally “drifts” as time goes. For example, the Belgian data distribution in July is not the same as the one in August. Consequently, we build 3 source domains where each month of Belgian data is considered one domain, and analogously 3 target German data domains. We denote by Bel-1 to Bel-3 the source domains and by Ger-1 to Ger-3 the target domains. Following this setting, the *target to source* DA transforms data from Ger-i to Bel-i. Due to confidential reasons, this dataset is not shared.

Although we evaluate our methods on two fraud detection datasets, the drifts between source and target domains of these two datasets are different. In the Kaggle fraud detection task, the drift comes from the change of device. In contrast, in the real fraud detection task, source and target distributions differ as geographical localization (users’ payment habits) and time change. Hence, they are entirely two different DA tasks.

**Amazon Reviews Dataset.** The dataset contains reviews of buyers on the Amazon website across different categories of products Blitzer et al. (2006). Each review is a small paragraph of texts, transformed into bags-of-words representation and labeled as positive or negative. Note that the sentiment classification model trained using supervised learning to predict buyers’ points of view for one category does not directly generalize to another. Following the setting of Chen et al. (2012), we consider 4 domains: Books (B), DVDs (D), Electronics (E), and Kitchen appliances (K). Each domain has 2,000 training examples and around 4,000 test examples with perfectly balanced labels.

---

<sup>2</sup>This private transaction dataset is provided by an IT company.

We keep the most frequent 400 words dimensions and generate features from bags-of-words representations using mSDA unsupervised auto-encoder Chen et al. (2012) with 5 layers. Instead of stacking all hidden dimensions as Chen et al. (2012) and Ganin et al. (2016), we take only the representation of the last layer. Different from the aforementioned two fraud detection datasets, the features of the Amazon reviews dataset may not have explicit meaning that can be easily interpreted.

**Other General Setup Details.** We pre-train source domain predictive models using supervised learning with 10 different random states and keep the one that achieves the best performance on source domain test datasets. One NN model and one GBDT model are built for each source domain of different tasks following this process. We compare our proposed method (SCDA) with *deep adaptation methods*: DAN Long et al. (2015), DANN Ganin et al. (2016), and MCD Saito et al. (2018), as well as *classical adaptation methods*: CORAL Sun et al. (2017), OTLin with a linear kernel Perrot et al. (2016), and CDA Zhang et al. (2021) that uses *coordinate-wise* transformations to adapt all features without a feature selection process. As described in Section 3.2, SCDA selects features based on *coordinate-wise* transformations. Therefore, we use the POT Flamary & Courty (2017) package to compute such optimal transport mappings. We chose to compare with CORAL and OTLin methods as they perform DAs without modifying the input space of data. As a result, we can extend such methods to address *target to source* DAs and leverage pre-trained GBDT models and NN models to predict target labels. However, they do not adapt categorical features. In contrast, *deep adaptation methods* can transform categorical attributes, whereas they do not satisfy the *target to source* DA setting. *Deep adaptation methods* transform source and target domain data into a latent space and require training a predictive model using source labels during adaptation processes. Our SCDA approach is *feature-type free* and, moreover, *hyper-parameter free*, such that no tuning process is required. *Deep adaptation methods* like DANN and DAN require to find the optimal weight of the adversarial (regularization) term, and MCD requires fine-tuning the learning rate. To select the optimal hyper-parameter for *deep adaptation methods*, we use a grid search process during the training and take the hyper-parameter that minimizes the classification error on test datasets of source domains. For DANN and DAN methods of the Amazon reviews datasets, we seek this weight in the set of values  $\{0.01, 0.05, 0.1\}$ . For the Kaggle fraud detection dataset, the set of values that we used to search the hyper-parameter is  $\{0.005, 0.01, 0.1\}$  for DAN models, and  $\{0.05, 0.1, 0.5\}$  for DANN models. As for MCD models, the Amazon review tasks seek the learning rate among  $\{0.0001, 0.0005, 0.001, 0.005\}$ , and the Kaggle fraud detection tasks seek the learning rate among  $\{0.0005, 0.0007, 0.001\}$ .

## 6.2 Interpretability of Coordinate-wise Domain Adaptation with Greedy Feature Selection

Compared to classical domain adaptation methods Ganin et al. (2016); Fernando et al. (2013), interpretability is one of our coordinate-wise optimal transport

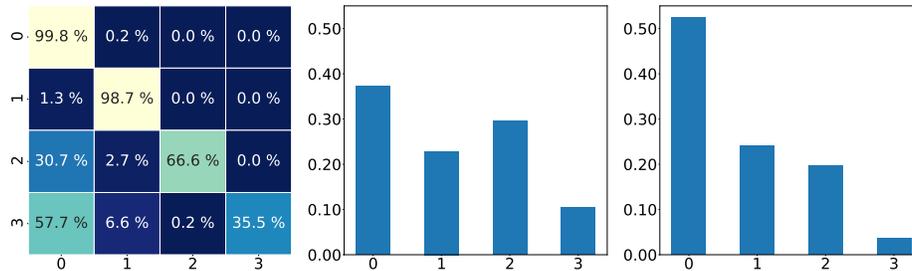


Figure 4: Left: The mapping matrix of a categorical feature where different values are encoded by integer numbers. Middle: The target domain distribution of this categorical feature. Right: The source domain distribution of this categorical feature.

method assets. For example, by investigating the obtained joint probability matrix  $\gamma^k$  of categorical features (Equation (11)), one can get details of mapping between each modality. We show one example in Figure 4 where the mapping matrix of a categorical feature in the Kaggle dataset is illustrated. For this categorical feature, source domain has more encoded value 0 than the one of target domain; thus, by solving Equation (11), the encoded values 2 and 3 in target domain have respectively 30.7% and 57.7% of probability to be mapped to the encoded value 0 in the source domain. This example is obtained by using the general occurrence frequency distance Jones (1972), while business specific distance between categorical values can also be applied to better fit different real-life industrial cases.

