PRELIMINARY PREPRINT VERSION: DO NOT CITE
The AAAI Digital Library will contain the published
version some time after the conference.

Interpretable Domain Adaptation for Hidden Subdomain Alignment in the Context of Pre-trained Source Models

Luxin Zhang[1], **Pascal Germain**[3], **Yacine Kessaci**[1], **Christophe Biernacki**[2]
[1] Worldline Labs, France
[2] Inria, Université de Lille, CNRS, France
[3] Université Laval, Canada
luxin.zhang@worldline.com, pascal.germain@ift.ulaval.ca, yacine.kessaci@worldline.com, christophe.biernacki@inria.fr

Abstract

Domain adaptation aims to leverage source domain knowledge to predict target domain labels. Most domain adaptation methods tackle a single-source, single-target scenario, whereas source and target domain data can often be subdivided into data from different distributions in real-life applications (*e.g.*, when the distribution of the collected data changes with time). However, such subdomains are rarely given and should be discovered automatically. To this end, some recent domain adaptation works seek separations of hidden subdomains, w.r.t. a known or fixed number of subdomains. In contrast, this paper introduces a new subdomain combination method that leverages a variable number of subdomains. Precisely, we propose to use an inter-subdomain divergence maximization criterion to exploit hidden subdomains. Besides, our proposition stands in a target-to-source domain adaptation scenario, where one exploits a pre-trained source model as a black box; thus, the proposed method is model-agnostic. By providing interpretability at two complementary levels (transformation and subdomain levels), our method can also be easily interpreted by practitioners with or without machine learning backgrounds. Experimental results over two fraud detection datasets demonstrate the efficiency of our method.

Introduction

In a traditional supervised learning paradigm, one supposes that testing data are from the same distribution as training. However, such an assumption is often violated in real-life applications. For example, to expand a company's business, a pre-trained fraud detection system may be reused to predict fraudsters of a new market where customers have different payment habits. Domain adaptation (DA) methods (Pan and Yang 2009; Torralba and Efros 2011) tackle this problem by mitigating the gaps between the training data, the so-called source domain data, and the testing data, the so-called target domain data. Most DA methods (*e.g.*, Long et al. (2015); Ganin et al. (2016)) map source data into the target domain or transform source and target domain into a latent space. Therefore, they require a training process to estimate a predictive model during the adaptation. However, such a training process could be undesirable, especially when dealing with a mix of predictive models.

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

In many real-life industrial applications, the predictive model is often given as a "black-box" of a mix of models of different types: SVMs, decision trees, and even expert rules, to name a few. Retraining such a model requires tedious hyper-parameter fine-tuning and is sometimes unfeasible due to no longer accessible expertise. Zhang et al. (2021b,a) proposed a *target to source* DA method leveraging one-dimensional optimal transport to address this adaptation problem with a pre-trained source model. The proposed method transforms target data into the source domain; thus, the pre-trained source model can be directly applied to adapted target data. This paper stands in a similar setting where a well-performed source domain predictive model is given and cannot be retrained. Since such a scenario is often used to predict tabular datasets, hereafter, we focus on the adaptation problem of tabular data.

Furthermore, different from works of Zhang et al. (2021b,a) that tackle a *single-source single-target* (*single-domain*) DA setting, this article addresses a more challenging *multi-source multi-target* DA problem. More specifically, we consider both source and target domains can be subdivided into data from different distributions, the so-called subdomains. Nevertheless, subdomain labels are rarely provided; thus one should propose methods to annotate them automatically. As discrepancies between subdomains of the same domain are not as significant as discrepancies between source and target domains, such intra-domain drifts may be omitted and even undiscovered as "hidden" when training a predictive model for a single domain. However, identifying such hidden subdomains contributes to DA by increasing the precision and flexibility of adaptation methods. Some recent works (Xu et al. 2018; Peng et al. 2019) that study *multi-source* or *multi-target* DA problems focus on a scenario where subdomain labels are provided. In contrast, we address a more challenging case where one needs to discover these hidden subdomains. Although Gong, Grauman, and Sha (2013); Mancini et al. (2018) tackle such a hidden subdomain discovering problem, they assume that the number of hidden subdomains is known *a priori*. Relying on the *single-domain* DA function of Zhang et al. (2021b,a), this paper proposes a method reweighting different target domain classifiers adapted from the best combination of subdomains. Therefore, it is not necessary to know the number of subdomains *a priori*. Precisely, we first provide a general separation criterion to exploit hidden subdomains. Then we specialize our proposition to a practical scenario where data drift in each domain is imputed to time. We name our method **H***idden* **S***ubdomain* **A***daptation with* **V***ariable Number of Subdomains* (HSAV).

To summarize, the contributions of this paper are threefold. i) We provide a new *multi-source multi-target* DA paradigm to address a *target to source* DA scenario where a pre-trained source domain predictive model is given and should be preserved; thus, our method is model-agnostic in that sense. This paradigm provides objective functions and efficient optimization methods in both weakly supervised (when few target labels are available) and unsupervised (without any target label) scenarios. ii) We leverage a predictor reweighting method over a variable number of hidden subdomains, where it is not necessary to know the number of hidden subdomains *a priori* to discover them. iii) Combined with the native interpretability of the *single-domain* transformations, we propose in HSAV a sparse weighting factor between target and source subdomains. The proposition enhances the interpretability of our method by giving practitioners, with or without machine learning backgrounds, insights on discovered hidden subdomains. We show the interpretability of our methods in experiments empirically.

Related Works

Single-domain DA. The *single-domain* DA method tackles the case where one source domain with labeled data and one target domain with unlabeled or weakly-labeled data are given. Some classical methods address this problem by minimizing measures like Kullback-Leibler divergence (Shimodaira 2000; Sugiyama et al. 2008) or Maximum Mean Discrepancy (MMD) (Pan et al. 2010; Baktashmotlagh et al. 2013; Long et al. 2013) between source and target distributions. Others focus on the alignment of the source and target domains correlations (Sun, Feng, and Saenko 2017) or principal axes (Fernando et al. 2013). Some recent researches (Perrot et al. 2016; Courty et al. 2016, 2017) transform source data to target domains by seeking the optimal transport plan and minimizing the Wasserstein distance. Deep learning approaches are also shown to be efficient for DA tasks (Yosinski et al. 2014), especially when dealing with image datasets. Long et al. (2015, 2017) enhance such transferability by plugging into neural networks an adaptation layer to minimize MMD. Ganin et al. (2016), Tzeng et al. (2017), Long et al. (2018), and Saito et al. (2018) leverage advances in adversarial learning to generate domain invariant features. Several recent works (Liang, Hu, and Feng 2020; Kurmi, Subramanian, and Namboodiri 2021; Yeh et al. 2021) leveraging a pre-trained source model for DAs stand in a setting where source domain data are not available. However, such methods are suboptimal when hidden subdomains exist due to undiscovered intra-domain drifts.

