Domain Adaptation with Conditional Distribution Matching and Generalized Label Shift

Remi Tachet des Combes*
Microsoft Research Montreal
Montreal, QC, Canada
retachet@microsoft.com

Han Zhao*
D. E. Shaw & Co.
New York, NY, USA
han.zhao@cs.cmu.edu

Yu-Xiang Wang
UC Santa Barbara
Santa Barbara, CA, USA
yuxiangw@cs.ucsb.edu

Geoff Gordon
Microsoft Research Montreal
Montreal, QC, Canada
ggordon@microsoft.com

Abstract

Adversarial learning has demonstrated good performance in the unsupervised domain adaptation setting, by learning domain-invariant representations. However, recent work has shown limitations of this approach when label distributions differ between the source and target domains. In this paper, we propose a new assumption, *generalized label shift* (*GLS*), to improve robustness against mismatched label distributions. *GLS* states that, conditioned on the label, there exists a representation of the input that is invariant between the source and target domains. Under *GLS*, we provide theoretical guarantees on the transfer performance of any classifier. We also devise necessary and sufficient conditions for *GLS* to hold, by using an estimation of the relative class weights between domains and an appropriate reweighting of samples. Our weight estimation method could be straightforwardly and generically applied in existing domain adaptation (DA) algorithms that learn domain-invariant representations, with small computational overhead. In particular, we modify three DA algorithms, JAN, DANN and CDAN, and evaluate their performance on standard and artificial DA tasks. Our algorithms outperform the base versions, with vast improvements for large label distribution mismatches. Our code is available at https://tinyurl.com/y585xt6j.

1 Introduction

In spite of impressive successes, most deep learning models [24] rely on huge amounts of labelled data and their features have proven brittle to distribution shifts [43, 59]. Building more robust models, that learn from fewer samples and/or generalize better out-of-distribution is the focus of many recent works [2, 5, 57]. The research direction of interest to this paper is that of domain adaptation, which aims at learning features that transfer well between domains. We focus in particular on unsupervised domain adaptation (UDA), where the algorithm has access to labelled samples from a source domain and unlabelled data from a target domain. Its objective is to train a model that generalizes well to the target domain. Building on advances in adversarial learning [25], adversarial domain adaptation (ADA) leverages the use of a discriminator to learn an intermediate representation that is invariant between the source and target domains. Simultaneously, the representation is paired with a classifier, trained to perform well on the source domain [22, 36, 53, 64]. ADA is rather successful on a variety

*The first two authors contributed equally to this work. Work done while HZ was at Carnegie Mellon University.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

of tasks, however, recent work has proven an upper bound on the performance of existing algorithms when source and target domains have mismatched label distributions [66]. Label shift is a property of two domains for which the marginal label distributions differ, but the conditional distributions of input given label stay the same across domains [52, 62].

In this paper, we study domain adaptation under mismatched label distributions and design methods that are robust in that setting. Our contributions are the following. First, we extend the upper bound by Zhao et al. [66] to k-class classification and to conditional domain adversarial networks, a recently introduced domain adaptation algorithm [41]. Second, we introduce *generalized label shift* (GLS), a broader version of the standard label shift where conditional invariance between source and target domains is placed in representation rather than input space. Third, we derive performance guarantees for algorithms that seek to enforce GLS via learnt feature transformations, in the form of upper bounds on the error gap and the joint error of the classifier on the source and target domains. Those guarantees suggest principled modifications to ADA to improve its robustness to mismatched label distributions. The modifications rely on estimating the class ratios between source and target domains and use those as importance weights in the adversarial and classification objectives. The importance weights estimation is performed using a method of moment by solving a quadratic program, inspired from Lipton et al. [35]. Following these theoretical insights, we devise three new algorithms based on learning importance-weighted representations, DANNs [22], JANs [40] and CDANs [41]. We apply our variants to artificial UDA tasks with large divergences between label distributions, and demonstrate significant performance gains compared to the algorithms' base versions. Finally, we evaluate them on standard domain adaptation tasks and also show improved performance.

2 Preliminaries

Notation We focus on the general k-class classification problem. $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output space, respectively. $\mathcal{Z}$ stands for the representation space induced from $\mathcal{X}$ by a feature transformation $g : \mathcal{X} \mapsto \mathcal{Z}$. Accordingly, we use X, Y, Z to denote random variables which take values in $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$. *Domain* corresponds to a joint distribution on the input space $\mathcal{X}$ and output space $\mathcal{Y}$, and we use $\mathcal{D}_S$ (resp. $\mathcal{D}_T$) to denote the source (resp. target) domain. Noticeably, this corresponds to a stochastic setting, which is stronger than the deterministic one previously studied [6, 7, 66]. A *hypothesis* is a function $h : \mathcal{X} \to [k]$. The *error* of a hypothesis h under distribution $\mathcal{D}_S$ is defined as: $\varepsilon_S(h) := \Pr_{\mathcal{D}_S}(h(X) \neq Y)$, i.e., the probability that h disagrees with Y under $\mathcal{D}_S$.

Domain Adaptation via Invariant Representations For source ($\mathcal{D}_S$) and target ($\mathcal{D}_T$) domains, we use $\mathcal{D}_S^X$, $\mathcal{D}_T^X$, $\mathcal{D}_S^Y$ and $\mathcal{D}_T^Y$ to denote the marginal data and label distributions. In UDA, the algorithm has access to n labeled points $\{(\mathbf{x}_i, y_i)\}_{i=1}^n \in (\mathcal{X} \times \mathcal{Y})^n$ and m unlabeled points $\{\mathbf{x}_j\}_{j=1}^m \in \mathcal{X}^m$ sampled i.i.d. from the source and target domains. Inspired by Ben-David et al. [7], a common approach is to learn representations invariant to the domain shift. With $g : \mathcal{X} \mapsto \mathcal{Z}$ a feature transformation and $h : \mathcal{Z} \mapsto \mathcal{Y}$ a hypothesis on the feature space, a domain invariant representation [22, 53, 63] is a function g that induces similar distributions on $\mathcal{D}_S$ and $\mathcal{D}_T$. g is also required to preserve rich information about the target task so that $\varepsilon_S(h \circ g)$ is small. The above process results in the following Markov chain (assumed to hold throughout the paper):

$$X \xrightarrow{g} Z \xrightarrow{h} \widehat{Y}, \tag{1}$$

where $\widehat{Y} = h(g(X))$. We let $\mathcal{D}_S^Z$, $\mathcal{D}_T^Z$, $\mathcal{D}_S^{\widehat{Y}}$ and $\mathcal{D}_T^{\widehat{Y}}$ denote the pushforwards (induced distributions) of $\mathcal{D}_S^X$ and $\mathcal{D}_T^X$ by g and $h \circ g$. Invariance in feature space is defined as minimizing a distance or divergence between $\mathcal{D}_S^Z$ and $\mathcal{D}_T^Z$.

Adversarial Domain Adaptation Invariance is often attained by training a discriminator $d : \mathcal{Z} \mapsto [0, 1]$ to predict if z is from the source or target. g is trained both to maximize the discriminator loss and minimize the classification loss of $h \circ g$ on the source domain (h is also trained with the latter objective). This leads to domain-adversarial neural networks [22, DANN], where g, h and d are parameterized with neural networks: g_θ, h_ϕ and d_ψ (see Algo. 1 and App. B.5). Building on DANN, conditional domain adversarial networks [41, CDAN] use the same adversarial paradigm. However, the discriminator now takes as input the outer product, for a given x, between the predictions of the network $h(g(x))$ and its representation $g(x)$. In other words, d acts on the outer product: $h \otimes g(x) := (h_1(g(x)) \cdot g(x), \ldots, h_k(g(x)) \cdot g(x))$ rather than on $g(x)$. h_i denotes the i-th element of vector h. We now highlight a limitation of DANNs and CDANs.

Table 1: Common assumptions in the domain adaptation literature.

Covariate Shift	Label Shift
$\mathcal{D}_S^X \neq \mathcal{D}_T^X$	$\mathcal{D}_S^Y \neq \mathcal{D}_T^Y$
$\forall \mathbf{x} \in \mathcal{X}, \mathcal{D}_S(Y \mid X = \mathbf{x}) = \mathcal{D}_T(Y \mid X = \mathbf{x})$	$\forall y \in \mathcal{Y}, \mathcal{D}_S(X \mid Y = y) = \mathcal{D}_T(X \mid Y = y)$

An Information-Theoretic Lower Bound We let D_{JS} denote the Jensen-Shanon divergence between two distributions (App. A.1), and $\widetilde{Z}$ correspond to Z (for DANN) or to $\widehat{Y} \otimes Z$ (for CDAN). The following theorem lower bounds the joint error of the classifier on the source and target domains:

Theorem 2.1. Assuming that $D_{\text{JS}}(\mathcal{D}_S^Y \parallel \mathcal{D}_T^Y) \geq D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}} \parallel \mathcal{D}_T^{\widetilde{Z}})$, then:

$$\varepsilon_S(h \circ g) + \varepsilon_T(h \circ g) \geq \frac{1}{2}\left(\sqrt{D_{\text{JS}}(\mathcal{D}_S^Y \parallel \mathcal{D}_T^Y)} - \sqrt{D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}} \parallel \mathcal{D}_T^{\widetilde{Z}})}\right)^2.$$

Remark The lower bound is algorithm-independent. It is also a population-level result and holds asymptotically with increasing data. Zhao et al. [66] prove the theorem for $k = 2$ and $\widetilde{Z} = Z$. We extend it to CDAN and arbitrary k (it actually holds for any $\widetilde{Z}$ s.t. $\widehat{Y} = \widetilde{h}(\widetilde{Z})$ for some $\widetilde{h}$, see App. A.3). Assuming that label distributions differ between source and target domains, the lower bound shows that: *the better the alignment of feature distributions, the worse the joint error.* For an invariant representation ($D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}}, \mathcal{D}_T^{\widetilde{Z}}) = 0$) with no source error, the target error will be larger than $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)/2$. Hence algorithms learning invariant representations and minimizing the source error are fundamentally flawed when label distributions differ between source and target domains.

Common Assumptions to Tackle Domain Adaptation Two common assumptions about the data made in DA are *covariate shift* and *label shift*. They correspond to different ways of decomposing the joint distribution over $X \times Y$, as detailed in Table 1. From a representation learning perspective, covariate shift is not robust to feature transformation and can lead to an effect called negative transfer [66]. At the same time, label shift clearly fails in most practical applications, e.g. transferring knowledge from synthetic to real images [55]. In that case, the input distributions are actually disjoint.

3 Main Results

In light of the limitations of existing assumptions, (e.g. covariate shift and label shift), we propose *generalized label shift* (*GLS*), a relaxation of label shift that substantially improves its applicability. We first discuss some of its properties and explain why the assumption is favorable in domain adaptation based on representation learning. Motivated by *GLS*, we then present a novel error decomposition theorem that directly suggests a bound minimization framework for domain adaptation. The framework is naturally compatible with $\mathcal{F}$-integral probability metrics [44, $\mathcal{F}$-IPM] and generates a family of domain adaptation algorithms by choosing various function classes $\mathcal{F}$. In a nutshell, the proposed framework applies a method of moments [35] to estimate the importance weight $\mathbf{w}$ of the *marginal label distributions* by solving a quadratic program (QP), and then uses $\mathbf{w}$ to align the weighted source feature distribution with the target feature distribution.

3.1 Generalized Label Shift

Definition 3.1 (Generalized Label Shift, *GLS*)**.** A representation $Z = g(X)$ satisfies *GLS* if

$$\mathcal{D}_S(Z \mid Y = y) = \mathcal{D}_T(Z \mid Y = y),\ \forall y \in \mathcal{Y}. \tag{2}$$

First, when g is the identity map, the above definition of *GLS* reduces to the original label shift assumption. Next, *GLS* is always achievable for any distribution pair $(\mathcal{D}_S, \mathcal{D}_T)$: any constant function $g \equiv c \in \mathbb{R}$ satisfies the above definition. The most important property is arguably that, unlike label shift, *GLS* is compatible with a perfect classifier (in the noiseless case). Precisely, if there exists a ground-truth labeling function h^* such that $Y = h^*(X)$, then h^* satisfies *GLS*. As a comparison, without conditioning on $Y = y$, h^* does not satisfy $\mathcal{D}_S(h^*(X)) = \mathcal{D}_T(h^*(X))$ if the marginal label distributions are different across domains. This observation is consistent with the lower bound in Theorem 2.1, which holds for arbitrary marginal label distributions.