Moreover, the greedy feature selection process enhances this interpretability through the selected subset of features  $\tilde{\alpha}_i$ , and their selected orders. Specifically, the feature subset  $\tilde{\alpha}_i$  reveals the source of drifts between source and target domains, and the order provides importance of each feature in DA tasks intuitively. All this information can provide business experts, with or without machine learning backgrounds, insights to better understand different domains. An example is illustrated in Figure 5 where evolution of log-loss risks at each step of greedy feature selection is shown over the Kaggle payment dataset, and the contribution of each feature can be measured by the differences of test risk. In this example, the contribution of the first adapted feature is significantly larger than the others. Consequently, one can investigate this feature to further modelize payment habits of customers from different domains.

### 6.3 Adaptation Performance

We consider the performance of using directly pre-trained source models on target test datasets as baselines. Since we expect predictive models to be well calibrated, we use the decreasing percentage of log-loss compared to the baseline as our evaluation metric. A positive value of metric means the adaptation method improves performances of predicting target domain labels, whereas a negative

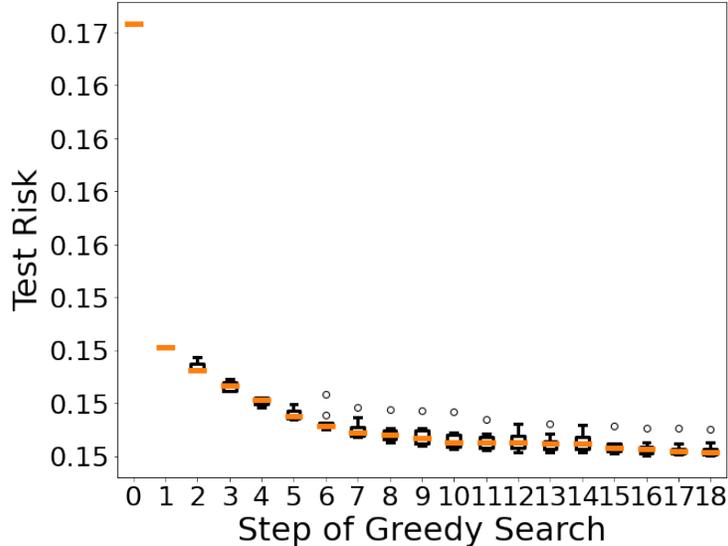


Figure 5: The evolution of log-loss risk at different steps of the greedy algorithm on Kaggle datasets in a supervised scenario. We repeat the feature selection process 10 times and report variations at each step by a box plot.

value refers to the negative transfer. Besides, we also report performances in terms of *area under the precision-recall curve* (PR\_AUC) for fraud detection tasks and in accuracy for the Amazon review tasks. All experiments are repeated 10 times with different random states, and standard deviations are reported to illustrate the stability of methods.

**Kaggle Fraud Detection Dataset.** Table 2 and Table 4 present performances of *classical adaptation methods* and our propositions using GBDT models. SUPERVISED stands for the results of supervised greedy feature selection. We estimate transformation functions of CORAL and CDA using all training input examples of source and target domains. In contrast, SCDA is trained with only 20% of input examples to accelerate the process of greedy search. OTLin does not address the problem of huge datasets; thus, we draw 2,000 input examples from each domain to compute the *target to source* mapping, and we present this result as OTLin(2K). Following the same setting as OTLin(2K), we refer the performance of SCDA on 2,000 examples as SCDA(2K). As our proposition estimates, one dimension by one dimension, the distributions, it can be trained using only a few input examples. Moreover, as CORAL and OTLin do not adapt categorical features, only numerical features are transformed by these two methods, while other methods adapt all features. On average and for each adaptation task, our propositions SCDA and SCDA(2K) achieve the best performance among all adaptation methods. Interestingly, SCDA(2K) achieves better

Table 2: The log-loss improvement (%) of Kaggle fraud detection datasets with GBDT models.

Method	D-1 to M-1	D-2 to M-2	D-3 to M-3	D-4 to M-4	Avg
SUPERVISED	13.41±0.12	15.30±0.36	10.76±0.24	11.46±0.29	12.73
CORAL	-13.76±0.47	-3.25±0.45	-14.06±0.31	-33.30±0.83	-16.09
OTLin(2K)	-11.66±3.60	-0.81±1.82	-14.19±2.29	-17.41±2.93	-11.02
CDA	9.37±0.20	12.36±0.24	6.71±0.19	7.61±0.20	9.01
SCDA	12.69±0.16	14.35±0.08	10.00±0.24	10.19±0.26	11.81
SCDA(2K)	<b>12.82±0.26</b>	<b>14.72±0.14</b>	<b>10.67±0.23</b>	<b>10.93±0.25</b>	<b>12.29</b>

performances than adaptation methods trained using more input examples. One may notice that CORAL and OTLin have a negative transfer using GBDT models. We explain this observation by the fact that they adapt only numerical features, which may be harmful to GBDT models.