Multi-subdomain DA. To address the *single-domain* DA scenario, Liu, Shao, and Fu (2016) and Peng et al. (2019) leverage a moment matching, Zhao et al. (2018) adopt adversarial networks, and Li et al. (2018) use a similarity graph. Recent works also propose using knowledge distilling (Zhao et al. 2020) or aligning outputs of classifiers seamlessly (Venkat et al. 2021). Similar to us, the works of Duan et al. (2009), Mansour, Mohri, and Rostamizadeh (2009), Xu et al. (2018), and Hoffman, Mohri, and Zhang (2018) adopt subdomain reweighing methods for target label predictions. However, in our proposition, instead of combining subdomains, HSAV reweights separations of subdomains w.r.t. different subdomain numbers. Besides, it can address different types of pre-trained source models and is not limited to neural networks. Moreover, all of the aforementioned methods that tackle the *multi-subdomain* DA problem suppose that subdomain labels are given.

When dealing with hidden subdomains without subdomain labels, Gong, Grauman, and Sha (2013) propose to discover them by maximizing domain discrepancies, and the number of subdomains is chosen to be the one that maximizes the source domain prediction performances. Hoffman et al. (2012) leverage a clustering method of the input space to get hidden subdomains. Xu et al. (2014) and Li et al. (2017) build subdomains by including only one source domain positive example, and all negative examples. However, such a method is not scalable to massive datasets. Recent works of Mancini et al. (2018, 2019) discover hidden subdomains relying on a neural network of subdomain classifier, while the number of subdomains should be known *a priori*. In our case, instead of considering one fixed number of subdomains, we leverage a variable number of subdomains, that we aggregate into an ensemble.

Supervised Hidden Subdomain Adaptation

We denote the input (resp. output) space of predictive models as $\mathcal{X}$ (resp. $\mathcal{Y}$). As we focus on a binary classification problem in this paper, $\mathcal{Y}=\{0,1\}$, and the pre-trained source domain predictor $h_s(x^s) : \mathcal{X}\to[0,1]$ gives the probability that one example x^s is classified as 1. In a *target to source* DA setting, $h_s(x^s)$ is given as a black-box classifier of a wide variety of types (*e.g.*, neural networks, decision trees). Previous works addressing this case focus on a *single-domain* DA problem (Zhang et al. 2021b,a). In this paper, we generalize the method to a *multi-subdomain* setting for extending the flexibility of such DA approaches while preserving their appealing properties. We denote by $\mathcal{X}_{\text{sub}}$ the feature space that encodes hidden subdomains. Note that features in $\mathcal{X}$ and $\mathcal{X}_{\text{sub}}$ can be different. $\mathcal{X}$ stands for the discriminative attributes in predicting class labels, whereas $\mathcal{X}_{\text{sub}}$ contains attributes that help discover hidden subdomains and may not be discriminative in classification; thus, $\mathcal{X}_{\text{sub}}$ can contain attributes that are not in $\mathcal{X}$.

Let X^t and X^s be respectively the target and source domain input variables over the support $\mathcal{X}$, and $P(X^t)$ and $P(X^s)$ represent their distributions. Analogously, X_i^t and X_j^s are the marginal variables of corresponding subdomains, and $P(X_i^t)$ and $P(X_j^s)$ are subdomain distributions. Thus,

$$P(X^t)=\sum_{i=1}^{k_t}\pi_i^t P(X_i^t)\,, \text{ and } P(X^s)=\sum_{j=1}^{k_s}\pi_j^s P(X_j^s)\,,$$

where π_i^t and π_j^s are proportions of subdomains, and $k_s \in \{1, \ldots, k_s^{\text{sup}}\}$ and $k_t \in \{1, \ldots, k_t^{\text{sup}}\}$ refer to the number of subdomains. $k_t^{\text{sup}} \in \mathbb{N}^*$ and $k_s^{\text{sup}} \in \mathbb{N}^*$ stand respectively for the maximum number of subdomains that we consider. Let $\mathbb{X}^t = \{x_l^t\}_{l=1}^{n_t}$ (resp. $\mathbb{X}^s = \{x_m^s\}_{m=1}^{n_s}$) be n^t (resp. n^s) target (resp. source) domain examples drawn from $P(X^t)$ (resp. $P(X^s)$). Analogously, we also define $\mathbb{X}_i^t = \{x_l^t\}_{l=1}^{n_i^t}$, $\mathbb{X}_j^s = \{x_m^s\}_{m=1}^{n_j^s}$ sets of examples of target and source subdomains respectively drawn from $P(X_i^t)$ and $P(X_j^s)$. For compactness, we denote $\mathbb{X} = \{\mathbb{X}_1^t, \ldots, \mathbb{X}_{k_t}^t, \mathbb{X}_1^s, \ldots, \mathbb{X}_{k_s}^s\}$ the set of all target and source subdomains.

Moreover, we assume that there exists a mapping matrix $\boldsymbol{S} \in \{0, 1\}^{k_t \times k_s}$ that relates hidden target subdomains to the source ones. We denote $S_{i,j}$ the scalar located at the i-th row and the j-th column of $\boldsymbol{S}$; $\boldsymbol{S}_{i,\cdot}$ and $\boldsymbol{S}_{\cdot,j}$ represent the row and column vectors. $S_{i,j}$ takes the value 1 if the target subdomain X_i^t is mapped to the source subdomain X_j^s, or the value 0 otherwise. Furthermore, to enhance the interpretability, we encourage $\boldsymbol{S}$ to be sparse. Typically, we want one target hidden subdomain maps to only one source hidden subdomain, that is, $\forall i \in \{1, \ldots, k_t\}\, , \sum_{j=1}^{k_s} S_{i,j} = 1\, .$

In the following, we first provide details of our adaptation methods by starting with a known number of subdomains, and then we generalize our method to handle a variable number of subdomains.