GLS imposes label shift in the feature space $\mathcal{Z}$ instead of the original input space $\mathcal{X}$. Conceptually, although samples from the same classes in the source and target domain can be dramatically different, the hope is to find an intermediate representation for both domains in which samples from a given class look similar to one another. Taking digit classification as an example and assuming the feature variable Z corresponds to the contour of a digit, it is possible that by using different contour extractors for e.g. MNIST and USPS, those contours look roughly the same in both domains. Technically, GLS can be facilitated by having separate representation extractors g_S and g_T for source and target [9, 53].

3.2 An Error Decomposition Theorem based on GLS

We now provide performance guarantees for models that satisfy GLS, in the form of upper bounds on the error gap and on the joint error between source and target domains. It requires two concepts:

Definition 3.2 (Balanced Error Rate)**.** The *balanced error rate* (BER) of predictor $\widehat{Y}$ on $\mathcal{D}_S$ is:

$$\mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y) := \max_{j \in [k]} \mathcal{D}_S(\widehat{Y} \neq Y \mid Y = j). \tag{3}$$

Definition 3.3 (Conditional Error Gap)**.** Given a joint distribution $\mathcal{D}$, the *conditional error gap* of a classifier $\widehat{Y}$ is $\Delta_{\mathrm{CE}}(\widehat{Y}) := \max_{y \neq y' \in \mathcal{Y}^2} |\mathcal{D}_S(\widehat{Y} = y' \mid Y = y) - \mathcal{D}_T(\widehat{Y} = y' \mid Y = y)|$.

When GLS holds, $\Delta_{\mathrm{CE}}(\widehat{Y})$ is equal to 0. We now give an upper bound on the error gap between source and target, which can also be used to obtain a generalization upper bound on the target risk.

Theorem 3.1. (Error Decomposition Theorem) For any classifier $\widehat{Y} = (h \circ g)(X)$,

$$|\varepsilon_S(h \circ g) - \varepsilon_T(h \circ g)| \leq \ \|\mathcal{D}_S^Y - \mathcal{D}_T^Y\|_1 \cdot \mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}),$$

where $\|\mathcal{D}_S^Y - \mathcal{D}_T^Y\|_1 := \sum_{i=1}^k |\mathcal{D}_S(Y = i) - \mathcal{D}_T(Y = i)|$ is the L_1 distance between $\mathcal{D}_S^Y$ and $\mathcal{D}_T^Y$.

Remark The upper bound in Theorem 3.1 provides a way to decompose the error gap between source and target domains. It also immediately gives a generalization bound on the target risk $\varepsilon_T(h \circ g)$. The bound contains two terms. The first contains $\|\mathcal{D}_S^Y - \mathcal{D}_T^Y\|_1$, which measures the distance between the marginal label distributions across domains and is a constant that only depends on the adaptation problem itself, and BER, a reweighted classification performance on the source domain. The second is $\Delta_{\mathrm{CE}}(\widehat{Y})$ measures the distance between the family of conditional distributions $\widehat{Y} \mid Y$. In other words, the bound is *oblivious* to the optimal labeling functions in feature space. This is in sharp contrast with upper bounds from previous work [7, Theorem 2], [66, Theorem 4.1], which essentially decompose the error gap in terms of the distance between the marginal feature distributions ($\mathcal{D}_S^Z$, $\mathcal{D}_T^Z$) and the optimal labeling functions (f_S^Z, f_T^Z). Because the optimal labeling function in feature space depends on Z and is unknown in practice, such decomposition is not very informative. As a comparison, Theorem 3.1 provides a decomposition orthogonal to previous results and does not require knowledge about unknown optimal labeling functions in feature space.

Notably, the balanced error rate, $\mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y)$, only depends on samples from the source domain and can be minimized. Furthermore, using a data-processing argument, the conditional error gap $\Delta_{\mathrm{CE}}(\widehat{Y})$ can be minimized by aligning the conditional feature distributions across domains. Putting everything together, the result suggests that, to minimize the error gap, it suffices to align the conditional distributions $Z \mid Y = y$ while simultaneously minimizing the balanced error rate. In fact, under the assumption that the conditional distributions are perfectly aligned (i.e., under GLS), we can prove a stronger result, guaranteeing that the joint error is small:

Theorem 3.2. If $Z = g(X)$ satisfies GLS, then for any $h : \mathcal{Z} \to \mathcal{Y}$ and letting $\widehat{Y} = h(Z)$ be the predictor, we have $\varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) \leq 2\mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y)$.

3.3 Conditions for Generalized Label Shift

The main difficulty in applying a bound minimization algorithm inspired by Theorem 3.1 is that we do not have labels from the target domain in UDA, so we cannot directly align the conditional label distributions. By using relative class weights between domains, we can provide a necessary condition for GLS that bypasses an explicit alignment of the conditional feature distributions.

Definition 3.4. Assuming $\mathcal{D}_S(Y = y) > 0, \forall y \in \mathcal{Y}$, we let $\mathbf{w} \in \mathbb{R}^k$ denote the importance weights of the target and source label distributions:

$$\mathbf{w}_y := \frac{\mathcal{D}_T(Y = y)}{\mathcal{D}_S(Y = y)}, \quad \forall y \in \mathcal{Y}. \tag{4}$$

Lemma 3.1. (Necessary condition for *GLS*) If $Z = g(X)$ satisfies *GLS*, then $\mathcal{D}_T(\widetilde{Z}) = \sum_{y \in \mathcal{Y}} \mathbf{w}_y \cdot \mathcal{D}_S(\widetilde{Z}, Y = y) =: \mathcal{D}_S^{\mathbf{w}}(\widetilde{Z})$ where $\widetilde{Z}$ verifies either $\widetilde{Z} = Z$ or $\widetilde{Z} = \widehat{Y} \otimes Z$.

Compared to previous work that attempts to align $\mathcal{D}_T(Z)$ with $\mathcal{D}_S(Z)$ (using adversarial discriminators [22] or maximum mean discrepancy (MMD) [38]) or $\mathcal{D}_T(\hat{Y} \otimes Z)$ with $\mathcal{D}_S(\hat{Y} \otimes Z)$ [41], Lemma 3.1 suggests to instead align $\mathcal{D}_T(\widetilde{Z})$ with the *reweighted* marginal distribution $\mathcal{D}_S^{\mathbf{w}}(\widetilde{Z})$. Reciprocally, the following two theorems give sufficient conditions to know when perfectly aligned target feature distribution and reweighted source feature distribution imply *GLS*:

Theorem 3.3. (Clustering structure implies sufficiency) Let $Z = g(X)$ such that $\mathcal{D}_T(Z) = \mathcal{D}_S^{\mathbf{w}}(Z)$. Assume $\mathcal{D}_T(Y = y) > 0, \forall y \in \mathcal{Y}$. If there exists a partition of $\mathcal{Z} = \cup_{y \in \mathcal{Y}} \mathcal{Z}_y$ such that $\forall y \in \mathcal{Y}$, $\mathcal{D}_S(Z \in \mathcal{Z}_y \mid Y = y) = \mathcal{D}_T(Z \in \mathcal{Z}_y \mid Y = y) = 1$, then $Z = g(X)$ satisfies *GLS*.

Remark Theorem 3.3 shows that if there exists a partition of the feature space such that instances with the same label are within the same component, then aligning the target feature distribution with the reweighted source feature distribution implies *GLS*. While this clustering assumption may seem strong, it is consistent with the goal of reducing classification error: if such a clustering exists, then there also exists a perfect predictor based on the feature $Z = g(X)$, i.e., the cluster index.

Theorem 3.4. Let $\widehat{Y} = h(Z)$, $\gamma := \min_{y \in \mathcal{Y}} \mathcal{D}_T(Y = y)$ and $\mathbf{w}_M := \max_{y \in \mathcal{Y}} \mathbf{w}_y$. For $\widetilde{Z} = Z$ or $\widetilde{Z} = \hat{Y} \otimes Z$, we have:

$$\max_{y \in \mathcal{Y}} d_{\mathrm{TV}}(\mathcal{D}_S(Z \mid Y = y), \mathcal{D}_T(Z \mid Y = y)) \leq \frac{\mathbf{w}_M \varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) + \sqrt{8 D_{\mathrm{JS}}(\mathcal{D}_S^{\mathbf{w}}(\widetilde{Z}) \| \mathcal{D}_T(\widetilde{Z}))}}{\gamma}.$$

Theorem 3.4 confirms that matching $\mathcal{D}_T(\widetilde{Z})$ with $\mathcal{D}_S^{\mathbf{w}}(\widetilde{Z})$ is the proper objective in the context of mismatched label distributions. It shows that, for matched feature distributions and a source error equal to zero, successful domain adaptation (*i.e.* a target error equal to zero) implies that *GLS* holds. Combined with Theorem 3.2, we even get equivalence between the two.

Remark Thm. 3.4 extends Thm. 3.3 by incorporating the clustering assumption in the joint error achievable by a classifier $\widehat{Y}$ based on a fixed Z. In particular, if the clustering structure holds, the joint error is 0 for an appropriate h, and aligning the reweighted feature distributions implies *GLS*.

3.4 Estimating the Importance Weights w

Inspired by the moment matching technique to estimate $\mathbf{w}$ under label shift from Lipton et al. [35], we propose a method to get $\mathbf{w}$ under *GLS* by solving a quadratic program (QP).

Definition 3.5. We let $\mathbf{C} \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{Y}|}$ denote the confusion matrix of the classifier on the source domain and $\boldsymbol{\mu} \in \mathbb{R}^{|\mathcal{Y}|}$ the distribution of predictions on the target one, $\forall y, y' \in \mathcal{Y}$:

$$\mathbf{C}_{y,y'} := \mathcal{D}_S(\widehat{Y} = y, Y = y'), \qquad \boldsymbol{\mu}_y := \mathcal{D}_T(\widehat{Y} = y).$$

Lemma 3.2. If *GLS* is verified, and if the confusion matrix $\mathbf{C}$ is invertible, then $\mathbf{w} = \mathbf{C}^{-1}\boldsymbol{\mu}$.

The key insight from Lemma 3.2 is that, to estimate the importance vector $\mathbf{w}$ under *GLS*, we do not need access to labels from the target domain. However, matrix inversion is notoriously numerically unstable, especially with finite sample estimates $\hat{\mathbf{C}}$ and $\hat{\boldsymbol{\mu}}$ of $\mathbf{C}$ and $\boldsymbol{\mu}$. We propose to solve instead the following QP (written as $QP(\hat{\mathbf{C}}, \hat{\boldsymbol{\mu}})$), whose solution will be consistent if $\hat{\mathbf{C}} \to \mathbf{C}$ and $\hat{\boldsymbol{\mu}} \to \boldsymbol{\mu}$:

$$\underset{\mathbf{w}}{\text{minimize}} \quad \frac{1}{2} \|\hat{\boldsymbol{\mu}} - \hat{\mathbf{C}}\mathbf{w}\|_2^2, \qquad \text{subject to} \quad \mathbf{w} \geq 0,\ \mathbf{w}^T \mathcal{D}_S(Y) = 1. \tag{5}$$

The above program (5) can be efficiently solved in time $O(|\mathcal{Y}|^3)$, with $|\mathcal{Y}|$ small and constant; and by construction, its solution is element-wise non-negative, even with limited amounts of data to estimate $\mathbf{C}$ and $\boldsymbol{\mu}$.

Algorithm 1 Importance-Weighted Domain Adaptation

1: **Input:** source and target data (x_S, y_S), x_T; g_θ, h_ϕ and d_ψ; epochs E, batches per epoch B
2: Initialize $\mathbf{w}_1 = 1$
3: **for** $t = 1$ **to** E **do**
4: Initialize $\hat{\mathbf{C}} = 0$, $\hat{\boldsymbol{\mu}} = 0$
5: **for** $b = 1$ **to** B **do**
6: Sample batches (x_S^i, y_S^i) and (x_T^i) of size s
7: Maximize $\mathcal{L}_{DA}^{\mathbf{w}_t}$ w.r.t. θ, minimize $\mathcal{L}_{DA}^{\mathbf{w}_t}$ w.r.t. ψ and minimize $\mathcal{L}_C^{\mathbf{w}_t}$ w.r.t. θ and ϕ
8: **for** $i = 1$ **to** s **do**
9: $\hat{\mathbf{C}}_{\cdot y_S^i} \leftarrow \hat{\mathbf{C}}_{\cdot y_S^i} + h_\phi(g_\theta(x_S^i))$ (y_S^i-th column) and $\hat{\boldsymbol{\mu}} \leftarrow \hat{\boldsymbol{\mu}} + h_\phi(g_\theta(x_T^i))$
10: $\hat{\mathbf{C}} \leftarrow \hat{\mathbf{C}}/sB$ and $\hat{\boldsymbol{\mu}} \leftarrow \hat{\boldsymbol{\mu}}/sB$; then $\mathbf{w}_{t+1} = \lambda \cdot QP(\hat{\mathbf{C}}, \hat{\boldsymbol{\mu}}) + (1-\lambda)\mathbf{w}_t$

Lemma 3.3. If the source error $\varepsilon_S(h \circ g)$ is zero and the source and target marginals verify $D_{\text{JS}}(\mathcal{D}_S^{\tilde{\mathbf{w}}}(Z), \mathcal{D}_T(Z)) = 0$, then the estimated weight vector $\mathbf{w}$ is equal to $\tilde{\mathbf{w}}$.