Regarding NN models, similar conclusions can be drawn from Table 3 and Table 5. *Deep adaptation methods* use all input examples and map categorical features to numerical spaces by an embedding layer so that categorical values can be transformed as numerical ones. On average, DANN, DAN, and MCD methods improve the performance compared to no-adaptation by respectively 3%, 4%, and 8% in terms of log-loss improvements, while they perform negative transfer in terms of PR\_AUC and have high standard deviations for every adaptation task. Alternatively, our proposition outperforms all adaptation methods and is robust in performances. Experimental results support our intuition that *deep adaptation methods* are complicated to train in an unsupervised setting. Moreover, note that *classical adaptation methods* like CORAL and OTLin perform differently in GBDT models and NN models in terms of log-loss improvements, whereas our propositions are totally model-independent. As we work with unbalanced datasets, improvements of performances in terms of PR\_AUC is in general less than log-loss.

Table 6 reports respectively the number of adapted features selected by our methods with GBDT and NN pre-trained models. Note that in the setting of 2,000 input examples, we adapt fewer features in general. Both the sparsity of selected features and the orders of their selections can help business experts to explain the different payment habits of customers in two domains. Note that methods like CDA improve prediction performances as well; however, no insights on drifts between domains are provided.

**Real Fraud Detection Dataset.** In this task, we follow a similar setting as Kaggle fraud detection experiments, where SCDA is trained this time using 1% of input data to accelerate the feature selection process. Table 7 and Table 9 reveal performances with GBDT pre-trained model in terms of log-loss and PR\_AUC improvements, and Table 8 and Table 10 reveal performances with NN pre-trained models. On average, the SCDA method outperforms other adaptation methods in both metrics when using GBDT models to predict target labels. Although trained with very few input examples, SCDA(2K) is the second best on average in terms of log-loss improvements. Regarding NN models, SCDA outperforms other methods in terms of PR\_AUC, and achieves the best performance in

Table 3: The log-loss improvement (%) of Kaggle fraud detection datasets with NN models.

Method	D-1 to M-1	D-2 to M-2	D-3 to M-3	D-4 to M-4	Avg
SUPERVISED	13.60±0.12	12.61±0.22	21.38±0.18	17.65±0.56	16.31
DANN	2.65±7.41	0.09±3.83	7.15±3.20	4.40±2.89	3.57
DAN	7.36±5.08	-1.27±5.36	6.99±4.90	4.18±5.39	4.31
MCD	9.21±2.73	4.71±3.64	9.29±2.84	9.84±1.80	8.26
CORAL	3.69±0.18	-5.18±0.27	-0.99±0.19	3.00±0.20	0.13
OTLin(2K)	6.33±2.18	-1.67±2.25	1.65±3.11	8.53±1.37	3.71
CDA	8.37±0.29	5.14±0.20	13.55±0.22	12.36±0.32	9.86
<b>SCDA</b>	<b>11.84±0.25</b>	<b>8.85±0.87</b>	<b>19.56±0.46</b>	14.60±0.63	<b>13.71</b>
<b>SCDA(2K)</b>	11.40±1.33	6.88±0.85	18.95±0.41	<b>15.18±0.53</b>	13.10

Table 4: The PR\_AUC improvement (%) of Kaggle fraud detection datasets with GBDT models.

Method	D-1 to M-1	D-2 to M-2	D-3 to M-3	D-4 to M-4	Avg
SUPERVISED	2.53±0.25	4.95±0.60	3.34±0.35	2.92±0.22	3.44
CORAL	-8.98±0.37	-9.91±0.38	-13.13±0.25	-17.96±0.37	-12.50
OTLin	-14.62±1.81	-10.37±1.67	-17.23±2.71	-13.38±2.84	-13.90
CDA	-0.43±0.24	2.37±0.26	-1.45±0.25	-0.91±0.28	-0.10
<b>SCDA</b>	<b>2.15±0.08</b>	4.25±0.13	3.02±0.15	2.55±0.17	2.99
<b>SCDA(2K)</b>	2.01±0.32	<b>4.38±0.23</b>	<b>3.34±0.30</b>	<b>3.04±0.21</b>	<b>3.19</b>

terms of log-loss improvements when adapting from Ger-1 to Bel-1 and has comparable results to CDA (adapting all features) when transforming Ger-3 to Bel-3. However, for the adaptation task Ger-2 to Bel-2, different conclusions can be drawn from two metrics. In terms of PR\_AUC, SCDA outperforms all other methods on this task, whereas it decreases the performance compared to CDA in terms of log-loss. Our assumption is that, leveraging on the work of Ben-David et al. Ben-David et al. (2007), our proposition minimizes the absolute difference between estimated predictions and the optimal prediction values, which is directly related to log-loss. However, the metric PR\_AUC only takes into account orders between predictions. Consequently, relative performances in terms of log-loss and PR\_AUC improvements may have different results. In both GBDT models and NN models, OTLin does not improve performances compared to no-adaptation, which is probably because only a few input examples are used to estimate transformation functions. If we focus only on the best performance of all repetitions, *deep adaptation methods* can achieve promising results in some tasks, for example, the adaptation task Ger-3 to Bel-3 of DANN, and the adaptation task Ger-1 to Bel-1 of MCD. However, the optimal hyperparameter is complicated to find without label information. One may note that DANN suffers in the adaptation task Ger-2 to Bel-2 and fails to converge. Alternatively, our adaptation method is parameter-free and requires no retraining of predictive models; thus, we do not need a time-consuming manual tuning procedure. *Classical adaptation methods* like CORAL and OTLin appear to do not have an adequate adaptation performance since they transform only numerical features and do not address the adaptation problem of extremely unbalanced classes.