Hidden Subdomain Adaptation with a Known Number of Subdomains. Let first assume that we face a DA problem from which we know *a priori* the underlying number of target and source subdomains k_t and k_s, and let $\boldsymbol{K} = (k_t, k_s)$. Given $x^t \in \mathbb{X}^t$, we formalize the target domain classifier as

$$h_t(x^t; \boldsymbol{K}, \mathbb{X}, \boldsymbol{S}) = \sum_{i=1}^{k_t} M_i(x^t; \mathbb{X}) h_t^i(x^t; \boldsymbol{K}, \mathbb{X}, \boldsymbol{S})\, , \quad (1)$$

where h_t^i is a predictor of the i-th target subdomain, and $M_i(x^t; \mathbb{X})$ is the probability that a target example x^t belongs to this subdomain. More precisely, we have

$$M_i(x^t; \mathbb{X}) = \frac{\pi_i^t P(X_i^t = x^t)}{\sum_{k=1}^{k_t} \pi_k^t P(X_k^t = x^t)}\, .$$

Of note, as our experiments involve both categorical and numerical features, the i-th subdomain density $P(X_i^t = x^t)$ is thus computed by a product of densities of each dimension of x^t. Furthermore, we formalize the classifier of the i-th target subdomain as

$$h_t^i(x^t; \boldsymbol{K}, \mathbb{X}, \boldsymbol{S}) = \sum_{j=1}^{k_s} S_{i,j} h_s \circ \mathcal{G}_{i,j}(x^t; \mathbb{X})\, , \quad (2)$$

where $\mathcal{G}_{i,j}(\cdot; \mathbb{X})$ is a *single-domain* DA function that transforms data from the i-th target subdomain $\mathbb{X}_i^t$ to the j-th source subdomain $\mathbb{X}_j^s$. Namely, we leverage the work of Zhang et al. (2021a) by setting $\mathcal{G}_{i,j}(\cdot; \mathbb{X})$ to be *coordinate-wise* DA functions, as their method is model-agnostic and easy to interpret.[1] With known subdomains numbers $\boldsymbol{K}$, the hidden subdomain adaptation consists of estimating the optimal subdomain separations $\mathbb{X}$ and their relation matrix $\boldsymbol{S}$.

Estimation of $\mathbb{X}$. Logically, separations of subdomains are significant if inter-subdomain discrepancies are large. As if subdomains were similar, they could be adapted using the same transformation, and there would be no need to distinguish them. Moreover, in a predictor weighting formalization as Equations (1) and (2), one benefits from a diversity between weighted elements. Here, such diversity is inherited from the differences between subdomains, as they likely spawn diverse transformations $\mathcal{G}_{i,j}(\cdot; \mathbb{X})$. Following the same convention of Gong, Grauman, and Sha (2013) and Hoffman et al. (2012), we search the optimal $\mathbb{X}$ by maximizing inter-subdomain discrepancies. That is,

$$\mathbb{X}^* = \underset{\mathbb{X}}{\operatorname{argmax}} \left[\sum_{i \neq j}^{k_t} D(\mathbb{X}_i^t, \mathbb{X}_j^t) + \sum_{i \neq j}^{k_s} D(\mathbb{X}_i^s, \mathbb{X}_j^s) \right], \quad (3)$$

where $D(\cdot, \cdot)$ is a domain discrepancy measure. In our case, as we focus on adapting tabular data where the input space contains categorical and numerical attributes, $D(\cdot, \cdot)$ is chosen to be the sum of one-dimensional Wasserstein distances over each feature.

Estimation of $\boldsymbol{S}$ in a weakly supervised setting. In this section, we stand in a case where a few target domain data are annotated with true labels. Such a small set of n_w examples is denoted $\mathbb{D} = \{(x_l^t, y_l^t)\}_{l=1}^{n_w}$. Note that the next section provides an unsupervised version of our method to address the more challenging scenario without target data.

In a weakly supervised scenario, once $\mathbb{X}$ is determined, one can minimize the prediction error over the few labeled target domain points to estimate $\boldsymbol{S}$. That is,

$$\boldsymbol{S}^* = \operatorname{argmin}_{\boldsymbol{S}} \tfrac{1}{n_w} \textstyle\sum_{(x^t, y^t) \in \mathbb{D}} l(h_t(x^t; \boldsymbol{K}, \mathbb{X}^*, \boldsymbol{S}), y^t)\, , \quad (4)$$

where $l : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$ is a loss function (*e.g.*, binary cross-entropy loss). The optimization details of this problem are provided in the implementation section. Basically, we leverage a Softmax function to approximate $\boldsymbol{S}^*$.

Hidden Subdomain Adaptation with a Variable Number of Subdomains. In many real-life scenarios, the number of hidden subdomains is unknown. In such cases, a natural choice is to estimate the optimal couple $\boldsymbol{K}$ using weakly labeled target examples, that is,

$$\boldsymbol{K}^* = \operatorname{argmin}_{\boldsymbol{K}} \tfrac{1}{n_w} \textstyle\sum_{(x^t, y^t) \in \mathbb{D}} l(h_t^\dagger(x^t; \boldsymbol{K}), y^t)\, ,$$

where $h_t^\dagger(x^t; \boldsymbol{K}) = h_t(x^t; \boldsymbol{K}, \mathbb{X}^*, \boldsymbol{S}^*)$ is the estimated optimal target classifier of the previous section with $\boldsymbol{K}$ subdomains. For a given $\boldsymbol{K}$, $h_t^\dagger(x^t; \boldsymbol{K})$ is the estimated optimal target domain classifier of the previous section. However, when using a Bagging strategy, we empirically observe that one can hardly find a single $\boldsymbol{K}^*$ that is significantly better than others. Indeed, the scarcity of $\mathbb{D}$ leads to high variability in Bagging predictive performances.

Therefore, instead of using $h_t^\dagger(x^t; \boldsymbol{K}^*)$ with the optimal estimated $\boldsymbol{K}^*$ as the target domain classifier, we propose aggregate multiple target predictors $h_t^\dagger(x^t; \boldsymbol{K})$. Each

[1]See a summary of this adaptation function in the supplementary material at https://github.com/marrvolo/HSAV.

$h_t^\dagger(x^t;\boldsymbol{K})$ is obtained for different values of $\boldsymbol{K}$. The weight associated with each possible subdomain number is handled by the matrix $\boldsymbol{A} \in \mathbb{R}^{k_t^{\text{sup}} \times k_s^{\text{sup}}}$, such that the target domain predictive model becomes

$$h_t^*(x^t;\boldsymbol{A}) = \sum_{k_t=1}^{k_t^{\text{sup}}} \sum_{k_s=1}^{k_s^{\text{sup}}} \sigma_{k_t,k_s}(\boldsymbol{A}) h_t^\dagger(x^t;(k_t,k_s)) ,$$

where σ_{k_t,k_s} is a Softmax function that encourages a sparsity of the weighting factor:

$$\sigma_{k_t,k_s}(\boldsymbol{A}) = \frac{\exp(A_{k_t,k_s})}{\sum_{u=1}^{k_t^{\text{sup}}} \sum_{v=1}^{k_s^{\text{sup}}} \exp(A_{u,v})} .$$

Consequently, the objective function becomes

$$\boldsymbol{A}^* = \operatorname{argmin}_{\boldsymbol{A}} \tfrac{1}{n_w} \textstyle\sum_{(x^t,y^t)\in\mathbb{D}} l(h_t^*(x^t;\boldsymbol{A}), y^t) , \qquad (5)$$

and the corresponding target domain predictor is $h_t^*(\cdot;\boldsymbol{A}^*)$.