Lemma 3.3 shows that the weight estimation is stable once the DA losses have converged, but it does not imply convergence to the true weights (see Sec. 4.2 and App. B.8 for more details).

3.5 $\mathcal{F}$-IPM for Distributional Alignment

To align the target feature distribution and the reweighted source feature distribution as suggested by Lemma 3.1, we now provide a general framework using the integral probability metric [44, IPM].

Definition 3.6. With $\mathcal{F}$ a set of real-valued functions, the $\mathcal{F}$-IPM between distributions $\mathcal{D}$ and $\mathcal{D}'$ is

$$d_{\mathcal{F}}(\mathcal{D}, \mathcal{D}') := \sup_{f \in \mathcal{F}} |\mathbb{E}_{X \sim \mathcal{D}}[f(X)] - \mathbb{E}_{X \sim \mathcal{D}'}[f(X)]|. \tag{6}$$

By approximating any function class $\mathcal{F}$ using parametrized models, e.g., neural networks, we obtain a general framework for domain adaptation by aligning reweighted source feature distribution and target feature distribution, i.e. by minimizing $d_{\mathcal{F}}(\mathcal{D}_T(\widetilde{Z}), \mathcal{D}_S^{\mathbf{w}}(\widetilde{Z}))$. Below, by instantiating $\mathcal{F}$ to be the set of bounded norm functions in a RKHS $\mathcal{H}$ [27], we obtain maximum mean discrepancy methods, leading to IWJAN (cf. Section 4.1), a variant of JAN [40] for UDA.

4 Practical Implementation

4.1 Algorithms

The sections above suggest simple algorithms based on representation learning: (i) estimate $\mathbf{w}$ on the fly during training, (ii) align the feature distributions $\widetilde{Z}$ of the target domain with the reweighted feature distribution of the source domain and, (iii) minimize the balanced error rate. Overall, we present the pseudocode of our algorithm in Alg. 1.

To compute $\mathbf{w}$, we build estimators $\hat{\mathbf{C}}$ and $\hat{\boldsymbol{\mu}}$ of $\mathbf{C}$ and $\boldsymbol{\mu}$ by averaging during each epoch the predictions of the classifier on the source (per true class) and target (overall). This corresponds to the inner-most loop of Algorithm 1 (lines 8-9). At epoch end, $\mathbf{w}$ is updated (line 10), and the estimators reset to 0. We have found empirically that using an exponential moving average of $\mathbf{w}$ performs better. Our results all use a factor $\lambda = 0.5$. We also note that Alg. 1 implies a minimal computational overhead (see App. B.1 for details): in practice our algorithms run as fast as their base versions.

Using $\mathbf{w}$, we can define our first algorithm, *Importance-Weighted Domain Adversarial Network* (IWDAN), that aligns $\mathcal{D}_S^{\mathbf{w}}(Z)$ and $\mathcal{D}_T(Z)$) using a discriminator. To that end, we modify the DANN losses $\mathcal{L}_{DA}$ and $\mathcal{L}_C$ as follows. For batches (x_S^i, y_S^i) and (x_T^i) of size s, the weighted DA loss is:

$$\mathcal{L}_{DA}^{\mathbf{w}}(x_S^i, y_S^i, x_T^i; \theta, \psi) = -\frac{1}{s}\sum_{i=1}^{s} \mathbf{w}_{y_S^i} \log(d_\psi(g_\theta(x_S^i))) + \log(1 - d_\psi(g_\theta(x_T^i))). \tag{7}$$

We verify in App. A.1, that the standard ADA framework applied to $\mathcal{L}_{DA}^{\mathbf{w}}$ indeed minimizes $D_{\text{JS}}(\mathcal{D}_S^{\mathbf{w}}(Z) \| \mathcal{D}_T(Z))$. Our second algorithm, *Importance-Weighted Joint Adaptation Networks*

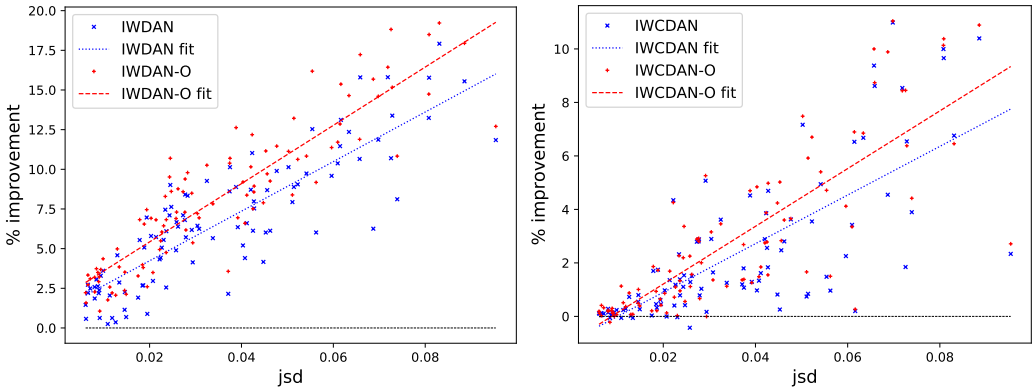


Figure 1: Gains of our algorithms vs their base versions (the horizontal grey line) for 100 tasks. The x-axis is $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$. The mean improvements for IWDAN and IWDAN-O (resp. IWCDAN and IWCDAN-O) are 6.55% and 8.14% (resp. 2.25% and 2.81%).

(IWJAN) is based on JAN [40] and follows the reweighting principle described in Section 3.5 with $\mathcal{F}$ a learnt RKHS (the exact JAN and IWJAN losses are specified in App. B.5). Finally, our third algorithm is *Importance-Weighted Conditional Domain Adversarial Network* (IWCDAN). It matches $\mathcal{D}_S^{\mathbf{w}}(\hat{Y} \otimes Z)$ with $\mathcal{D}_T(\hat{Y} \otimes Z)$ by replacing the standard adversarial loss in CDAN with Eq. 7, where d_ψ takes as input $(h_\phi \circ g_\theta) \otimes g_\theta$ instead of g_θ. The classifier loss for our three variants is:

$$\mathcal{L}_C^{\mathbf{w}}(x_S^i, y_S^i; \theta, \phi) = -\frac{1}{s}\sum_{i=1}^{s} \frac{1}{k \cdot \mathcal{D}_S(Y = y)} \log(h_\phi(g_\theta(x_S^i))_{y_S^i}). \tag{8}$$

This reweighting is suggested by our theoretical analysis from Section 3, where we seek to minimize the balanced error rate $\text{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y)$. We also define oracle versions, IWDAN-O, IWJAN-O and IWCDAN-O, where the weights $\mathbf{w}$ are the true weights. It gives an idealistic version of the reweighting method, and allows to assess the soundness of GLS. IWDAN, IWJAN and IWCDAN are Alg. 1 with their respective loss functions, the oracle versions use the true weights instead of $\mathbf{w}_t$.

4.2 Experiments

We apply our three base algorithms, their importance weighted versions, and the oracles to 4 standard DA datasets generating 21 tasks: Digits (MNIST $\leftrightarrow$ USPS [19, 32]), Visda [55], Office-31 [49] and Office-Home [54]. All values are averages over 5 runs of the best test accuracy throughout training (evaluated at the end of each epoch). We used that value for fairness with respect to the baselines (as shown in the left panel of Figure 2, the performance of DANN decreases as training progresses, due to the inappropriate matching of representations showcased in Theorem 2.1). For full details, see App. B.2 and B.7.

Performance vs $\mathbf{D_{JS}}$ We artificially generate 100 tasks from MNIST and USPS by considering various random subsets of the classes in either the source or target domain (see Appendix B.6 for details). These 100 DA tasks have a $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$ varying between 0 and 0.1. Applying IWDAN and IWCDAN results in Fig. 1. We see a clear correlation between the improvements provided by our algorithms and $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$, which is well aligned with Theorem 2.1. Moreover, IWDAN outperfoms DANN on the 100 tasks and IWCDAN bests CDAN on 94. Even on small divergences, our algorithms do not suffer compared to their base versions.

Original Datasets The average results on each dataset are shown in Table 2 (see App.B.3 for the per-task breakdown). IWDAN outperforms the basic algorithm DANN by 1.75%, 1.64%, 1.16% and 2.65% on the Digits, Visda, Office-31 and Office-Home tasks respectively. Gains for IWCDAN are more limited, but still present: 0.18%, 0.89%, 0.07% and 1.07% respectively. This might be explained by the fact that CDAN enforces a weak form of GLS (App. B.5.2). Gains for JAN are 0.58%, 0.19% and 0.19%. We also show the fraction of times (over all seeds and tasks) our variants outperform the original algorithms. Even for small gains, the variants provide consistent improvements. Additionally, the oracle versions show larger improvements, which strongly supports enforcing GLS.

Table 2: Average results on the various domains (Digits has 2 tasks, Visda 1, Office-31 6 and Office-Home 12). The prefix *s* denotes the experiment where the source domain is subsampled to increase $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$. Each number is a mean over 5 seeds, the subscript denotes the fraction of times (out of 5 *seeds* $\times$ #*tasks*) our algorithms outperform their base versions. JAN is not available on Digits.

METHOD	DIGITS	*s*DIGITS	VISDA	*s*VISDA	O-31	*s*O-31	O-H	*s*O-H
NO DA	77.17	75.67	48.39	49.02	77.81	75.72	56.39	51.34
DANN	93.15	83.24	61.88	52.85	82.74	76.17	59.62	51.83
IWDAN	**94.90**$_{100\%}$	**92.54**$_{100\%}$	**63.52**$_{100\%}$	**60.18**$_{100\%}$	**83.90**$_{87\%}$	**82.60**$_{100\%}$	**62.27**$_{97\%}$	**57.61**$_{100\%}$
IWDAN-O	95.27$_{100\%}$	94.46$_{100\%}$	64.19$_{100\%}$	62.10$_{100\%}$	85.33$_{97\%}$	84.41$_{100\%}$	64.68$_{100\%}$	60.87$_{100\%}$
CDAN	95.72	88.23	65.60	60.19	87.23	81.62	64.59	56.25
IWCDAN	**95.90**$_{80\%}$	**93.22**$_{100\%}$	**66.49**$_{60\%}$	**65.83**$_{100\%}$	**87.30**$_{73\%}$	**83.88**$_{100\%}$	**65.66**$_{70\%}$	**61.24**$_{100\%}$
IWCDAN-O	95.85$_{90\%}$	94.81$_{100\%}$	68.15$_{100\%}$	66.85$_{100\%}$	88.14$_{90\%}$	85.47$_{100\%}$	67.64$_{98\%}$	63.73$_{100\%}$
JAN	N/A	N/A	56.98	50.64	85.13	78.21	59.59	53.94
IWJAN	N/A	N/A	**57.56**$_{100\%}$	**57.12**$_{100\%}$	**85.32**$_{60\%}$	**82.61**$_{97\%}$	**59.78**$_{63\%}$	**55.89**$_{100\%}$
IWJAN-O	N/A	N/A	61.48$_{100\%}$	61.30$_{100\%}$	87.14$_{100\%}$	86.24$_{100\%}$	60.73$_{92\%}$	57.36$_{100\%}$

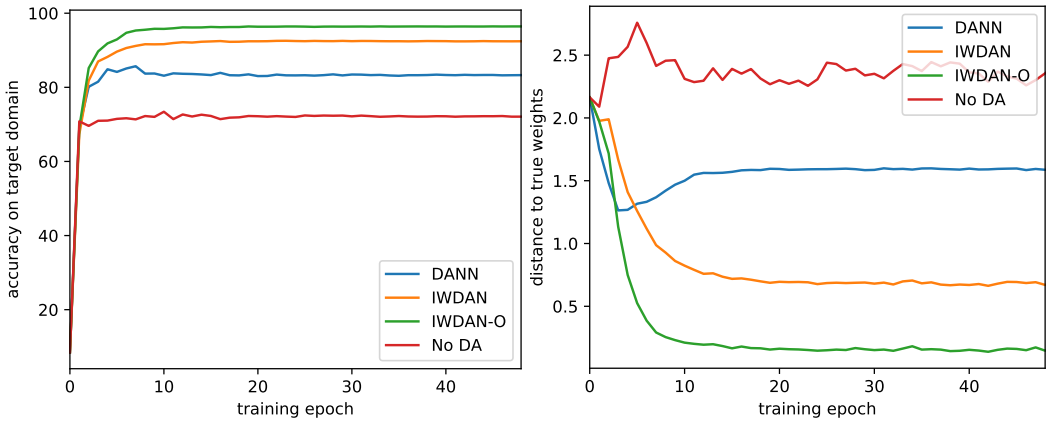


Figure 2: *Left* Accuracy on sDigits. *Right* Euclidian distance between estimated and true weights.