Table 5: The PR\_AUC improvement (%) of Kaggle fraud detection datasets with NN models.

Method	D-1 to M-1	D-2 to M-2	D-3 to M-3	D-4 to M-4	Avg
SUPERVISED	4.14±0.34	3.16±0.24	4.73±0.14	4.79±0.34	4.21
DANN	-3.40±5.88	-4.67±3.53	-6.02±4.40	-4.08±3.68	-4.54
DAN	-1.52±5.66	-5.22±4.49	-5.10±3.71	-4.33±5.77	-4.04
MCD	1.76±3.09	-0.99±2.26	-2.84±2.94	-1.03±2.10	-0.77
CORAL	-5.72±0.17	-12.97±0.24	-17.14±0.33	-14.04±0.20	-12.47
OTLin	-5.34±3.15	-12.85±3.15	-17.45±4.78	-8.22±3.21	-10.96
CDA	0.70±0.16	-1.14±0.20	-0.24±0.27	2.04±0.27	0.34
SCDA	<b>2.08±0.35</b>	<b>0.38±0.62</b>	<b>3.67±0.30</b>	2.50±0.35	<b>2.16</b>
SCDA(2K)	1.72±1.74	-0.15±0.58	3.23±0.55	<b>3.27±0.85</b>	2.02

Table 6: Number of adapted features of Kaggle fraud detection datasets with GBDT and NN models.

Method	D-1 to M-1	D-2 to M-2	D-3 to M-3	D-4 to M-4	Avg
SCDA GBDT	14.9±4.8	15.3±3.0	18.9±5.4	16.9±2.9	16.5
SCDA(2K) GBDT	8.0±3.2	8.1±2.1	6.3±1.1	7.1±1.6	7.3
SCDA NN	12.2±3.1	9.9±3.1	12.6±2.6	11.3±4.1	11.5
SCDA(2K) NN	9.9±1.8	3.4±2.1	4.5±1.5	4.9±1.5	5.6

Table 11 reports the number of adapted features of each task. On average, for the pre-trained GBDT models, SCDA selects 8 features, and SCDA(2K) selects 5 features. Regarding NN models, SCDA selects 6 features, and SCDA(2K) selects 4 features. By adapting very few features, we can achieve comparable or better performances than methods that adapt all features.

**Amazon Reviews Dataset.** As introduced in the setup of experiments, different from fraud detection datasets, features of Amazon reviews datasets are generated using a particular neural network: auto-encoder. As a result, individual features may not have interpretable meanings. Table 12 to Table 15 provide results of Amazon reviews datasets using GBDT and NN models in log-loss and accuracy. Adaptation results appear to show that SCDA does not improve performances compared to the all adaptation method (CDA). However, when we look at the number of features adapted by SCDA in Table 16, for GBDT models, SCDA adapts on average only 5.4 features among 400. Regarding NN models, SCDA selects 10.2 features on averages.

The forward feature selection process appears to stop at a very early step and seems to be stuck at local minima. We explain this phenomenon by the fact that features generated by neural networks are highly correlated (see Figure 6). The greedy algorithm that considers at each step only one feature may not be able to identify all features that contribute to DAs in this case. However, in a classical tabular dataset, since all features are generated manually and redundant features are removed, it is less common to have highly correlated features like the ones generated by neural networks.

Nevertheless, by adapting on average 5.4 features for GBDT models and 10.2 features for NN models, our feature selection method SCDA achieves the second-best among other adaptation methods in Table 14, Table 12, and Table 13.

Table 7: The log-loss improvement (%) of real fraud detection datasets with GBDT models.

Method	Ger-1 to Bel-1	Ger-2 to Bel-2	Ger-3 to Bel-3	Avg
SUPERVISED	6.81±0.26	14.52±0.12	10.87±0.23	10.73
CORAL	0.73±0.01	4.13±0.01	-0.91±0.01	1.32
OTLin(2K)	-15.09±3.24	-13.07±2.35	-8.76±2.21	-12.31
CDA	3.08±0.05	9.52±0.04	5.37±0.05	5.99
<b>SCDA</b>	<b>4.13±0.22</b>	<b>12.71±0.36</b>	<b>5.42±0.99</b>	<b>7.42</b>
<b>SCDA(2K)</b>	3.63±0.82	11.81±2.42	4.21±1.07	6.55

Table 8: The log-loss improvement (%) of real fraud detection datasets with NN models.