Unsupervised Hidden Subdomain Adaptation

We now address a more challenging unsupervised DA scenario, such that one can still get separations of subdomains $\mathbb{X}^*$ using Equation (3), whereas it is no longer possible to estimate $\boldsymbol{S}^*$ relying on Equation (4), nor get the target domain predictive model $h_t^*(\cdot;\boldsymbol{A}^*)$ using Equation (5), since y^t is not accessible. Alternately, we propose to leverage a necessary condition of the optimal DA to estimate $\boldsymbol{S}$ and $\boldsymbol{A}$.

In real-life applications, the given pre-trained source domain predictive model is often well trained to be the optimal one in source domains, that is, $h_s(x^s) = P(Y^s{=}1|X^s{=}x^s)$. Under such a setting, when $\boldsymbol{K}$ is known, a necessary condition for $\mathbb{X}^*$ and $\boldsymbol{S}^*$ to be the optimal ones is

$$P(h_t(X^t;\boldsymbol{K},\mathbb{X}^*,\boldsymbol{S}^*)) = P(h_s(X^s)) . \qquad (6)$$

When we have a variable number of subdomains, a necessary condition of the optimal $\boldsymbol{A}^*$ is

$$P(h_t^*(X^t;\boldsymbol{A}^*)) = P(h_s(X^s)) . \qquad (7)$$

Basically, domain adaptation aims to align joint distributions of target and source domains; thus the alignment of the output distributions of classifiers is a necessary condition.[2] In practice, $\mathbb{X}^*$ is fixed using Equation (3), and one can search for $\boldsymbol{S}^*$ and $\boldsymbol{A}^*$ relying on such conditions.

Estimation of $\boldsymbol{S}$. Inspired by Equation (6), given $\boldsymbol{K}{\in}\{(k_t,k_s)|k_t{\in}\{1,\ldots,k_t^{\text{sup}}\}, k_s{\in}\{1,\ldots,k_s^{\text{sup}}\}\}$, the unsupervised optimization problem of $\boldsymbol{S}$ is formulated as

$$\boldsymbol{S}^* = \operatorname{argmin}_{\boldsymbol{S}} W(h_t(X^t;\boldsymbol{K},\mathbb{X}^*,\boldsymbol{S}), h_s(X^s)) , \qquad (8)$$

where $W(\cdot,\cdot)$ is the one-dimensional Wasserstein distance over the distribution of positive outputs. Empirically, $W(\cdot,\cdot)$ is given by

$$W(h_t(X^t;\boldsymbol{K},\mathbb{X}^*,\boldsymbol{S}), h_s(X^s)) =$$
$$\textstyle\sum_{x^t\in\mathbb{X}^t} \left(h_t(x^t;\boldsymbol{K},\mathbb{X}^*,\boldsymbol{S}) - F_s^{-1}(h_t(x^t;\boldsymbol{K},\mathbb{X}^*,\boldsymbol{S}))\right)^2 ,$$

where F^s is the cumulative distribution function of h_s.

[2]See details at https://github.com/marrvolo/HSAV.

Estimation of $\boldsymbol{A}$. For a variable number of subdomains, inspired by Equation (7), we propose the following unsupervised objective function to mimic Equation (5):

$$\boldsymbol{A}^* = \operatorname{argmin}_{\boldsymbol{A}} W(h_t^*(X^t;\boldsymbol{A}), h_s(X^s)) . \qquad (9)$$

In the implementation section, we give optimization details of our proposed objective functions.

Specialization to Temporal Drift. In this paper, we focus on a practical case where the feature $\mathcal{X}_{\text{sub}}$ that encodes hidden subdomains is one temporal dimension (the time). Such a scenario is very common in real-life applications where data arrive as time goes on, and there is a drift between collected data. For example, in a payment fraud detection system, payment habits are different due to the change of seasonality. Moreover, time is a one-dimensional feature that can be efficiently subdivided. Consequently, subdomains $\mathbb{X}_i^t$ and $\mathbb{X}_j^s$ are respectively parameterized by separations of time $\{t_1^t,\ldots,t_{k_t-1}^t\}$ and $\{t_1^s,\ldots,t_{k_s-1}^s\}$ in such a setting. We have

$$\forall x^t \in \mathbb{X}_i^t , t_{i-1}^t < T(x^t) \le t_i^t , \text{and}$$
$$\forall x^s \in \mathbb{X}_j^s , t_{j-1}^s < T(x^s) \le t_j^s ,$$

where $T(\cdot) : \mathcal{X} \to \mathcal{X}_{\text{sub}}$ gives the hidden subdomain feature (the time in our case). As we separate subdomains following the axis of time, instead of computing inter-subdomain discrepancies between every pair of subdomains, we take into account only discrepancies between successive subdomains. Equation (3) is redefined as

$$\mathbb{X}^* {=} \operatorname*{argmax}_{\mathbb{X}} \left[\sum_{i=1}^{k_t-1} D(\mathbb{X}_i^t,\mathbb{X}_{i+1}^t) {+} \sum_{j=1}^{k_s-1} D(\mathbb{X}_j^s,\mathbb{X}_{j+1}^s) \right] . \qquad (10)$$

Recent DA works (Bobu et al. 2018; Wang, He, and Katabi 2020) addressing the temporal drift do not apply to our case, as we adapt from target subdomains to source subdomains, whereas they align subdomains data in the same domain.

Implementation

For both weakly supervised and unsupervised cases, the proposed HSAV consists of 3 steps. i) We estimate separations of subdomains $\mathbb{X}$ w.r.t. different numbers of subdomains. ii) We estimate the corresponding sparse mapping factor $\boldsymbol{S}$. iii) We combine predictions of variable numbers of subdomains relying on $\boldsymbol{A}$.

Optimization over $\mathbb{X}$. Since Equation (10) is not differentiable due to the existence of categorical features, we rely on the Nelder-Mead method (Nelder and Mead 1965) to solve this objective function. Note that, since the two sums of Equation (10) are independent, one can solve both parts individually.