Subsampled datasets The original datasets have fairly balanced classes, making the JSD between source and target label distributions $D_{\text{JS}}(\mathcal{D}_S^Y \,\|\, \mathcal{D}_T^Y)$ rather small (Tables 11a, 12a and 13a in App. B.4). To evaluate our algorithms on larger divergences, we arbitrarily modify the source domains above by considering only 30% of the samples coming from the first half of their classes. This results in larger divergences (Tables 11b, 12b and 13b). Performance is shown in Table 2 (datasets prefixed by s). For IWDAN, we see gains of 9.3%, 7.33%, 6.43% and 5.58% on the digits, Visda, Office-31 and Office-Home datasets respectively. For IWCDAN, improvements are 4.99%, 5.64%, 2.26% and 4.99%, and IWJAN shows gains of 6.48%, 4.40% and 1.95%. Moreover, on all seeds and tasks but one, our variants outperform their base versions.

Importance weights While our method demonstrates gains empirically, Lemma 3.2 does not guarantee convergence of $\mathbf{w}$ to the true weights. In Fig. 2, we show the test accuracy and distance between estimated and true weights during training on sDigits. We see that DANN's performance gets worse after a few epoch, as predicted by Theorem 2.1. The representation matching objective collapses classes that are over-represented in the target domain on the under-represented ones (see App. B.9). This phenomenon does not occur for IWDAN and IWDAN-O. Both monotonously improve in accuracy and estimation (see Lemma 3.3 and App. B.8 for more details). We also observe that IWDAN's weights do not converge perfectly. This suggests that fine-tuning λ (we used $\lambda = 0.5$ in all our experiments for simplicity) or updating $\mathbf{w}$ more or less often could lead to better performance.

Ablation Study Our algorithms have two components, a weighted adversarial loss $\mathcal{L}_{DA}^{\mathbf{w}}$ and a weighted classification loss $\mathcal{L}_{C}^{\mathbf{w}}$. In Table 3, we augment DANN and CDAN using those losses separately (with the true weights). We observe that DANN benefits essentially from the reweighting of its adversarial loss $\mathcal{L}_{DA}^{\mathbf{w}}$, the classification loss has little effect. For CDAN, gains are essentially seen on the subsampled datasets. Both losses help, with a $+2\%$ extra gain for $\mathcal{L}_{DA}^{\mathbf{w}}$.

Table 3: Ablation study on the Digits tasks.

Method	Digits	sDigits	Method	Digits	sDigits
DANN	93.15	83.24	CDAN	95.72	88.23
DANN + $\mathcal{L}_C^{\mathbf{w}}$	93.27	84.52	CDAN + $\mathcal{L}_C^{\mathbf{w}}$	95.65	91.01
DANN + $\mathcal{L}_{DA}^{\mathbf{w}}$	**95.31**	**94.41**	CDAN + $\mathcal{L}_{DA}^{\mathbf{w}}$	95.42	93.18
IWDAN-O	**95.27**	**94.46**	IWCDAN-O	**95.85**	**94.81**

5 Related Work

Covariate shift has been studied and used in many adaptation algorithms [1, 3, 26, 30, 48, 53, 65]. While less known, label shift has also been tackled from various angles over the years: applying EM to learn $\mathcal{D}_T^Y$ [14], placing a prior on the label distribution [52], using kernel mean matching [20, 46, 61], etc. Schölkopf et al. [50] cast the problem in a causal/anti-causal perspective corresponding to covariate/label shift. That perspective was then further developed [4, 23, 35, 61]. Numerous domain adaptation methods rely on learning invariant representations, and minimize various metrics on the marginal feature distributions: total variation or equivalently D_{JS} [22, 36, 53, 64], maximum mean discrepancy [27, 37–40], Wasserstein distance [15–17, 33, 51], etc. Other noteworthy DA methods use reconstruction losses and cycle-consistency to learn transferable classifiers [29, 56, 67]. Recently, Liu et al. [36] have introduced Transferable Adversarial Training (TAT), where transferable examples are generated to fill the gap in feature space between source and target domains, the datasets is then augmented with those samples. Applying our method to TAT is a future research direction.

Other relevant settings include partial ADA, i.e. UDA when target labels are a strict subset of the source labels / some components of $\mathbf{w}$ are 0 [11–13]. Multi-domain adaptation, where multiple source or target domains are given, is also very studied [18, 28, 42, 45, 47, 63]. Recently, Binkowski et al. [8] study sample reweighting in the domain transfer to handle mass shifts between distributions.

Prior work on combining importance weight in domain-invariant representation learning also exists in the setting of partial DA [60]. However, the importance ratio in these works is defined over the features Z, rather than the class label Y. Compared to our method, this is both statistically inefficient and computationally expensive, since the feature space $\mathcal{Z}$ is often a high-dimensional continuous space, whereas the label space $\mathcal{Y}$ only contains a finite number (k) of distinct labels. In a separate work, Yan et al. [58] proposed a weighted MMD distance to handle target shift in UDA. However, their weights are estimated based on pseudo-labels obtained from the learned classifier, hence it is not clear whether the pseudo-labels provide accurate estimation of the importance weights even in simple settings. As a comparison, under GLS, we show that our weight estimation by solving a quadratic program converges asymptotically.

6 Conclusion and Future Work

We have introduced the generalized label shift assumption, GLS, and theoretically-grounded variations of existing algorithms to handle mismatched label distributions. On tasks from classic benchmarks as well as artificial ones, our algorithms consistently outperform their base versions. The gains, as expected theoretically, correlate well with the JSD between label distributions across domains. In real-world applications, the JSD is unknown, and might be larger than in ML datasets where classes are often purposely balanced. Being simple to implement and adding barely any computational cost, the robustness of our method to mismatched label distributions makes it very relevant to such applications.

Extensions The framework we define in this paper relies on appropriately reweighting the domain adversarial losses. It can be straightforwardly applied to settings where multiple source and/or target domains are used, by simply maintaining one importance weights vector $\mathbf{w}$ for each source/target pair [47, 63]. In particular, label shift could explain the observation from Zhao et al. [63] that too many source domains hurt performance, and our framework might alleviate the issue. One can also think of settings (e.g. semi-supervised domain adaptation) where estimations of $\mathcal{D}_T^Y$ can be obtained via other means. A more challenging but also more interesting future direction is to extend our framework to *domain generalization*, where the learner has access to multiple labeled source domains but no access to (even unlabelled) data from the target domain.

Acknowledgements

The authors thank Romain Laroche and Alessandro Sordoni for useful feedback and helpful discussions. HZ and GG would like to acknowledge support from the DARPA XAI project, contract #FA87501720152 and a Nvidia GPU grant. YW would like acknowledge partial support from NSF Award #2029626, a start-up grant from UCSB Department of Computer Science, as well as generous gifts from Amazon, Adobe, Google and NEC Labs.

Broader Impact

Our work focuses on domain adaptation and attempts to properly handle mismatches in the label distributions between the source and target domains. Domain Adaptation as a whole aims at transferring knowledge gained from a certain domain (or data distribution) to another one. It can potentially be used in a variety of decision making systems, such as spam filters, machine translation, etc.. One can also potentially think of much more sensitive applications such as recidivism prediction, or loan approvals.

While it is unclear to us to what extent DA is currently applied, or how it will be applied in the future, the bias formalized in Th. 2.1 and verified in Table 17 demonstrates that imbalances between classes will result in poor transfer performance of standard ADA methods on a subset of them, which is without a doubt a source of potential inequalities. Our method is actually aimed at counter-balancing the effect of such imbalances. As shown in our empirical results (for instance Table 18) it is rather successful at it, especially on significant shifts. This makes us rather confident in the algorithm's ability to mitigate potential effects of biases in the datasets. On the downside, failure in the weight estimation of some classes might result in poor performance on those. However, we have not observed, in any of our experiments, our method performing significantly worse than its base version. Finally, our method is a variation over existing deep learning algorithms. As such, it carries with it the uncertainties associated to deep learning models, in particular a lack of interpretability and of formal convergence guarantees.

References

[1] Tameem Adel, Han Zhao, and Alexander Wong. Unsupervised domain adaptation with a relaxed covariate shift assumption. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[2] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2019. URL `http://arxiv.org/abs/1907.02893`. cite arxiv:1907.02893.

[3] Jordan T Ash, Robert E Schapire, and Barbara E Engelhardt. Unsupervised domain adaptation using approximate label matching. *arXiv preprint arXiv:1602.04889*, 2016.

[4] Kamyar Azizzadenesheli, Anqi Liu, Fanny Yang, and Animashree Anandkumar. Regularized learning for domain adaptation under label shifts. In *ICLR (Poster)*. OpenReview.net, 2019. URL `http://dblp.uni-trier.de/db/conf/iclr/iclr2019.html#Azizzadenesheli19`.

[5] Philip Bachman, R. Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *CoRR*, abs/1906.00910, 2019. URL `http://dblp.uni-trier.de/db/journals/corr/corr1906.html#abs-1906-00910`.

[6] Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137, 2007.

[7] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79 (1-2):151–175, 2010.

[8] Mikolaj Binkowski, R. Devon Hjelm, and Aaron C. Courville. Batch weight for domain adaptation with mass shift. *CoRR*, abs/1905.12760, 2019. URL `http://dblp.uni-trier.de/db/journals/corr/corr1905.html#abs-1905-12760`.

[9] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *Advances in Neural Information Processing Systems*, pages 343–351, 2016.

[10] Jop Briët and Peter Harremoës. Properties of classical and quantum jensen-shannon divergence. *Phys. Rev. A*, 79:052311, May 2009. doi: 10.1103/PhysRevA.79.052311. URL https://link.aps.org/doi/10.1103/PhysRevA.79.052311.

[11] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Partial transfer learning with selective adversarial networks. In *CVPR*, pages 2724–2732. IEEE Computer Society, 2018. URL http://dblp.uni-trier.de/db/conf/cvpr/cvpr2018.html#CaoL0J18.

[12] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *ECCV (8)*, volume 11212 of *Lecture Notes in Computer Science*, pages 139–155. Springer, 2018. ISBN 978-3-030-01237-3. URL http://dblp.uni-trier.de/db/conf/eccv/eccv2018-8.html#CaoMLW18.

[13] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *CVPR*, pages 2985–2994. Computer Vision Foundation / IEEE, 2019. URL http://dblp.uni-trier.de/db/conf/cvpr/cvpr2019.html#CaoYLW019.

[14] Yee Seng Chan and Hwee Tou Ng. Word sense disambiguation with distribution estimation. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI*, pages 1010–1015. Professional Book Center, 2005. ISBN 0938075934. URL http://dblp.uni-trier.de/db/conf/ijcai/ijcai2005.html#ChanN05.

[15] Qingchao Chen, Yang Liu, Zhaowen Wang, Ian Wassell, and Kevin Chetty. Re-weighted adversarial adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7976–7985, 2018.

[16] Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 3730–3739, 2017.

[17] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9): 1853–1865, 2017.

[18] Hal Daumé III. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*, 2009.

[19] Dua Dheeru and Efi Karra. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

[20] Marthinus Christoffel du Plessis and Masashi Sugiyama. Semi-supervised learning of class balance under class-prior change by distribution matching. *Neural Networks*, 50:110–119, 2014. URL http://dblp.uni-trier.de/db/journals/nn/nn50.html#PlessisS14.

[21] Dominik Maria Endres and Johannes E Schindelin. A new metric for probability distributions. *IEEE Transactions on Information theory*, 2003.

[22] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

[23] Mingming Gong, Kun Zhang, Tongliang Liu, Dacheng Tao, Clark Glymour, and Bernhard Schölkopf. Domain adaptation with conditional transferable components. In *International conference on machine learning*, pages 2839–2848, 2016.

[24] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. 2017. ISBN 9780262035613 0262035618. URL https://www.worldcat.org/title/deep-learning/oclc/985397543&referer=brief_results.

[25] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL http://arxiv.org/abs/1406.2661. cite arxiv:1406.2661.

[26] Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

[27] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[28] Jiang Guo, Darsh J Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. *arXiv preprint arXiv:1809.02256*, 2018.