Method	Ger-1 to Bel-1	Ger-2 to Bel-2	Ger-3 to Bel-3	Avg
SUPERVISED	17.41±0.21	12.10±0.24	8.18±0.86	12.56
DANN	13.90±1.67	-405.76±154.74	<b>9.14±1.72</b>	-127.58
DAN	13.52±1.64	-1.53±3.81	-4.45±5.38	2.52
MCD	16.65±2.49	5.80±3.98	-3.82±6.74	6.21
CORAL	-17.77±0.02	-1.91±0.01	3.15±0.01	-5.51
OTLin(2K)	-14.96±1.38	-9.12±1.43	-9.23±1.47	-11.10
CDA	14.57±0.06	<b>9.58±0.09</b>	5.83±0.07	<b>9.99</b>
<b>SCDA</b>	<b>17.13±0.16</b>	4.63±2.30	5.61±0.34	9.12
<b>SCDA(2K)</b>	14.06±7.28	4.28±3.48	2.64±0.94	6.99

Note that, regarding transport theory methods, our proposition shows better results with an ability to handle a high number of dimensions through a one-by-one feature adaptation compared to OTLin that adapts all features at once relying on the Euclidean distance in a multi-dimensional space.

## 7 Conclusion

In this paper, we proposed a stability-based pseudo-label estimator and an unsupervised feature selection process to address the *target to source* DA problem for tabular data. Different from previous works that rely on tedious parameter fine-tuning or address only numerical features, our proposition (SCDA) is *model-agnostic*, *retraining-free* and *feature-type independent*. In addition, the sparsity and orders of selected features by the unsupervised process can reveal the meaningful source of gaps between source and target domains. As for perspectives, first, we aim to investigate how to apply greedy algorithms to huge dimensional and highly correlated datasets. Second, although our proposed pseudo-labeling technique can be easily generalized to other classes of transformations, we aim to investigate more formally its flexibility over different other adaptation functions.

Table 9: The PR\_AUC improvement (%) of real fraud detection datasets with GBDT models.

Method	Ger-1 to Bel-1	Ger-2 to Bel-2	Ger-3 to Bel-3	Avg
SUPERVISED	6.31±0.63	20.93±2.01	6.08±0.85	11.10
CORAL	-8.75±0.04	-6.97±0.03	-19.37±0.05	-11.70
OTLin(2K)	-50.57±10.44	-40.04±5.41	-44.33±5.39	-44.98
CDA	5.30±0.25	<b>14.95±0.36</b>	-2.11±0.18	6.05
<b>SCDA</b>	<b>5.47±0.87</b>	13.24±1.91	<b>0.76±0.77</b>	<b>6.49</b>
<b>SCDA(2K)</b>	3.77±0.77	12.44±6.37	0.65±2.74	5.62

Table 10: The PR\_AUC improvement (%) of real fraud detection datasets with NN models.

Method	Ger-1 to Bel-1	Ger-2 to Bel-2	Ger-3 to Bel-3	Avg
SUPERVISED	26.49±0.75	2.26±1.26	7.13±0.69	11.96
DANN	-3.33±6.30	-62.05±37.60	<b>8.92±3.32</b>	-18.82
DAN	6.07±8.38	-3.40±6.84	1.32±4.58	1.33
MCD	19.73±8.15	-2.05±5.89	8.01±4.61	8.57
CORAL	-7.44±0.02	-8.50±0.04	-5.57±0.04	-7.17
OTLin(2K)	-41.69±5.15	-31.15±3.36	-36.20±2.99	-36.35
CDA	19.19±0.19	1.58±0.27	1.16±0.19	7.31
<b>SCDA</b>	<b>24.38±0.53</b>	<b>4.09±0.70</b>	1.96±1.63	<b>10.15</b>
<b>SCDA(2K)</b>	19.44±10.39	1.00±3.19	-0.95±3.35	6.50

Table 11: Number of adapted features of real fraud detection datasets with GBDT and NN models.

Method	Ger-1 to Bel-1	Ger-2 to Bel-2	Ger-3 to Bel-3	Avg
SCDA GBDT	8.7±3.1	7.7±0.7	7.7±1.0	8.0
SCDA(2K) GBDT	3.7±2.7	5.1±3.1	6.4±2.3	5.0
SCDA NN	5.7±1.8	4.5±1.4	8.0±1.3	6.0
SCDA(2K) NN	4.5±2.9	3.3±1.8	4.5±2.9	4.1

Table 12: The log-loss Improvement (%) of Amazon Reviews Datasets with GBDT models.

Adaptation Method	D to B	E to B	K to B	B to D	E to D	K to D	B to E	D to E	K to E	B to K	D to K	E to K	Avg
SUP	0.97±0.19	3.68±0.02	5.63±0.02	1.73±0.07	8.83±0.02	11.97±0.04	2.40±0.05	1.37±0.04	1.98±0.03	3.58±0.05	2.65±0.01	2.67±0.05	3.96
CORAL	-0.80±0.00	-2.36±0.00	-0.76±0.00	-0.14±0.00	2.02±0.00	5.78±0.00	-2.28±0.00	-2.65±0.00	0.26±0.00	1.61±0.00	-0.37±0.00	<b>2.32±0.00</b>	0.21
OTLin	-4.33±0.00	-3.99±0.00	-3.76±0.00	-4.41±0.00	-1.90±0.00	0.69±0.00	-5.87±0.00	-5.66±0.00	-5.77±0.00	-2.62±0.00	-4.23±0.00	-1.80±0.00	-3.57
CDA	0.29±0.02	<b>0.60±0.01</b>	2.26±0.02	<b>1.06±0.04</b>	<b>5.09±0.03</b>	<b>8.01±0.03</b>	<b>0.29±0.01</b>	<b>0.02±0.02</b>	<b>1.69±0.02</b>	<b>2.88±0.04</b>	<b>1.07±0.02</b>	2.23±0.01	<b>2.13</b>
<b>SCDA</b>	<b>0.39±0.03</b>	0.55±0.27	<b>2.46±0.57</b>	0.87±0.21	2.79±0.95	7.37±2.29	-0.32±0.07	-0.39±0.15	1.50±0.11	2.15±0.39	-0.08±0.14	1.95±0.02	1.60

Table 13: The log-loss Improvement (%) of Amazon Reviews Datasets with NN models.