The maximum number of subdomains is determined by gradually increasing the number of subdomains and solving Equation (10). Namely, we start from $k_t = 2$ (resp. $k_s = 2$) and compute the subdomain separations. We increase k_t (resp. k_s) by 1 if $\forall i \in \{1,\ldots,k_t\}, n_i^t > n_m$ (resp. $\forall j \in \{1,\ldots,k_s\}, n_j^s > n_m$), where n_m is the minimum number of examples that we expect to have in each subdomain. Otherwise, we stop the process and take the current value of k_t (resp. k_s) as the maximum number of target (resp. source) subdomains k_t^{sup} (resp. k_s^{sup}).

Optimization over $\boldsymbol{S}$. Since the discrete optimization problems (Equations 4 and 8) are computationally expensive, instead of directly searching $\boldsymbol{S}$ in the discrete space $\{0,1\}^{k_t \times k_s}$, we relax the constraints over $\boldsymbol{S}$ relying on a Softmax function by row. Specifically, we set

$$\omega_{i,j}(\tilde{\boldsymbol{S}}) = \frac{\exp(\tilde{S}_{i,j})}{\sum_{k=1}^{k_s}\exp(\tilde{S}_{i,k})} ,$$

with $\tilde{\boldsymbol{S}} \in \mathbb{R}^{k_t \times k_s}$, and $\omega(\tilde{\boldsymbol{S}})$ is a matrix where the i-th row the j-th column is referred to as $\omega_{i,j}(\tilde{\boldsymbol{S}})$.

Therefore, in a weakly supervised scenario, we can optimize $\tilde{\boldsymbol{S}}$ in place of $\boldsymbol{S}$, and Equation (4) becomes

$$\tilde{\boldsymbol{S}}^* = \operatorname{argmin}_{\tilde{\boldsymbol{S}}} \frac{1}{n_w} \sum_{(x^t,y^t)\in\mathbb{D}} l(h_t(x^t; \boldsymbol{K}, \mathbb{X}^*, \omega(\tilde{\boldsymbol{S}})), y^t) ,$$

which can be solved by classical optimization methods (*e.g.*, gradient descent). Analogously, in an unsupervised scenario, Equation (8) becomes

$$\tilde{\boldsymbol{S}}^* = \operatorname{argmin}_{\tilde{\boldsymbol{S}}} W(h_t(X^t; \boldsymbol{K}, \mathbb{X}^*, \omega(\tilde{\boldsymbol{S}})), h_s(X^s)) .$$

As F_s^{-1} is not differentiable, we solve it using an iterative method relying on the gradient descent. At each step (denoted by p) of iterations, the gradient is computed by

$$\nabla g_s = \frac{\partial \sum_{x^t \in \mathbb{X}^t} \left(h_t(x^t; \boldsymbol{K}, \mathbb{X}^*, \omega(\tilde{\boldsymbol{S}})) - H(x^t)\right)^2}{\partial \tilde{\boldsymbol{S}}} , \tag{11}$$

where $H(x^t) = F_s^{-1}(h_t(x^t; \boldsymbol{K}, \mathbb{X}^*, \omega(\tilde{\boldsymbol{S}}^{p-1})))$. Note that we compute $H(x)$ using $\tilde{\boldsymbol{S}}^{p-1}$ of the previous iteration, which is considered as a constant with respect to $\tilde{\boldsymbol{S}}$ and does not contribute to the gradient of $\tilde{\boldsymbol{S}}^p$. Then we set $\tilde{\boldsymbol{S}}^p = \tilde{\boldsymbol{S}}^{p-1} - c_1 * \nabla g_s$ with c_1 as a learning rate and continue the process until the stop criterion is met. We set all $\tilde{S}_{i,j} = 0$ at initialization for both supervised and unsupervised cases; thus the initial factor $\tilde{\boldsymbol{S}}$ between each pair of target-source subdomain is uniform.

To further reduce the computational cost of the proposed method, instead of using one $\mathcal{G}_{i,j}(\cdot; \mathbb{X})$ between each pair of target-source subdomains, we compute a global transformation function $\mathcal{G}(\cdot)$ that adapts X^t to X^s and let $\mathcal{G}_{i,j}(\cdot; \mathbb{X}) = \mathcal{G}(\cdot)$ during optimizations. The final discrete solution of $\boldsymbol{S}$ is

$$S_{i,j}^* = \begin{cases} 1 & \text{if } \tilde{S}_{i,j}^* = \max_k \tilde{S}_{i,k}^* , \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Once the discrete factor $\boldsymbol{S}$ between target and source subdomains is determined, we re-estimate transformation functions $\mathcal{G}_{i,j}(\cdot; \mathbb{X})$ between the i-th subdomain of target and the j-th subdomain of source for all $S_{i,j}^* = 1$.

Optimization over $\boldsymbol{A}$. Following a similar approach as the optimization of $\tilde{\boldsymbol{S}}$, in a supervised setting, we solve Equation (5) using a gradient descent method. Supposing that $\min(k_t^{\text{sup}}, k_s^{\text{sup}}) > 1$, at initialization, we set

$$A_{k_t,k_s} = \begin{cases} \log(3(k_t^{\text{sup}} \times k_s^{\text{sup}} - 1)) & \text{if } k_t{=}k_s{=}1 , \\ 0 & \text{otherwise,} \end{cases}$$

which makes the weighting factor of target classifiers of different numbers of subdomains to be

$$\sigma_{k_t,k_s}(\boldsymbol{A}) = \begin{cases} 3/4 & \text{if } k_t{=}k_s{=}1 , \\ 1/(4(k_t^{\text{sup}} \times k_s^{\text{sup}} - 1)) & \text{otherwise.} \end{cases}$$

We set $\sigma_{k_t,k_s}(\boldsymbol{A}){=}3/4$ for $k_t{=}k_s{=}1$ to privilege this choice if subdomains cannot significantly improve prediction performances. Besides, the value of the gradient at $\sigma_{k_t,k_s}(\boldsymbol{A}) = 3/4$ is not too small to conduct effective training.

In an unsupervised setting, we solve Equation (9) by the same approach as Equation (11). Namely, the gradient at the p-th step of iterations is computed by

$$\nabla g_u = \frac{\partial \sum_{x^t \in \mathbb{X}^t} \left(h_t^*(x^t; \boldsymbol{A}) - Q(x^t)\right)^2}{\partial \boldsymbol{A}} ,$$

where $Q(x^t){=}F_s^{-1}(h_t^*(x^t; \boldsymbol{A}^{p-1}))$. Then we set $\boldsymbol{A}^p = \boldsymbol{A}^{p-1} - c_2 \nabla g_u$ with c_2 as a learning rate until the stop criteria is met.