[29] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017.

[30] Jiayuan Huang, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pages 601–608, 2006.

[31] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. ISSN 0018-9219. doi: 10.1109/5.726791.

[32] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010. URL `http://yann.lecun.com/exdb/mnist/`.

[33] Jaeho Lee and Maxim Raginsky. Minimax statistical learning with wasserstein distances. In *Advances in Neural Information Processing Systems*, pages 2692–2701, 2018.

[34] Jianhua Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[35] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International Conference on Machine Learning*, pages 3128–3136, 2018.

[36] Hong Liu, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 4013–4022. PMLR, 2019. URL `http://dblp.uni-trier.de/db/conf/icml/icml2019.html#LiuLWJ19`.

[37] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1410–1417, 2014.

[38] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015.

[39] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016.

[40] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2208–2217. JMLR, 2017.

[41] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *NeurIPS*, pages 1647–1657, 2018. URL `http://dblp.uni-trier.de/db/conf/nips/nips2018.html#LongC0J18`.

[42] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in neural information processing systems*, pages 1041–1048, 2009.

[43] R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *Proceedings of the ACL*, 2019.

[44] Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29(2):429–443, 1997.

[45] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[46] Tuan Duong Nguyen, Marthinus Christoffel du Plessis, and Masashi Sugiyama. Continuous target shift adaptation in supervised learning. In *ACML*, volume 45 of *JMLR Workshop and Conference Proceedings*, pages 285–300. JMLR.org, 2015. URL http://dblp.uni-trier.de/db/conf/acml/acml2015.html#NguyenPS15.

[47] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 5102–5112. PMLR, 2019. URL http://dblp.uni-trier.de/db/conf/icml/icml2019.html#PengHSS19.

[48] Ievgen Redko, Nicolas Courty, Rémi Flamary, and Devis Tuia. Optimal transport for multi-source domain adaptation under target shift. In *22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019*, volume 89, 2019.

[49] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *ECCV (4)*, volume 6314 of *Lecture Notes in Computer Science*, pages 213–226. Springer, 2010. ISBN 978-3-642-15560-4. URL http://dblp.uni-trier.de/db/conf/eccv/eccv2010-4.html#SaenkoKFD10.

[50] Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris M. Mooij. On causal and anticausal learning. In *ICML*. icml.cc / Omnipress, 2012. URL http://dblp.uni-trier.de/db/conf/icml/icml2012.html#ScholkopfJPSZM12.

[51] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[52] Amos Storkey. When training and test sets are different: Characterising learning transfer. *Dataset shift in machine learning.*, 2009.

[53] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017.

[54] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *(IEEE) Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[55] Visda. Visual domain adaptation challenge, 2017. URL http://ai.bu.edu/visda-2017/.

[56] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. In Jennifer G. Dy and Andreas Krause, editors, *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 5419–5428. PMLR, 2018. URL http://dblp.uni-trier.de/db/conf/icml/icml2018.html#XieZCC18.

[57] Yadollah Yaghoobzadeh, Remi Tachet des Combes, Timothy J. Hazen, and Alessandro Sordoni. Robust natural language inference models with example forgetting. *CoRR*, abs/1911.03861, 2019. URL http://dblp.uni-trier.de/db/journals/corr/corr1911.html#abs-1911-03861.

[58] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2272–2281, 2017.

[59] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[60] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8156–8164, 2018.

[61] Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. In *International Conference on Machine Learning*, pages 819–827, 2013.

[62] Xu Zhang, Felix X. Yu, Shih-Fu Chang, and Shengjin Wang. Deep transfer network: Unsupervised domain adaptation. *CoRR*, abs/1503.00591, 2015. URL http://dblp.uni-trier.de/db/journals/corr/corr1503.html#ZhangYCW15.

[63] Han Zhao, Shanghang Zhang, Guanhang Wu, Geoffrey J Gordon, et al. Multiple source domain adaptation with adversarial learning. In *International Conference on Learning Representations*, 2018.

[64] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. In *Advances in Neural Information Processing Systems*, pages 8568–8579, 2018.

[65] Han Zhao, Junjie Hu, Zhenyao Zhu, Adam Coates, and Geoff Gordon. Deep generative and discriminative domain adaptation. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2315–2317. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[66] Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoffrey J. Gordon. On learning invariant representations for domain adaptation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 7523–7532. PMLR, 2019. URL http://dblp.uni-trier.de/db/conf/icml/icml2019.html#0002CZG19.

[67] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2242–2251. IEEE Computer Society, 2017. ISBN 978-1-5386-1032-9. URL http://dblp.uni-trier.de/db/conf/iccv/iccv2017.html#ZhuPIE17.

A Omitted Proofs

In this section, we provide the theoretical material that completes the main text.

A.1 Definition

Definition A.1. Let us recall that for two distributions $\mathcal{D}$ and $\mathcal{D}'$, the Jensen-Shannon (JSD) divergence $D_{\text{JS}}(\mathcal{D} \parallel \mathcal{D}')$ is defined as:

$$D_{\text{JS}}(\mathcal{D} \parallel \mathcal{D}') := \frac{1}{2} D_{\text{KL}}(\mathcal{D} \parallel \mathcal{D}_M) + \frac{1}{2} D_{\text{KL}}(\mathcal{D}' \parallel \mathcal{D}_M),$$

where $D_{\text{KL}}(\cdot \parallel \cdot)$ is the Kullback–Leibler (KL) divergence and $\mathcal{D}_M := (\mathcal{D} + \mathcal{D}')/2$.

A.2 Consistency of the Weighted Domain Adaptation Loss (7)

For the sake of conciseness, we verify here that the domain adaptation training objective does lead to minimizing the Jensen-Shannon divergence between the weighted feature distribution of the source domain and the feature distribution of the target domain.

Lemma A.1. Let $p(x, y)$ and $q(x)$ be two density distributions, and $w(y)$ be a positive function such that $\int p(y)w(y)dy = 1$. Let $p^w(x) = \int p(x, y)w(y)dy$ denote the w-reweighted marginal distribution of x under p. The minimum value of

$$I(d) := \mathbb{E}_{(x,y)\sim p, x'\sim q}[-w(y)\log(d(x)) - \log(1 - d(x'))]$$

is $\log(4) - 2D_{\text{JS}}(p^w(x) \parallel q(x))$, and is attained for $d^*(x) = \frac{p^w(x)}{p^w(x)+q(x)}$.

Proof. We see that:

$$I(d) = -\iiint [w(y)\log(d(x)) + \log(1 - d(x'))]p(x, y)q(x')dxdx'dy \tag{9}$$

$$= -\int [\int w(y)p(x, y)dy]\log(d(x)) + q(x)\log(1 - d(x))dx \tag{10}$$

$$= -\int p^w(x)\log(d(x)) + q(x)\log(1 - d(x))dx. \tag{11}$$

From the last line, we follow the exact method from Goodfellow et al. [25] to see that point-wise in x the minimum is attained for $d^*(x) = \frac{p^w(x)}{p^w(x)+q(x)}$ and that $I(d^*) = \log(4) - 2D_{\text{JS}}(p^w(x) \parallel q(x))$. ■

Applying Lemma A.1 to $\mathcal{D}_S(Z, Y)$ and $\mathcal{D}_T(Z)$ proves that the domain adaptation objective leads to minimizing $D_{\text{JS}}(\mathcal{D}_S^w(Z) \parallel \mathcal{D}_T(Z))$.

A.3 k-class information-theoretic lower bound

In this section, we prove Theorem 2.1 that extends previous result to the general k-class classification problem.

Theorem 2.1. Assuming that $D_{\text{JS}}(\mathcal{D}_S^Y \parallel \mathcal{D}_T^Y) \geq D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}} \parallel \mathcal{D}_T^{\widetilde{Z}})$, then:

$$\varepsilon_S(h \circ g) + \varepsilon_T(h \circ g) \geq \frac{1}{2}\left(\sqrt{D_{\text{JS}}(\mathcal{D}_S^Y \parallel \mathcal{D}_T^Y)} - \sqrt{D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}} \parallel \mathcal{D}_T^{\widetilde{Z}})}\right)^2.$$

Proof. We essentially follow the proof from Zhao et al. [66], except for Lemmas 4.6 that needs to be adapted to the CDAN framework and Lemma 4.7 to k-class classification.

Lemma 4.6 from Zhao et al. [66] states that $D_{\text{JS}}(\mathcal{D}_S^{\widehat{Y}}, \mathcal{D}_T^{\widehat{Y}}) \leq D_{\text{JS}}(\mathcal{D}_S^Z, \mathcal{D}_T^Z)$, which covers the case $\widetilde{Z} = Z$.

When $\widetilde{Z} = \widehat{Y} \otimes Z$, let us first recall that we assume h or equivalently $\widehat{Y}$ to be a one-hot prediction of the class. We have the following Markov chain:

$$X \xrightarrow{g} Z \xrightarrow{\tilde{h}} \widetilde{Z} \xrightarrow{l} \widehat{Y},$$

where $\tilde{h}(z) = h(z) \otimes z$ and $l : \mathcal{Y} \otimes \mathcal{Z} \to \mathcal{Y}$ returns the index of the non-zero block in $\tilde{h}(z)$. There is only one such block since h is a one-hot, and its index corresponds to the class predicted by h. Given the definition of l, we clearly see that $\widehat{Y}$ is independent of X knowing $\widetilde{Z}$. We can now apply the same proof than in Zhao et al. [66] to conclude that:

$$D_{\text{JS}}(\mathcal{D}_S^{\widehat{Y}}, \mathcal{D}_T^{\widehat{Y}}) \leq D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}}, \mathcal{D}_T^{\widetilde{Z}}). \tag{12}$$

It essentially boils down to a data-processing argument: the discrimination distance between two distributions cannot increase after the same (possibly stochastic) channel (kernel) is applied to both. Here, the channel corresponds to the (potentially randomized) function l.

Remark Additionally, we note that the above inequality holds for *any* $\tilde{Z}$ such that $\widehat{Y} = l(\widetilde{Z})$ for a (potentially randomized) function l. This covers any and all potential combinations of representations at various layers of the deep net, including the last layer (which corresponds to its predictions $\widehat{Y}$).

Let us move to the second part of the proof. We wish to show that $D_{\text{JS}}(\mathcal{D}^Y, \mathcal{D}^{\widehat{Y}}) \leq \varepsilon(h \circ g)$, where $\mathcal{D}$ can be either $\mathcal{D}_S$ or $\mathcal{D}_T$:

$$\begin{aligned}
2D_{\text{JS}}(\mathcal{D}^Y, \mathcal{D}^{\widehat{Y}}) &\leq \|\mathcal{D}^Y - \mathcal{D}^{\widehat{Y}}\|_1 && \text{[34]} \\
&= \sum_{i=1}^{k} |\mathcal{D}(\widehat{Y} = i) - \mathcal{D}(Y = i)| \\
&= \sum_{i=1}^{k} |\sum_{j=1}^{k} \mathcal{D}(\widehat{Y} = i | Y = j)\mathcal{D}(Y = j) - \mathcal{D}(Y = i)| \\
&= \sum_{i=1}^{k} |\mathcal{D}(\widehat{Y} = i | Y = i)\mathcal{D}(Y = i) - \mathcal{D}(Y = i) + \sum_{j \neq i} \mathcal{D}(\widehat{Y} = i | Y = j)\mathcal{D}(Y = j)| \\
&\leq \sum_{i=1}^{k} |\mathcal{D}(\widehat{Y} = i | Y = i) - 1|\mathcal{D}(Y = i) + \sum_{i=1}^{k} \sum_{j \neq i} \mathcal{D}(\widehat{Y} = i | Y = j)\mathcal{D}(Y = j) \\
&= \sum_{i=1}^{k} \mathcal{D}(\widehat{Y} \neq Y | Y = i)\mathcal{D}(Y = i) + \sum_{j=1}^{k} \sum_{i \neq j} \mathcal{D}(\widehat{Y} = i | Y = j)\mathcal{D}(Y = j) \\
&= 2\sum_{i=1}^{k} \mathcal{D}(\widehat{Y} \neq Y | Y = i)\mathcal{D}(Y = i) = 2\mathcal{D}(\widehat{Y} \neq Y) = 2\varepsilon(h \circ g). && (13)
\end{aligned}$$

We can now apply the triangular inequality to $\sqrt{D_{\text{JS}}}$, which is a distance metric [21], called the Jensen-Shannon distance. This gives us:

$$\begin{aligned}
\sqrt{D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)} &\leq \sqrt{D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_S^{\widehat{Y}})} + \sqrt{D_{\text{JS}}(\mathcal{D}_S^{\widehat{Y}}, \mathcal{D}_T^{\widehat{Y}})} + \sqrt{D_{\text{JS}}(\mathcal{D}_T^{\widehat{Y}}, \mathcal{D}_T^Y)} \\
&\leq \sqrt{D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_S^{\widehat{Y}})} + \sqrt{D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}}, \mathcal{D}_T^{\widetilde{Z}})} + \sqrt{D_{\text{JS}}(\mathcal{D}_T^{\widehat{Y}}, \mathcal{D}_T^Y)} \\
&\leq \sqrt{\varepsilon_S(h \circ g)} + \sqrt{D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}}, \mathcal{D}_T^{\widetilde{Z}})} + \sqrt{\varepsilon_T(h \circ g)}.
\end{aligned}$$

where we used Equation (12) for the second inequality and (13) for the third.