Adaptation Method	D to B	E to B	K to B	B to D	E to D	K to D	B to E	D to E	K to E	B to K	D to K	E to K	Avg
SUP	0.96±0.01	12.31±0.01	18.91±0.00	0.17±0.01	5.74±0.00	9.50±0.00	7.88±0.02	3.70±0.01	5.46±0.00	3.97±0.03	4.45±0.05	1.27±0.00	6.19
DANN	-3.02±1.15	6.75±2.32	11.62±2.55	-2.16±0.77	0.07±1.41	1.04±1.88	-1.98±1.25	-1.23±2.14	-0.59±1.26	0.88±0.87	1.44±0.94	-3.96±1.13	0.57
DAN	-1.45±0.30	<b>10.63±0.59</b>	10.81±1.03	-1.19±0.46	4.24±0.50	7.27±0.50	<b>5.51±1.42</b>	1.51±0.74	-1.07±3.50	-1.51±1.11	-3.02±0.95	-1.08±1.22	2.55
MCD	-3.20±2.66	7.65±1.94	10.72±3.65	-0.72±0.55	0.03±2.73	-0.40±4.32	-5.35±4.48	-2.90±2.48	3.21±0.85	1.19±1.39	1.00±2.36	-2.41±2.25	0.42
CORAL	0.31±0.00	7.88±0.00	14.54±0.00	-0.68±0.00	2.65±0.00	6.10±0.00	-1.15±0.00	-2.97±0.00	3.79±0.00	-3.06±0.00	-2.76±0.00	-1.75±0.00	1.91
OTLin	-6.30±0.00	5.74±0.00	10.49±0.00	-4.64±0.00	0.69±0.00	3.12±0.00	-4.45±3.77	-2.23±2.46	1.71±0.98	1.34±0.76	2.08±0.58	-2.41±2.25	0.84
CDA	<b>0.35±0.01</b>	9.77±0.00	<b>16.68±0.00</b>	-0.65±0.90	<b>5.24±0.00</b>	<b>8.84±0.00</b>	5.42±0.00	<b>2.25±0.00</b>	<b>4.90±0.00</b>	<b>1.58±0.00</b>	<b>2.44±0.01</b>	-0.65±0.00	<b>4.69</b>
<b>SCDA</b>	-0.55±0.06	7.85±0.56	14.23±0.22	<b>-0.12±0.04</b>	3.04±0.20	6.38±0.02	3.23±0.51	-0.68±0.15	4.69±0.11	-0.78±0.41	-1.33±1.38	<b>-0.10±0.07</b>	2.99

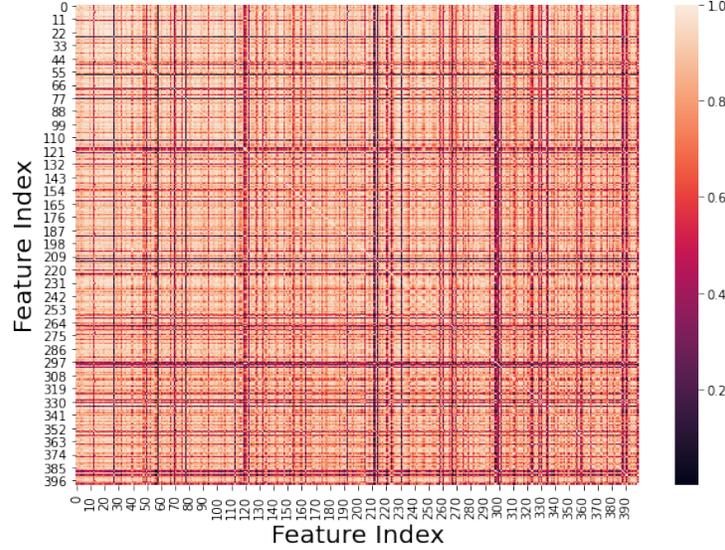


Figure 6: Absolute values of the correlation matrix of mSDA representations of Amazon reviews dataset (Electronics).

Table 14: The Accuracy Improvement (%) of Amazon Reviews Datasets with GBDT models.