Furthermore, we adopt a Bagging method (Breiman 1996) to enhance the robustness of estimated parameters. Precisely, in both supervised and unsupervised scenarios, $\tilde{\boldsymbol{S}}$ and $\boldsymbol{A}$ are estimated over 10 Bagging datasets, and the average value is used as the final estimation results. Then $\boldsymbol{S}$ is obtained relying on Equation (12) over the average of $\tilde{\boldsymbol{S}}$.

Complexity Analysis. Note that the number of repetitions of the Nelder-Mead search to estimate $\mathbb{X}$ increases linearly as the maximum number of subdomains increases. The factors $\boldsymbol{S}$ and $\boldsymbol{A}$ can be estimated efficiently, relying on the gradient descent or its variant. However, the number of estimated adaptation functions $\mathcal{G}_{i,j}(\cdot; \mathbb{X})$ is of the order of $k_s^{\text{sup}}(k_t^{\text{sup}})^2$, which seems not scalable when the expected number of subdomains is large. Nonetheless, k_s^{sup} and k_t^{sup} are generally small in a *target to source* DA setting.

Experiment

We tackle a *target to source* DA scenario with different types of given pre-trained source models. We evaluate HSAV over two fraud detection datasets. Note that these datasets are *tabular* by nature. Our method is suited for these problems, whereas many recent DA methods (involving neural networks) are designed for image data.[3]

Kaggle Fraud Detection Dataset.[4] The dataset contains online payment transactions issued from mobile devices and desktop devices. The objective of this challenge is to predict whether a transaction is fraudulent or not. The input space of the raw dataset is of dimension greater than 400, while most dimensions contain missing values. We discard features with more than 1% of missing values and are not discriminative when predicting fraudsters. Furthermore, examples with any missing values are also removed from the dataset. After the preprocessing, we have around 150,000 examples with 43 numerical and 8 categorical features, and the proportion of fraudsters is around 7% on average. We consider one source

[3]See more details and experiments at https://github.com/marrvolo/HSAV.

[4]www.kaggle.com/c/ieee-fraud-detection

Method	D-1 to M	D-2 to M	D-3 to M	AVG
NN HSAV	5.26	**3.24**	**5.51**	**4.67**
MultiDA	4.59	-7.29	-4.64	-2.45
DCTN	-16.87	-13.40	-8.59	-12.95
NN SCDA	3.23	2.88	5.41	3.84
DAN	**12.31**	-3.20	1.41	3.51
DANN	3.47	-2.90	-4.21	-1.21
MCD	-11.47	-6.29	-6.81	-8.19
LGB HSAV	**32.87**	**7.41**	**14.43**	**18.24**
LGB SCDA	32.68	7.14	14.31	18.04
LGB Baseline	26.43	4.36	7.60	12.80

(a) Kaggle fraud detection dataset in an unsupervised setting.

Method	G-1 to B	G-2 to B	G-3 to B	AVG
NN HSAV	**8.77**	**12.93**	7.38	**9.70**
MultiDA	-7.07	-6.68	8.26	-1.83
DCTN	2.14	-0.15	1.98	1.32
NN SCDA	8.02	11.72	5.95	8.56
DAN	7.31	5.47	**10.01**	7.60
DANN	4.38	6.43	5.28	5.37
MCD	6.46	1.41	6.84	4.91
LGB HSAV	5.60	**6.28**	**9.97**	**7.28**
LGB SCDA	1.83	2.77	7.11	3.90
LGB Baseline	**9.91**	3.59	-1.94	3.85

(b) Real fraud detection dataset in an unsupervised setting.

Method	D-1 to M	D-2 to M	D-3 to M	AVG
NN HSAV	12.61	**5.64**	4.87	**7.70**
MultiDA	-5.71	-2.87	-0.25	-2.94
DCTN	-13.58	-11.77	-5.00	-10.12
NN SCDA	1.30	2.98	3.72	2.66
DAN	**15.40**	-0.57	1.53	5.45
DANN	5.77	-0.87	-2.40	0.83
MCD	8.96	-2.48	-2.89	1.20
FineTune	4.23	2.12	**4.95**	3.76
LGB HSAV	**29.52**	**8.49**	**14.51**	**17.51**
LGB SCDA	27.85	7.18	13.66	16.23
LGB Baseline	26.43	4.36	7.60	12.80

(c) Kaggle fraud detection dataset in a weakly supervised setting.

Method	D-1 to M	D-2 to M	D-3 to M	AVG
NN HSAV	**12.57**	**14.30**	11.79	**12.88**
MultiDA	0.90	-3.85	**19.18**	5.41
DCTN	8.30	10.41	19.17	12.63
NN SCDA	11.75	8.89	13.62	11.42
DAN	9.40	11.85	8.85	10.03
DANN	9.56	10.27	10.46	10.10
MCD	1.64	1.77	12.71	5.38
FineTune	8.04	10.91	5.32	8.09
LGB HSAV	**23.43**	**19.52**	**17.39**	**20.11**
LGB SCDA	22.65	17.84	15.21	18.56
LGB Baseline	9.91	3.59	-1.94	3.85

(d) Real fraud detection dataset in a weakly supervised setting.

Table 1: Prediction performances in PR_AUC of multi-subdomain adaptation (above dashed lines) and single-source single-target adaptation (below dashed lines). All performances are compared to NN baseline models without any adaptation, and the percentage of improvements is reported. Methods above solid lines are NN based, and that below solid lines are LGB based.

domain (denoted by M) that contains all data from mobile devices and three target domains (denoted by D-i) of desktop devices over 3 different periods. Hidden subdomains are further discovered automatically by our proposed method.

Real Fraud Detection Dataset. This private dataset (provided by an IT company) contains real anonymous clients' card transactions from July 2018 to September 2018 of two geographical domains: Belgium and Germany. The input space has 23 numerical features and 7 categorical ones. The Belgian dataset has over 30 million examples with a fraud rate of 0.3%, and the German dataset has around 15 million examples with 0.5% of frauds. We consider the Belgian dataset the source domain (denoted by B) and let each month of German data be one target domain (denoted by G-i).

Other General Setup Details. We pre-train two source domain predictive models: a neural network (NN) and a gradient boosting decision tree implemented in LightGBM (LGB) Python package. The models are estimated using source domain data with 10 different random states. The one that achieves the best prediction performance over source domain testing data is further used as the pre-trained model during our DA process. We compare our proposition with *single-domain* DA methods: DAN (Long et al. 2015), DANN (Ganin et al. 2016), MCD (Saito et al. 2018), and SCDA (Zhang et al. 2021a), and *multi-subdomain* DA methods: DCTN (Xu et al. 2018) and MultiDA (Mancini et al. 2018, 2019). All these adaptation methods are estimated under different hyper-parameters, and the one that achieves the best performance over source domain testing data is chosen. For compared *multi-subdomain* DA methods, we fix the number of hidden subdomains to be 2 in the source domain and 1 in target domains, as target domains contain a shorter period than the source one. At the preprocessing stage of all adaptation methods, we adopt the strategy of Lin, Lee, and Wahba (2002) to adjust the proportion of fraud in the source and target domains to be the same. In a weakly supervised case, we also train a FineTune model by fine-tuning the last layer of NN pre-trained models using weakly labeled target data.