Finally, assuming that $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y) \geq D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}}, \mathcal{D}_T^{\widetilde{Z}})$, we get:

$$\left(\sqrt{D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)} - \sqrt{D_{\text{JS}}(\mathcal{D}_S^{\widetilde{Z}}, \mathcal{D}_T^{\widetilde{Z}})}\right)^2 \leq \left(\sqrt{\varepsilon_S(h \circ g)} + \sqrt{\varepsilon_T(h \circ g)}\right)^2 \leq 2\left(\varepsilon_S(h \circ g) + \varepsilon_T(h \circ g)\right).$$

which concludes the proof. ■

A.4 Proof of Theorem 3.1

To simplify the notation, we define the error gap $\Delta_\varepsilon(\widehat{Y})$ as follows:

$$\Delta_\varepsilon(\widehat{Y}) := |\varepsilon_S(\widehat{Y}) - \varepsilon_T(\widehat{Y})|.$$

Also, in this case we use $\mathcal{D}_a,\ a \in \{S,T\}$ to mean the source and target distributions respectively. Before we give the proof of Theorem 3.1, we first prove the following two lemmas that will be used in the proof.

Lemma A.2. Define $\gamma_{a,j} := \mathcal{D}_a(Y=j), \forall a \in \{S,T\}, \forall j \in [k]$, then $\forall \alpha_j, \beta_j \geq 0$ such that $\alpha_j + \beta_j = 1$, and $\forall i \neq j$, the following upper bound holds:

$$|\gamma_{S,j}\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \gamma_{T,j}\mathcal{D}_T(\widehat{Y}=i \mid Y=j)| \leq$$
$$|\gamma_{S,j} - \gamma_{T,j}| \cdot \Big(\alpha_j\mathcal{D}_S(\widehat{Y}=i \mid Y=j) + \beta_j\mathcal{D}_T(\widehat{Y}=i \mid Y=j)\Big) + \gamma_{S,j}\beta_j\Delta_{\text{CE}}(\widehat{Y}) + \gamma_{T,j}\alpha_j\Delta_{\text{CE}}(\widehat{Y}).$$

Proof. To make the derivation uncluttered, define $\mathcal{D}_j(\widehat{Y}=i) := \alpha_j\mathcal{D}_S(\widehat{Y}=i \mid Y=j) + \beta_j\mathcal{D}_T(\widehat{Y}=i \mid Y=j)$ to be the mixture conditional probability of $\widehat{Y}=i$ given $Y=j$, where the mixture weight is given by α_j and β_j. Then in order to prove the upper bound in the lemma, it suffices if we give the desired upper bound for the following term

$$\begin{aligned}
&\Big||\gamma_{S,j}\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \gamma_{T,j}\mathcal{D}_T(\widehat{Y}=i \mid Y=j)| - |(\gamma_{S,j}-\gamma_{T,j})\mathcal{D}_j(\widehat{Y}=i)|\Big| \\
&\leq \Big|\Big(\gamma_{S,j}\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \gamma_{T,j}\mathcal{D}_T(\widehat{Y}=i \mid Y=j)\Big) - (\gamma_{S,j}-\gamma_{T,j})\mathcal{D}_j(\widehat{Y}=i)\Big| \\
&= \Big|\gamma_{S,j}(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)) - \gamma_{T,j}(\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i))\Big|,
\end{aligned}$$

following which we will have:

$$\begin{aligned}
&|\gamma_{S,j}\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \gamma_{T,j}\mathcal{D}_T(\widehat{Y}=i \mid Y=j)| \leq |(\gamma_{S,j}-\gamma_{T,j})\mathcal{D}_j(\widehat{Y}=i)| \\
&+ \Big|\gamma_{S,j}(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)) - \gamma_{T,j}(\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i))\Big| \\
&\leq |\gamma_{S,j}-\gamma_{T,j}|\Big(\alpha_j\mathcal{D}_S(\widehat{Y}=i \mid Y=j) + \beta_j\mathcal{D}_T(\widehat{Y}=i \mid Y=j)\Big) \\
&+ \gamma_{S,j}\Big|\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)\Big| + \gamma_{T,j}\Big|\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)\Big|.
\end{aligned}$$

To proceed, let us first simplify $\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)$. By definition of $\mathcal{D}_j(\widehat{Y}=i) = \alpha_j\mathcal{D}_S(\widehat{Y}=i \mid Y=j) + \beta_j\mathcal{D}_T(\widehat{Y}=i \mid Y=j)$, we know that:

$$\begin{aligned}
&\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i) \\
&= \mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \big(\alpha_j\mathcal{D}_S(\widehat{Y}=i \mid Y=j) + \beta_j\mathcal{D}_T(\widehat{Y}=i \mid Y=j)\big) \\
&= \big(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \alpha_j\mathcal{D}_S(\widehat{Y}=i \mid Y=j)\big) - \beta_j\mathcal{D}_T(\widehat{Y}=i \mid Y=j) \\
&= \beta_j\big(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_T(\widehat{Y}=i \mid Y=j)\big).
\end{aligned}$$

Similarly, for the second term $\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)$, we can show that:

$$\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i) = \alpha_j\big(\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_S(\widehat{Y}=i \mid Y=j)\big).$$

Plugging these two identities into the above, we can continue the analysis with

$$\begin{aligned}
&\Big|\gamma_{S,j}(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i)) - \gamma_{T,j}(\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_j(\widehat{Y}=i))\Big| \\
&= \Big|\gamma_{S,j}\beta(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_T(\widehat{Y}=i \mid Y=j)) - \gamma_{T,j}\alpha_j(\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_S(\widehat{Y}=i \mid Y=j))\Big| \\
&\leq \Big|\gamma_{S,j}\beta_j(\mathcal{D}_S(\widehat{Y}=i \mid Y=j) - \mathcal{D}_T(\widehat{Y}=i \mid Y=j))\Big| + \Big|\gamma_{T,j}\alpha_j(\mathcal{D}_T(\widehat{Y}=i \mid Y=j) - \mathcal{D}_S(\widehat{Y}=i \mid Y=j))\Big| \\
&\leq \gamma_{S,j}\beta_j\Delta_{\text{CE}}(\widehat{Y}) + \gamma_{T,j}\alpha_j\Delta_{\text{CE}}(\widehat{Y}).
\end{aligned}$$

The first inequality holds by the triangle inequality and the second by the definition of the conditional error gap. Combining all the inequalities above completes the proof. ■

We are now ready to prove the theorem:

Theorem 3.1. (Error Decomposition Theorem) For any classifier $\widehat{Y} = (h \circ g)(X)$,

$$|\varepsilon_S(h \circ g) - \varepsilon_T(h \circ g)| \leq \|\mathcal{D}_S^Y - \mathcal{D}_T^Y\|_1 \cdot \mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}),$$

where $\|\mathcal{D}_S^Y - \mathcal{D}_T^Y\|_1 := \sum_{i=1}^k |\mathcal{D}_S(Y=i) - \mathcal{D}_T(Y=i)|$ is the L_1 distance between $\mathcal{D}_S^Y$ and $\mathcal{D}_T^Y$.

Proof of Theorem 3.1. First, by the law of total probability, it is easy to verify that following identity holds for $a \in \{S, T\}$:

$$\mathcal{D}_a(\widehat{Y} \neq Y) = \sum_{i \neq j} \mathcal{D}_a(\widehat{Y} = i, Y = j) = \sum_{i \neq j} \gamma_{a,j} \mathcal{D}_a(\widehat{Y} = i \mid Y = j).$$

Using this identity, to bound the error gap, we have:

$$\begin{aligned}
&|\mathcal{D}_S(Y \neq \widehat{Y}) - \mathcal{D}_T(Y \neq \widehat{Y})| \\
&= \Big| \sum_{i \neq j} \gamma_{S,j} \mathcal{D}_S(\widehat{Y} = i \mid Y = j) - \sum_{i \neq j} \gamma_{T,j} \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \Big| \\
&\leq \sum_{i \neq j} \big| \gamma_{S,j} \mathcal{D}_S(\widehat{Y} = i \mid Y = j) - \gamma_{T,j} \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \big|.
\end{aligned}$$

Invoking Lemma A.2 to bound the above terms, and since $\forall j \in [k], \gamma_{S,j}, \gamma_{T,j} \in [0,1], \alpha_j + \beta_j = 1$, we get:

$$\begin{aligned}
&|\mathcal{D}_S(Y \neq \widehat{Y}) - \mathcal{D}_T(Y \neq \widehat{Y})| \\
&\leq \sum_{i \neq j} \big| \gamma_{S,j} \mathcal{D}_S(\widehat{Y} = i \mid Y = j) - \gamma_{T,j} \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \big| \\
&\leq \sum_{i \neq j} |\gamma_{S,j} - \gamma_{T,j}| \cdot \Big(\alpha_j \mathcal{D}_S(\widehat{Y} = i \mid Y = j) + \beta_j \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \Big) + \gamma_{S,j} \beta_j \Delta_{\mathrm{CE}}(\widehat{Y}) + \gamma_{T,j} \alpha_j \Delta_{\mathrm{CE}}(\widehat{Y}) \\
&\leq \sum_{i \neq j} |\gamma_{S,j} - \gamma_{T,j}| \cdot \Big(\alpha_j \mathcal{D}_S(\widehat{Y} = i \mid Y = j) + \beta_j \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \Big) + \gamma_{S,j} \Delta_{\mathrm{CE}}(\widehat{Y}) + \gamma_{T,j} \Delta_{\mathrm{CE}}(\widehat{Y}) \\
&= \sum_{i \neq j} |\gamma_{S,j} - \gamma_{T,j}| \cdot \Big(\alpha_j \mathcal{D}_S(\widehat{Y} = i \mid Y = j) + \beta_j \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \Big) + \sum_{i=1}^k \sum_{j \neq i} \gamma_{S,j} \Delta_{\mathrm{CE}}(\widehat{Y}) + \gamma_{T,j} \Delta_{\mathrm{CE}}(\widehat{Y}) \\
&= \sum_{i \neq j} |\gamma_{S,j} - \gamma_{T,j}| \cdot \Big(\alpha_j \mathcal{D}_S(\widehat{Y} = i \mid Y = j) + \beta_j \mathcal{D}_T(\widehat{Y} = i \mid Y = j) \Big) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}).
\end{aligned}$$

Note that the above holds $\forall \alpha_j, \beta_j \geq 0$ such that $\alpha_j + \beta_j = 1$. By choosing $\alpha_j = 1, \forall j \in [k]$ and $\beta_j = 0, \forall j \in [k]$, we have:

$$\begin{aligned}
&= \sum_{i \neq j} |\gamma_{S,j} - \gamma_{T,j}| \cdot \mathcal{D}_S(\widehat{Y} = i \mid Y = j) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}) \\
&= \sum_{j=1}^k |\gamma_{S,j} - \gamma_{T,j}| \cdot \left(\sum_{i=1, i \neq j}^k \mathcal{D}_S(\widehat{Y} = i \mid Y = j) \right) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}) \\
&= \sum_{j=1}^k |\gamma_{S,j} - \gamma_{T,j}| \cdot \mathcal{D}_S(\widehat{Y} \neq Y \mid Y = j) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}) \\
&\leq \|\mathcal{D}_S^Y - \mathcal{D}_T^Y\|_1 \cdot \mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y) + 2(k-1)\Delta_{\mathrm{CE}}(\widehat{Y}),
\end{aligned}$$

where the last line is due to Holder's inequality, completing the proof. ■

A.5 Proof of Theorem 3.2

Theorem 3.2. If $Z = g(X)$ satisfies *GLS*, then for any $h : \mathcal{Z} \to \mathcal{Y}$ and letting $\widehat{Y} = h(Z)$ be the predictor, we have $\varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) \leq 2\mathrm{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y)$.