Adaptation Method	D to B	E to B	K to B	B to D	E to D	K to D	B to E	D to E	K to E	B to K	D to K	E to K	Avg
SUP	0.97±0.37	0.68±0.02	1.78±0.03	1.54±0.11	6.74±0.03	10.66±0.09	1.26±0.13	0.51±0.06	0.81±0.03	2.17±0.06	1.87±0.05	1.21±0.03	2.52
CORAL	-1.46±0.00	-3.82±0.00	-1.10±0.00	<b>1.51±0.00</b>	3.22±0.00	7.08±0.00	-0.26±0.00	-1.11±0.00	-1.11±0.00	2.21±0.00	0.64±0.00	<b>1.33±0.00</b>	0.68
OTLin	-2.52±0.00	-3.27±0.00	-3.00±0.00	-2.15±0.00	1.02±0.00	4.81±0.00	-2.03±0.00	-1.83±0.00	-2.45±0.00	1.27±0.00	-2.04±0.00	0.41±0.00	-0.98
CDA	<b>-0.13±0.06</b>	-0.03±0.05	0.94±0.03	1.03±0.09	<b>5.84±0.06</b>	<b>8.57±0.05</b>	0.25±0.04	-0.19±0.08	0.37±0.05	<b>2.54±0.03</b>	0.57±0.04	0.94±0.02	<b>1.73</b>
<b>SCDA</b>	-0.28±0.00	<b>0.37±0.27</b>	<b>0.99±0.18</b>	-0.06±0.36	2.06±1.36	7.72±2.58	<b>0.32±0.08</b>	<b>0.27±0.14</b>	<b>0.40±0.09</b>	1.76±0.48	<b>0.73±0.26</b>	1.30±0.06	1.30

Table 15: The Accuracy Improvement (%) of Amazon Reviews Datasets with NN models.

Adaptation Method	D to B	E to B	K to B	B to D	E to D	K to D	B to E	D to E	K to E	B to K	D to K	E to K	Avg
SUP	0.44±0.10	8.66±0.02	14.10±0.02	0.01±0.10	3.79±0.05	7.96±0.03	2.29±0.10	1.31±0.10	2.04±0.01	1.52±0.07	1.95±0.06	0.54±0.02	3.72
DANN	-3.58±1.31	6.07±2.64	10.59±3.20	-1.91±0.81	0.77±1.69	2.17±2.19	-2.50±1.60	-1.93±1.42	0.21±0.72	-0.46±0.50	0.63±0.74	-1.35±0.75	0.73
DAN	-1.07±0.40	<b>7.94±0.59</b>	8.65±1.43	-1.13±0.55	<b>4.30±0.23</b>	<b>7.65±0.27</b>	1.07±1.05	0.21±0.38	-0.63±1.87	-0.26±0.48	-2.21±0.44	-0.03±0.70	1.96
MCD	-2.57±2.13	6.03±1.39	9.09±2.94	-0.52±0.42	-0.41±2.48	0.40±3.98	-2.29±1.37	-1.28±1.12	1.13±0.57	0.14±0.33	1.03±0.56	-1.33±1.37	0.78
CORAL	<b>-0.38±0.00</b>	5.76±0.00	12.90±0.00	-0.35±0.00	2.51±0.00	7.30±0.00	-0.16±0.00	0.40±0.00	1.71±0.00	0.26±0.00	0.32±0.00	-1.10±0.00	2.43
OTLin	-4.02±0.00	5.51±0.00	10.41±0.00	-3.83±0.00	1.36±0.00	5.18±0.00	-1.18±0.00	-2.03±0.00	-1.62±0.00	-0.91±0.00	-1.12±0.00	-2.33±0.00	0.45
CDA	-1.07±0.03	7.47±0.04	<b>13.09±0.04</b>	-0.52±0.04	3.61±0.04	7.54±0.02	<b>2.11±0.02</b>	<b>1.22±0.02</b>	<b>1.76±0.03</b>	<b>0.87±0.02</b>	<b>1.24±0.03</b>	<b>-0.22±0.02</b>	<b>3.09</b>
<b>SCDA</b>	-0.76±0.10	5.72±0.40	11.29±0.13	<b>0.16±0.14</b>	2.35±0.16	6.20±0.08	0.64±0.19	0.23±0.21	1.75±0.06	-0.27±0.42	-0.33±0.98	-0.28±0.04	2.23

Table 16: Number of Adapted Features of Amazon Reviews Datasets with GBDT and NN models.

Method	D to B	E to B	K to B	B to D	E to D	K to D	B to E	D to E	K to E	B to K	D to K	E to K	Avg
SCDA GBDT	3.1±0.8	4.9±2.7	3.8±2.6	4.8±2.1	3.7±1.5	6.2±2.9	5.5±1.1	8.0±2.4	6.1±2.8	5.6±1.5	8.8±3.5	4.5±0.9	5.4
SCDA NN	1.9±1.1	3.2±1.6	7.8±3.4	7.0±2.8	5.5±2.7	8.5±2.5	11.3±5.3	9.7±4.6	30.5±5.0	10.0±2.6	7.4±2.1	19.6±2.3	10.2