Adaptation Performance. As datasets are highly imbalanced, we evaluate predictive models based on the area under the precision-recall curve (PR_AUC). For the sake of comparison, we set the NN model as a reference and report the percentage of PR_AUC improvements compared to NN. In a weakly supervised setting, we annotate respectively 200 and 5000 labeled examples of target domains of the Kaggle and real fraud detection datasets. For all other compared methods, we extend them to a weakly supervised case by us-

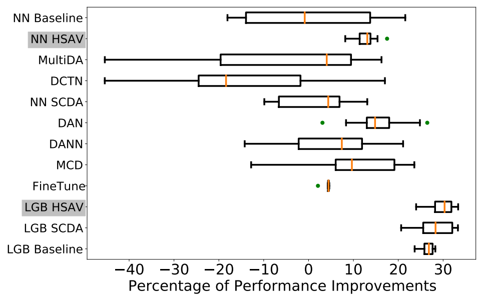
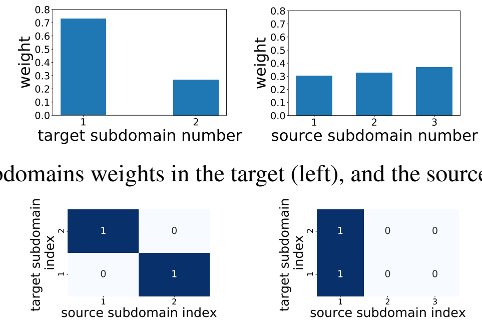


Figure 1: Variances of improvements in PR_AUC of the Kaggle adaptation task D-1 to M in a weakly supervised setting with different random states.



(a) Subdomains weights in the target (left), and the source (right).

(b) Sparse matrix $\boldsymbol{S}$, k_t=k_s=2 (left), and k_t=2 , k_s=3 (right).

Figure 2: Interpretability study on the Kaggle task D-2 to M.

ing labeled target data for hyper-parameter selection. In the unsupervised setting, no labeled target data are used.

Kaggle Adaptation Task. Table 1a reports performances of HSAV based on NN pre-trained model (NN HSAV) and LGB pre-trained model (LGB HSAV) in an unsupervised setting. Regarding NN models, NN HSAV outperforms all other adaptation methods on average. Nevertheless, other subdomain adaptation methods with a fixed number of subdomains (MultiDA, DCTN) have an effect of negative transfer over this dataset. Although DAN significantly improves the baseline of NN models of the adaptation task D-1 to M, the improvement is less significant compared to LGB models. The LGB baseline model without adaptation method exceeds all NN models. Indeed, the LGB model is efficient in dealing with tabular data with categorical variables. Moreover, LGB HSAV further improves the LGB baseline and is also better than the *single-domain* DA method LGB SCDA. Table 1c reports the results in a weakly supervised case. Relying on labeled target data, all adaptation methods based on NN models improve their performances compared to the unsupervised case. However, LGB HSAV and LGB SCDA are no better than their unsupervised version, mainly in the adaptation task D-1 to M. We explain this result by the instability related to the scarcity of weakly labeled data. To better understand the variance of all adaptation methods, we illustrate the adaptation task D-1 to M using a boxplot (Figure 1). Note that, in this task, our method is much more stable than most of the other adaptation methods.

Real Data Adaptation Task. A similar conclusion can be drawn from Tables 1b and 1d where we evaluate adaptation methods over the real fraud detection dataset. On average, our propositions (NN HSAV and LGB HSAV) outperform other adaptation methods. In contrast to the Kaggle adaptation tasks, in the unsupervised setting, LGB HSAV performs no better than NN HSAV. This may be related to the low performance of LGB SCDA compared to NN SCDA, as LGB HSAV combines LGB SCDA as elementary adaptation methods. Unlike the weakly supervised case on the Kaggle datasets, labeled target data always increase the performances of all adaptation methods in this case, especially for the *multi-subdomain* DA method DCTN. The weakly supervised DCTN achieves the second-best performance among all NN models. Furthermore, LGB HSAV also improves NN HSAV significantly in the weakly supervised case. Note that, for both Kaggle and real data adaptation tasks, HSAV succeeds in leveraging multi hidden subdomains information to improve DAs. However, other *multi-subdomain* DA methods (MultiDA, DCTN) suffer in most adaptation tasks. Indeed, such methods generally have high variances and are very sensitive to hyper-parameters and even random states.

Interpretability study. The following shows the interpretability of our method by illustrating the adaptation results of the Kaggle adaptation task D-2 to M. Parameter $\sigma(\boldsymbol{A})$ and $\boldsymbol{S}$ values are displayed in Figure 2. More precisely, Figure 2a is obtained by summing $\sigma(\boldsymbol{A})$ by rows and columns. It shows the weights of each number of subdomains in source and target domains. Specifically, in the target domain, we give more weights to the case where k_t=1 (no subdomain). As for the number of subdomains in the source domain, although k_s=3 has more weights than the others, the difference is not very significant. Therefore, one should also take into account the case when k_s=1 and k_s=2 to have a good prediction performance. Such an observation intuitively explains the reason why traditional *multi-subdomain* DA methods with a fixed number of subdomains cannot achieve good results. Figure 2b provides the mapping matrix $\boldsymbol{S}$ of the case when k_t=2 and $k_s \in \{2, 3\}$. When k_s=2, the first target subdomain maps to the second source subdomain, while the second target subdomain maps to the first source subdomain. When k_s=3, all subdomains of the target domain are mapped to the first source subdomain.

Conclusion

Standing in a *target to source* DA scenario, we provide a predictor reweighting method for a *multi-subdomain* DA scenario in weakly supervised and unsupervised settings. We first introduce a general subdomain division criterion and then specialize in a real-life case of temporal drift. We evaluate HSAV over two fraud detection datasets and illustrate the estimated parameters. Our current proposition focuses more on the small number of hidden subdomains. We aim to generalize our method to deal with a larger number of subdomains for future research.

References

Baktashmotlagh, M.; Harandi, M. T.; Lovell, B. C.; and Salzmann, M. 2013. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, 769–776.

Bobu, A.; Tzeng, E.; Hoffman, J.; and Darrell, T. 2018. Adapting to Continuously Shifting Domains.