Proof. First, by the law of total probability, we have:

$$\begin{aligned}\varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) &= \mathcal{D}_S(Y \neq \widehat{Y}) + \mathcal{D}_T(Y \neq \widehat{Y}) \\ &= \sum_{j=1}^{k}\sum_{i\neq j} \mathcal{D}_S(\widehat{Y} = i|Y = j)\mathcal{D}_S(Y = j) + \mathcal{D}_T(\widehat{Y} = i|Y = j)\mathcal{D}_T(Y = j).\end{aligned}$$

Now, since $\widehat{Y} = (h \circ g)(X) = h(Z)$, $\widehat{Y}$ is a function of Z. Given the generalized label shift assumption, this guarantees that:

$$\forall y, y' \in \mathcal{Y}, \quad \mathcal{D}_S(\widehat{Y} = y' \mid Y = y) = \mathcal{D}_T(\widehat{Y} = y' \mid Y = y).$$

Thus:

$$\begin{aligned}\varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) &= \sum_{j=1}^{k}\sum_{i\neq j} \mathcal{D}_S(\widehat{Y} = i|Y = j)(\mathcal{D}_S(Y = j) + \mathcal{D}_T(Y = j)) \\ &= \sum_{j\in[k]} \mathcal{D}_S(\widehat{Y} \neq Y \mid Y = j) \cdot (\mathcal{D}_S(Y = j) + \mathcal{D}_T(Y = j)) \\ &\leq \max_{j\in[k]} \mathcal{D}_S(\widehat{Y} \neq Y \mid Y = j) \cdot \sum_{j\in[k]} \mathcal{D}_S(Y = j) + \mathcal{D}_T(Y = j) \\ &= 2\text{BER}_{\mathcal{D}_S}(\widehat{Y} \parallel Y).\end{aligned}$$

■

A.6 Proof of Lemma 3.1

Lemma 3.1. (Necessary condition for *GLS*) If $Z = g(X)$ satisfies *GLS*, then $\mathcal{D}_T(\widetilde{Z}) = \sum_{y\in\mathcal{Y}} \mathbf{w}_y \cdot \mathcal{D}_S(\widetilde{Z}, Y = y) =: \mathcal{D}_S^{\mathbf{w}}(\widetilde{Z})$ where $\widetilde{Z}$ verifies either $\widetilde{Z} = Z$ or $\widetilde{Z} = \widehat{Y} \otimes Z$.

Proof. From *GLS*, we know that $\mathcal{D}_S(Z \mid Y = y) = \mathcal{D}_T(Z \mid Y = y)$. Applying any function $\tilde{h}$ to Z will maintain that equality (in particular $\tilde{h}(Z) = \tilde{Y} \otimes Z$). Using that fact and Eq. (4) on the second line gives:

$$\begin{aligned}\mathcal{D}_T(\tilde{Z}) &= \sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y = y) \cdot \mathcal{D}_T(\tilde{Z} \mid Y = y) \\ &= \sum_{y\in\mathcal{Y}} \mathbf{w}_y \cdot \mathcal{D}_S(Y = y) \cdot \mathcal{D}_S(\tilde{Z} \mid Y = y) \\ &= \sum_{y\in\mathcal{Y}} \mathbf{w}_y \cdot \mathcal{D}_S(\tilde{Z}, Y = y).\end{aligned}$$

■

A.7 Proof of Theorem 3.3

Theorem 3.3. (Clustering structure implies sufficiency) Let $Z = g(X)$ such that $\mathcal{D}_T(Z) = \mathcal{D}_S^{\mathbf{w}}(Z)$. Assume $\mathcal{D}_T(Y = y) > 0, \forall y \in \mathcal{Y}$. If there exists a partition of $\mathcal{Z} = \cup_{y\in\mathcal{Y}} \mathcal{Z}_y$ such that $\forall y \in \mathcal{Y}$, $\mathcal{D}_S(Z \in \mathcal{Z}_y \mid Y = y) = \mathcal{D}_T(Z \in \mathcal{Z}_y \mid Y = y) = 1$, then $Z = g(X)$ satisfies *GLS*.

Proof. Follow the condition that $\mathcal{D}_T(Z) = \mathcal{D}_S^{\mathbf{w}}(Z)$, by definition of $\mathcal{D}_S^{\mathbf{w}}(Z)$, we have:

$$\begin{aligned}&\mathcal{D}_T(Z) = \sum_{y\in\mathcal{Y}} \frac{\mathcal{D}_T(Y = y)}{\mathcal{D}_S(Y = y)} \mathcal{D}_S(Z, Y = y) \\ \iff &\mathcal{D}_T(Z) = \sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y = y)\mathcal{D}_S(Z \mid Y = y) \\ \iff &\sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y = y)\mathcal{D}_T(Z \mid Y = y) = \sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y = y)\mathcal{D}_S(Z \mid Y = y).\end{aligned}$$

Note that the above equation holds for all measurable subsets of $\mathcal{Z}$. Now by the assumption that $\mathcal{Z} = \cup_{y\in\mathcal{Y}}\mathcal{Z}_y$ is a partition of $\mathcal{Z}$, consider $\mathcal{Z}_{y'}$:

$$\sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y=y)\mathcal{D}_T(Z\in\mathcal{Z}_{y'} \mid Y=y) = \sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y=y)\mathcal{D}_S(Z\in\mathcal{Z}_{y'} \mid Y=y).$$

Due to the assumption $\mathcal{D}_S(Z\in\mathcal{Z}_y \mid Y=y) = \mathcal{D}_T(Z\in\mathcal{Z}_y \mid Y=y) = 1$, we know that $\forall y'\neq y$, $\mathcal{D}_T(Z\in\mathcal{Z}_{y'} \mid Y=y) = \mathcal{D}_S(Z\in\mathcal{Z}_{y'} \mid Y=y) = 0$. This shows that both the supports of $\mathcal{D}_S(Z \mid Y=y)$ and $\mathcal{D}_T(Z \mid Y=y)$ are contained in $\mathcal{Z}_y$. Now consider an arbitrary measurable set $E\subseteq\mathcal{Z}_y$, since $\cup_{y\in\mathcal{Y}}\mathcal{Z}_y$ is a partition of $\mathcal{Z}$, we know that

$$\mathcal{D}_S(Z\in E \mid Y=y') = \mathcal{D}_T(Z\in E \mid Y=y') = 0, \quad \forall y'\neq y.$$

Plug $Z\in E$ into the following identity:

$$\begin{aligned}
&\sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y=y)\mathcal{D}_T(Z\in E \mid Y=y) = \sum_{y\in\mathcal{Y}} \mathcal{D}_T(Y=y)\mathcal{D}_S(Z\in E \mid Y=y)\\
\implies &\mathcal{D}_T(Y=y)\mathcal{D}_T(Z\in E \mid Y=y) = \mathcal{D}_T(Y=y)\mathcal{D}_S(Z\in E \mid Y=y)\\
\implies &\mathcal{D}_T(Z\in E \mid Y=y) = \mathcal{D}_S(Z\in E \mid Y=y),
\end{aligned}$$

where the last line holds because $\mathcal{D}_T(Y=y)\neq 0$. Realize that the choice of E is arbitrary, this shows that $\mathcal{D}_S(Z \mid Y=y) = \mathcal{D}_T(Z \mid Y=y)$, which completes the proof. ■

A.8 Sufficient Conditions for *GLS*

Theorem 3.4. Let $\widehat{Y} = h(Z)$, $\gamma := \min_{y\in\mathcal{Y}} \mathcal{D}_T(Y=y)$ and $\mathbf{w}_M := \max_{y\in\mathcal{Y}} \mathbf{w}_y$. For $\widetilde{Z} = Z$ or $\widetilde{Z} = \widehat{Y}\otimes Z$, we have:

$$\max_{y\in\mathcal{Y}} d_{\mathrm{TV}}(\mathcal{D}_S(Z \mid Y=y), \mathcal{D}_T(Z \mid Y=y)) \leq \frac{\mathbf{w}_M\varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) + \sqrt{8D_{\mathrm{JS}}(\mathcal{D}_S^{\mathbf{w}}(\widetilde{Z})\|\mathcal{D}_T(\widetilde{Z}))}}{\gamma}.$$

Proof. To prove the above upper bound, let us first fix a $y\in\mathcal{Y}$ and fix a classifier $\widehat{Y} = h(Z)$ for some $h:\mathcal{Z}\to\mathcal{Y}$. Now consider any measurable subset $E\subseteq\mathcal{Z}$, we would like to upper bound the following quantity:

$$\begin{aligned}
&|\mathcal{D}_S(Z\in E \mid Y=y) - \mathcal{D}_T(Z\in E \mid Y=y)|\\
&\qquad = \frac{1}{\mathcal{D}_T(Y=y)}\cdot|\mathcal{D}_S(Z\in E, Y=y)\mathbf{w}_y - \mathcal{D}_T(Z\in E, Y=y)|\\
&\qquad \leq \frac{1}{\gamma}\cdot|\mathcal{D}_S(Z\in E, Y=y)\mathbf{w}_y - \mathcal{D}_T(Z\in E, Y=y)|.
\end{aligned}$$

Hence it suffices if we can upper bound $|\mathcal{D}_S(Z\in E, Y=y)\mathbf{w}_y - \mathcal{D}_T(Z\in E, Y=y)|$. To do so, consider the following decomposition:

$$\begin{aligned}
|\mathcal{D}_T(Z\in E, Y=y) - \mathcal{D}_S(Z\in E, Y=y)\mathbf{w}_y| &= |\mathcal{D}_T(Z\in E, Y=y) - \mathcal{D}_T(Z\in E, \widehat{Y}=y)\\
&\quad + \mathcal{D}_T(Z\in E, \widehat{Y}=y) - \mathcal{D}_S^{\mathbf{w}}(Z\in E, \widehat{Y}=y)\\
&\quad + \mathcal{D}_S^{\mathbf{w}}(Z\in E, \widehat{Y}=y) - \mathcal{D}_S(Z\in E, Y=y)\mathbf{w}_y|\\
&\leq |\mathcal{D}_T(Z\in E, Y=y) - \mathcal{D}_T(Z\in E, \widehat{Y}=y)|\\
&\quad + |\mathcal{D}_T(Z\in E, \widehat{Y}=y) - \mathcal{D}_S^{\mathbf{w}}(Z\in E, \widehat{Y}=y)|\\
&\quad + |\mathcal{D}_S^{\mathbf{w}}(Z\in E, \widehat{Y}=y) - \mathcal{D}_S(Z\in E, Y=y)\mathbf{w}_y|.
\end{aligned}$$

We bound the above three terms in turn. First, consider $|\mathcal{D}_T(Z \in E, Y = y) - \mathcal{D}_T(Z \in E, \widehat{Y} = y)|$:

$$\begin{aligned}
&|\mathcal{D}_T(Z \in E, Y = y) - \mathcal{D}_T(Z \in E, \widehat{Y} = y)| \\
&\qquad = |\sum_{y'} \mathcal{D}_T(Z \in E, Y = y, \widehat{Y} = y') - \sum_{y'} \mathcal{D}_T(Z \in E, \widehat{Y} = y, Y = y')| \\
&\qquad \leq \sum_{y' \neq y} |\mathcal{D}_T(Z \in E, Y = y, \widehat{Y} = y') - \mathcal{D}_T(Z \in E, \widehat{Y} = y, Y = y')| \\
&\qquad \leq \sum_{y' \neq y} \mathcal{D}_T(Z \in E, Y = y, \widehat{Y} = y') + \mathcal{D}_T(Z \in E, \widehat{Y} = y, Y = y') \\
&\qquad \leq \sum_{y' \neq y} \mathcal{D}_T(Y = y, \widehat{Y} = y') + \mathcal{D}_T(\widehat{Y} = y, Y = y') \\
&\qquad \leq \mathcal{D}_T(Y \neq \widehat{Y}) \\
&\qquad = \varepsilon_T(\widehat{Y}),
\end{aligned}$$

where the last inequality is due to the fact that the definition of error rate corresponds to the sum of all the off-diagonal elements in the confusion matrix while the sum here only corresponds to the sum of all the elements in two slices. Similarly, we can bound the third term as follows:

$$\begin{aligned}
&|\mathcal{D}_S^{\mathbf{w}}(Z \in E, \widehat{Y} = y) - \mathcal{D}_S(Z \in E, Y = y)\mathbf{w}_y| \\
&\qquad = |\sum_{y'} \mathcal{D}_S(Z \in E, \widehat{Y} = y, Y = y')\mathbf{w}_{y'} - \sum_{y'} \mathcal{D}_S(Z \in E, \widehat{Y} = y', Y = y)\mathbf{w}_y| \\
&\qquad \leq |\sum_{y' \neq y} \mathcal{D}_S(Z \in E, \widehat{Y} = y, Y = y')\mathbf{w}_{y'} - \mathcal{D}_S(Z \in E, \widehat{Y} = y', Y = y)\mathbf{w}_y| \\
&\qquad \leq \mathbf{w}_M \sum_{y' \neq y} \mathcal{D}_S(Z \in E, \widehat{Y} = y, Y = y') + \mathcal{D}_S(Z \in E, \widehat{Y} = y', Y = y) \\
&\qquad \leq \mathbf{w}_M \mathcal{D}_S(Z \in E, \widehat{Y} \neq Y) \\
&\qquad \leq \mathbf{w}_M \varepsilon_S(\widehat{Y}).
\end{aligned}$$