## References

- Baktashmotlagh, M., Harandi, M. T., Lovell, B. C., and Salzmann, M. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, pp. 769–776, 2013.
- Ben-David, S., Blitzer, J., Crammer, K., Pereira, F., et al. Analysis of representations for domain adaptation. *NIPS*, 19:137, 2007.
- Blitzer, J., McDonald, R., and Pereira, F. Domain adaptation with structural correspondence learning. In *Conference on Empirical Methods in Natural Language Processing*, pp. 120–128, 2006.
- Chen, C., Fu, Z., Chen, Z., Jin, S., Cheng, Z., Jin, X., and Hua, X.-S. Homm: Higher-order moment matching for unsupervised domain adaptation. In *AAAI*, volume 34, pp. 3422–3429, 2020.
- Chen, M., Weinberger, K. Q., and Blitzer, J. Co-training for domain adaptation. In *NIPS*, pp. 2456–2464, 2011.
- Chen, M., Xu, Z. E., Weinberger, K. Q., and Sha, F. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 2012.
- Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1853–1865, 2016.
- Courty, N., Flamary, R., Habrard, A., and Rakotomamonjy, A. Joint distribution optimal transportation for domain adaptation. In *NIPS*, 2017.
- Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *NIPS*, 2013.
- Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, pp. 2960–2967, 2013.
- Flamary, R. and Courty, N. Pot python optimal transport library, 2017.
- French, G., Mackiewicz, M., and Fisher, M. Self-ensembling for visual domain adaptation. February 2018. ICLR.
- Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016.
- Gautheron, L., Redko, I., and Lartzien, C. Feature selection for unsupervised domain adaptation using optimal transport. In *ECML PKDD*, pp. 759–776. Springer, 2018.
- Gong, M., Zhang, K., Liu, T., Tao, D., Glymour, C., and Schölkopf, B. Domain adaptation with conditional transferable components. In *ICML*, pp. 2839–2848. PMLR, 2016.

- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. 27, 2014.
- Heckman, J. J. Sample selection bias as a specification error. *Econometrica: Journal of the Econometric Society*, pp. 153–161, 1979.
- Japkowicz, N. and Stephen, S. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.
- Jones, K. S. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.
- Kantorovich, L. On the translocation of masses. *Management Science*, 5(1):1–4, 1958.
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. *NIPS*, 30: 3146–3154, 2017.
- Kurmi, V. K., Subramanian, V. K., and Namboodiri, V. P. Domain impression: A source data free domain adaptation method. In *WACV*, pp. 615–625, 2021.
- Lee, D.-H. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2013.
- Liang, J., Hu, D., and Feng, J. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *ICML*, pp. 6028–6039. PMLR, 2020.
- Lin, Y., Lee, Y., and Wahba, G. Support vector machines for classification in nonstandard situations. *Machine learning*, 46(1-3):191–202, 2002.
- Long, M., Wang, J., Ding, G., Sun, J., and Yu, P. S. Transfer feature learning with joint distribution adaptation. In *ICCV*, 2013.
- Long, M., Cao, Y., Wang, J., and Jordan, M. I. Learning transferable features with deep adaptation networks. In *ICML*, 2015.
- Long, M., Zhu, H., Wang, J., and Jordan, M. I. Deep transfer learning with joint adaptation networks. In *ICML*, pp. 2208–2217. PMLR, 2017.
- Long, M., Cao, Z., Wang, J., and Jordan, M. I. Conditional adversarial domain adaptation. In *NeurIPS*, pp. 1640–1650, 2018.
- Monge, G. Mémoire sur la théorie des déblais et des remblais. *Histoire de l’Académie Royale des Sciences de Paris*, 1781.
- Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2009.

- Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2): 199–210, 2010.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Neurips*, volume 32, pp. 8026–8037. Curran Associates, Inc., 2019.
- Perrot, M., Courty, N., Flamary, R., and Habrard, A. Mapping estimation for discrete optimal transport. In *NIPS*, pp. 4204–4212, 2016.
- Peyré, G. and Cuturi, M. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5-6):355–607, 2019.
- Saerens, M., Latinne, P., and Decaestecker, C. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation*, 14(1):21–41, 2002.
- Saito, K., Ushiku, Y., and Harada, T. Asymmetric tri-training for unsupervised domain adaptation. In *ICML*, 2017.
- Saito, K., Watanabe, K., Ushiku, Y., and Harada, T. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, pp. 3723–3732, 2018.
- Satpal, S. and Sarawagi, S. Domain adaptation of conditional probability models via feature subsetting. In *ECML PKDD*, pp. 224–235. Springer, 2007.
- 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.
- Shu, R., Bui, H. H., Narui, H., and Ermon, S. A DIRT-T approach to unsupervised domain adaptation. In *ICLR*. OpenReview.net, 2018.
- Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., and Kawanabe, M. Direct importance estimation with model selection and its application to covariate shift adaptation. In *NIPS*, 2008.
- Sun, B., Feng, J., and Saenko, K. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pp. 153–171. Springer, 2017.
- Torralba, A. and Efros, A. A. Unbiased look at dataset bias. In *CVPR*, pp. 1521–1528. IEEE, 2011.
- Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *CVPR*, pp. 7167–7176, 2017.

- Uguroglu, S. and Carbonell, J. Feature selection for transfer learning. In *ECML PKDD*, pp. 430–442. Springer, 2011.
- Yeh, H.-W., Yang, B., Yuen, P. C., and Harada, T. Sofa: Source-data-free feature alignment for unsupervised domain adaptation. In *WACV*, pp. 474–483, 2021.
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? In *NIPS*, pp. 3320–3328, 2014.
- You, K., Wang, X., Long, M., and Jordan, M. Towards accurate model selection in deep unsupervised domain adaptation. In *ICML*, pp. 7124–7133. PMLR, 2019.
- Zellinger, W., Grubinger, T., Lughofer, E., Natschläger, T., and Saminger-Platz, S. Central moment discrepancy (cmd) for domain-invariant representation learning. 04 2017.
- Zhang, L., Germain, P., Kessaci, Y., and Biernacki, C. Target to source coordinate-wise adaptation of pre-trained models. In *ECML PKDD*, pp. 378–394. Springer International Publishing, 2021.
- Zhu, X. J. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.