Breiman, L. 1996. Bagging predictors. *Machine learning*, 24(2): 123–140.

Courty, N.; Flamary, R.; Habrard, A.; and Rakotomamonjy, A. 2017. Joint distribution optimal transportation for domain adaptation. In *NIPS*.

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

Duan, L.; Tsang, I. W.; Xu, D.; and Chua, T.-S. 2009. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML*, 289–296.

Fernando, B.; Habrard, A.; Sebban, M.; and Tuytelaars, T. 2013. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2960–2967.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *JMLR*, 17(1): 2096–2030.

Gong, B.; Grauman, K.; and Sha, F. 2013. Reshaping visual datasets for domain adaptation. *NIPS*, 26: 1286–1294.

Hoffman, J.; Kulis, B.; Darrell, T.; and Saenko, K. 2012. Discovering latent domains for multisource domain adaptation. In *ECCV*, 702–715. Springer.

Hoffman, J.; Mohri, M.; and Zhang, N. 2018. Algorithms and Theory for Multiple-Source Adaptation. In *NeurIPS*.

Kurmi, V. K.; Subramanian, V. K.; and Namboodiri, V. P. 2021. Domain Impression: A Source Data Free Domain Adaptation Method. In *WACV*, 615–625.

Li, W.; Xu, Z.; Xu, D.; Dai, D.; and Van Gool, L. 2017. Domain generalization and adaptation using low rank exemplar SVMs. *IEEE TPAMI*, 40(5): 1114–1127.

Li, Y.; Murias, M.; Major, S.; Dawson, G.; and Carlson, D. E. 2018. Extracting relationships by multi-domain matching. In *NeurIPS*, 6799–6810.

Liang, J.; Hu, D.; and Feng, J. 2020. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *ICML*, 6028–6039. PMLR.

Lin, Y.; Lee, Y.; and Wahba, G. 2002. Support vector machines for classification in nonstandard situations. *Machine learning*, 46(1-3): 191–202.

Liu, H.; Shao, M.; and Fu, Y. 2016. Structure-preserved multi-source domain adaptation. In *ICDM*, 1059–1064. IEEE.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. I. 2015. Learning Transferable Features with Deep Adaptation Networks. In *ICML*.

Long, M.; Cao, Z.; Wang, J.; and Jordan, M. I. 2018. Conditional adversarial domain adaptation. In *NeurIPS*, 1640–1650.

Long, M.; Wang, J.; Ding, G.; Sun, J.; and Yu, P. S. 2013. Transfer feature learning with joint distribution adaptation. In *ICCV*.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2017. Deep transfer learning with joint adaptation networks. In *ICML*, 2208–2217. PMLR.

Mancini, M.; Porzi, L.; Bulo, S. R.; Caputo, B.; and Ricci, E. 2018. Boosting domain adaptation by discovering latent domains. In *CVPR*, 3771–3780.

Mancini, M.; Porzi, L.; Bulo, S. R.; Caputo, B.; and Ricci, E. 2019. Inferring latent domains for unsupervised deep domain adaptation. *IEEE TPAMI*.

Mansour, Y.; Mohri, M.; and Rostamizadeh, A. 2009. Domain Adaptation with Multiple Sources. In Koller, D.; Schuurmans, D.; Bengio, Y.; and Bottou, L., eds., *NIPS*, volume 21. Curran Associates, Inc.

Nelder, J. A.; and Mead, R. 1965. A simplex method for function minimization. *The computer journal*, 7(4): 308–313.

Pan, S. J.; Tsang, I. W.; Kwok, J. T.; and Yang, Q. 2010. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2): 199–210.

Pan, S. J.; and Yang, Q. 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10): 1345–1359.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *ICCV*, 1406–1415.

Perrot, M.; Courty, N.; Flamary, R.; and Habrard, A. 2016. Mapping estimation for discrete optimal transport. In *NIPS*, 4204–4212.

Saito, K.; Watanabe, K.; Ushiku, Y.; and Harada, T. 2018. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 3723–3732.

Shimodaira, H. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2): 227–244.

Sugiyama, M.; Nakajima, S.; Kashima, H.; Buenau, P. V.; and Kawanabe, M. 2008. Direct importance estimation with model selection and its application to covariate shift adaptation. In *NIPS*.

Sun, B.; Feng, J.; and Saenko, K. 2017. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, 153–171. Springer.

Torralba, A.; and Efros, A. A. 2011. Unbiased look at dataset bias. In *CVPR*, 1521–1528. IEEE.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *CVPR*, 7167–7176.

Venkat, N.; Kundu, J.; Singh, D.; Revanur, A.; and Babu, R. 2021. Your Classifier can Secretly Suffice Multi-Source Domain Adaptation. In *NeurIPS*.

Wang, H.; He, H.; and Katabi, D. 2020. Continuously Indexed Domain Adaptation. In III, H. D.; and Singh, A., eds., *ICML*, volume 119 of *Proceedings of Machine Learning Research*, 9898–9907. PMLR.

Xu, R.; Chen, Z.; Zuo, W.; Yan, J.; and Lin, L. 2018. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *CVPR*, 3964–3973.

Xu, Z.; Li, W.; Niu, L.; and Xu, D. 2014. Exploiting low-rank structure from latent domains for domain generalization. In *ECCV*, 628–643. Springer.

Yeh, H.-W.; Yang, B.; Yuen, P. C.; and Harada, T. 2021. SoFA: Source-Data-Free Feature Alignment for Unsupervised Domain Adaptation. In *WACV*, 474–483.

Yosinski, J.; Clune, J.; Bengio, Y.; and Lipson, H. 2014. How transferable are features in deep neural networks? In *NIPS*, 3320–3328.

Zhang, L.; Germain, P.; Kessaci, Y.; and Biernacki, C. 2021a. Interpretable Domain Adaptation Using Unsupervised Feature Selection on Pre-trained Source Models.

Zhang, L.; Germain, P.; Kessaci, Y.; and Biernacki, C. 2021b. Target to Source Coordinate-Wise Adaptation of Pre-trained Models. In *ECML PKDD*, 378–394. Springer International Publishing.

Zhao, H.; Zhang, S.; Wu, G.; Moura, J. M.; Costeira, J. P.; and Gordon, G. J. 2018. Adversarial multiple source domain adaptation. *NeurIPS*, 31: 8559–8570.

Zhao, S.; Wang, G.; Zhang, S.; Gu, Y.; Li, Y.; Song, Z.; Xu, P.; Hu, R.; Chai, H.; and Keutzer, K. 2020. Multi-source distilling domain adaptation. In *AAAI*, 07, 12975–12983.