Now we bound the last term. Recall the definition of total variation, we have:

$$\begin{aligned}
&|\mathcal{D}_T(Z \in E, \widehat{Y} = y) - \mathcal{D}_S^{\mathbf{w}}(Z \in E, \widehat{Y} = y)| \\
&\qquad = |\mathcal{D}_T(Z \in E \wedge Z \in \widehat{Y}^{-1}(y)) - \mathcal{D}_S^{\mathbf{w}}(Z \in E \wedge Z \in \widehat{Y}^{-1}(y))| \\
&\qquad \leq \sup_{E' \text{ is measurable}} |\mathcal{D}_T(Z \in E') - \mathcal{D}_S^{\mathbf{w}}(Z \in E')| \\
&\qquad = d_{\mathrm{TV}}(\mathcal{D}_T(Z), \mathcal{D}_S^{\mathbf{w}}(Z)).
\end{aligned}$$

Combining the above three parts yields

$$|\mathcal{D}_S(Z \in E \mid Y = y) - \mathcal{D}_T(Z \in E \mid Y = y)| \leq \frac{1}{\gamma} \cdot \left(\mathbf{w}_M \varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) + d_{\mathrm{TV}}(\mathcal{D}_S^{\mathbf{w}}(Z), \mathcal{D}_T(Z)) \right).$$

Now realizing that the choice of $y \in \mathcal{Y}$ and the measurable subset E on the LHS is arbitrary, this leads to

$$\begin{aligned}
&\max_{y \in \mathcal{Y}} \sup_{E} |\mathcal{D}_S(Z \in E \mid Y = y) - \mathcal{D}_T(Z \in E \mid Y = y)| \\
&\qquad\qquad \leq \frac{1}{\gamma} \cdot \left(\mathbf{w}_M \varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) + d_{\mathrm{TV}}(\mathcal{D}_S^{\mathbf{w}}(Z), \mathcal{D}_T(Z)) \right).
\end{aligned}$$

From Briët and Harremoës [10], we have:

$$d_{\mathrm{TV}}(\mathcal{D}_S^{\mathbf{w}}(Z), \mathcal{D}_T(Z)) \leq \sqrt{8 D_{\mathrm{JS}}(\mathcal{D}_S^{\mathbf{w}}(Z) \,||\, \mathcal{D}_T(Z))}$$

(the total variation and Jensen-Shannon distance are equivalent), which gives the results for $\tilde{Z} = Z$. Finally, noticing that $z \to h(z) \otimes z$ is a bijection ($h(z)$ sums to 1), we have:

$$D_{\mathrm{JS}}(\mathcal{D}_S^{\mathbf{w}}(Z) \,||\, \mathcal{D}_T(Z)) = D_{\mathrm{JS}}(\mathcal{D}_S^{\mathbf{w}}(\widehat{Y} \otimes Z) \,||\, \mathcal{D}_T(\widehat{Y} \otimes Z)),$$

which completes the proof. ■

Furthermore, since the above upper bound holds for any classifier $\widehat{Y} = h(Z)$, we even have:

$$\max_{y \in \mathcal{Y}} d_{\mathrm{TV}}(\mathcal{D}_S(Z \in E \mid Y = y), \mathcal{D}_T(Z \in E \mid Y = y)) \\ \leq \frac{1}{\gamma} \cdot \inf_{\widehat{Y}} \left(\mathbf{w}_M \varepsilon_S(\widehat{Y}) + \varepsilon_T(\widehat{Y}) + d_{\mathrm{TV}}(\mathcal{D}_S^{\mathbf{w}}(Z), \mathcal{D}_T(Z)) \right).$$

A.9 Proof of Lemma 3.2

Lemma 3.2. If *GLS* is verified, and if the confusion matrix $\mathbf{C}$ is invertible, then $\mathbf{w} = \mathbf{C}^{-1}\boldsymbol{\mu}$.

Proof. Given (2), and with the joint hypothesis $\widehat{Y} = h(Z)$ over both source and target domains, it is straightforward to see that the induced conditional distributions over predicted labels match between the source and target domains, i.e.:

$$\mathcal{D}_S(\widehat{Y} = h(Z) \mid Y = y) = \mathcal{D}_T(\widehat{Y} = h(Z) \mid Y = y), \ \forall y \in \mathcal{Y}. \tag{14}$$

This allows us to compute $\boldsymbol{\mu}_y, \ \forall y \in \mathcal{Y}$ as

$$\begin{aligned} \mathcal{D}_T(\widehat{Y} = y) &= \sum_{y' \in \mathcal{Y}} \mathcal{D}_T(\widehat{Y} = y \mid Y = y') \cdot \mathcal{D}_T(Y = y') \\ &= \sum_{y' \in \mathcal{Y}} \mathcal{D}_S(\widehat{Y} = y \mid Y = y') \cdot \mathcal{D}_T(Y = y') \\ &= \sum_{y' \in \mathcal{Y}} \mathcal{D}_S(\widehat{Y} = y, Y = y') \cdot \frac{\mathcal{D}_T(Y = y')}{\mathcal{D}_S(Y = y')} \\ &= \sum_{y' \in \mathcal{Y}} \mathbf{C}_{y,y'} \cdot \mathbf{w}_{y'}. \end{aligned}$$

where we used (14) for the second line. We thus have $\boldsymbol{\mu} = \mathbf{C}\mathbf{w}$ which concludes the proof. ■

A.10 $\mathcal{F}$-IPM for Distributional Alignment

In Table 4, we list different instances of IPM with different choices of the function class $\mathcal{F}$ in the above definition, including the total variation distance, Wasserstein-1 distance and the Maximum mean discrepancy [27].

Table 4: List of IPMs with different $\mathcal{F}$. $\|\cdot\|_{\mathrm{Lip}}$ denotes the Lipschitz seminorm and $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS).

$\mathcal{F}$	$d_{\mathcal{F}}$
$\{f : \|f\|_\infty \leq 1\}$	Total Variation
$\{f : \|f\|_{\mathrm{Lip}} \leq 1\}$	Wasserstein-1 distance
$\{f : \|f\|_{\mathcal{H}} \leq 1\}$	Maximum mean discrepancy

B Experimentation Details

B.1 Computational Complexity

Our algorithms imply negligible time and memory overhead compared to their base versions. They are, in practice, indistinguishable from the underlying baseline:

- Weight estimation requires storing the confusion matrix C and the predictions μ. This has a memory cost of $O(k^2)$, small compared to the size of a neural network that performs well on k classes.

- The extra computational cost comes from solving the quadratic program 5, which only depends on the number of classes k and is solved once per epoch (not per gradient step). For Office-Home, it is a 65×65 QP, solved ≈ 100 times. Its runtime is negligible compared to tens of thousands of gradient steps.

B.2 Description of the domain adaptation tasks

Digits We follow a widely used evaluation protocol [29, 41]. For the digits datasets MNIST (M, LeCun and Cortes [32]) and USPS (U, Dheeru and Karra [19]), we consider the DA tasks: M $\rightarrow$ U and U $\rightarrow$ M. Performance is evaluated on the 10,000/2,007 examples of the MNIST/USPS test sets.

Visda [55] is a sim-to-real domain adaptation task. The synthetic domain contains 2D rendering of 3D models captured at different angles and lighting conditions. The real domain is made of natural images. Overall, the training, validation and test domains contain 152,397, 55,388 and 5,534 images, from 12 different classes.

Office-31 [49] is one of the most popular dataset for domain adaptation . It contains 4,652 images from 31 classes. The samples come from three domains: Amazon (A), DSLR (D) and Webcam (W), which generate six possible transfer tasks, A $\rightarrow$ D, A $\rightarrow$ W, D $\rightarrow$ A, D $\rightarrow$ W, W $\rightarrow$ A and W $\rightarrow$ D, which we all evaluate.

Office-Home [54] is a more complex dataset than Office-31. It consists of 15,500 images from 65 classes depicting objects in office and home environments. The images form four different domains: Artistic (A), Clipart (C), Product (P), and Real-World images (R). We evaluate the 12 possible domain adaptation tasks.

B.3 Full results on the domain adaptation tasks

Tables 5, 6, 7, 8, 9 and 10 show the detailed results of all the algorithms on each task of the domains described above. The performance we report is the best test accuracy obtained during training over a fixed number of epochs. We used that value for fairness with respect to the baselines (as shown in Figure 2 Left, the performance of DANN decreases as training progresses, due to the inappropriate matching of representations showcased in Theorem 2.1).

The subscript denotes the fraction of seeds for which our variant outperforms the base algorithm. More precisely, by outperform, we mean that for a given seed (which fixes the network initialization as well as the data being fed to the model) the variant has a larger accuracy on the test set than its base version. Doing so allows to assess specifically the effect of the algorithm, all else kept constant.

Table 5: Results on the Digits tasks. M and U stand for MNIST and USPS, the prefix s denotes the experiment where the source domain is subsampled to increase $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$.

METHOD	M $\rightarrow$ U	U $\rightarrow$ M	AVG.	sM $\rightarrow$ U	sU $\rightarrow$ M	AVG.
NO AD.	79.04	75.30	77.17	76.02	75.32	75.67
DANN	90.65	95.66	93.15	79.03	87.46	83.24
IWDAN	$\mathbf{93.28}_{100\%}$	$\mathbf{96.52}_{100\%}$	$\mathbf{94.90}_{100\%}$	$\mathbf{91.77}_{100\%}$	$\mathbf{93.32}_{100\%}$	$\mathbf{92.54}_{100\%}$
IWDAN-O	$93.73_{100\%}$	$96.81_{100\%}$	$95.27_{100\%}$	$92.50_{100\%}$	$96.42_{100\%}$	$94.46_{100\%}$
CDAN	94.16	97.29	95.72	84.91	91.55	88.23
IWCDAN	$\mathbf{94.36}_{60\%}$	$\mathbf{97.45}_{100\%}$	$\mathbf{95.90}_{80\%}$	$\mathbf{93.42}_{100\%}$	$\mathbf{93.03}_{100\%}$	$\mathbf{93.22}_{100\%}$
IWCDAN-O	$94.34_{80\%}$	$97.35_{100\%}$	$95.85_{90\%}$	$93.37_{100\%}$	$96.26_{100\%}$	$94.81_{100\%}$

B.4 Jensen-Shannon divergence of the original and subsampled domain adaptation datasets

Tables 11, 12 and 13 show $D_{\text{JS}}(\mathcal{D}_S(Z)||\mathcal{D}_T(Z))$ for our four datasets and their subsampled versions, rows correspond to the source domain, and columns to the target one. We recall that subsampling simply consists in taking 30% of the first half of the classes in the source domain (which explains why $D_{\text{JS}}(\mathcal{D}_S(Z)||\mathcal{D}_T(Z))$ is not symmetric for the subsampled datasets).

Table 6: Results on the Visda domain. The prefix s denotes the experiment where the source domain is subsampled to increase $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$.

METHOD	VISDA	sVISDA
NO AD.	48.39	49.02
DANN	61.88	52.85
IWDAN	$\mathbf{63.52}_{100\%}$	$\mathbf{60.18}_{100\%}$
IWDAN-O	$64.19_{100\%}$	$62.10_{100\%}$
CDAN	65.60	60.19
IWCDAN	$\mathbf{66.49}_{60\%}$	$\mathbf{65.83}_{100\%}$
IWCDAN-O	$68.15_{100\%}$	$66.85_{100\%}$
JAN	$56.98_{100\%}$	$50.64_{100\%}$
IWJAN	$\mathbf{57.56}_{100\%}$	$\mathbf{57.12}_{100\%}$
IWJAN-O	$61.48_{100\%}$	$61.30_{100\%}$

Table 7: Results on the Office dataset.

METHOD	A → D	A → W	D → A	D → W	W → A	W → D	AVG.
NO DA	79.60	73.18	59.33	96.30	58.75	99.68	77.81
DANN	84.06	85.41	64.67	96.08	66.77	99.44	82.74
IWDAN	$\mathbf{84.30}_{60\%}$	$\mathbf{86.42}_{100\%}$	$\mathbf{68.38}_{100\%}$	$\mathbf{97.13}_{100\%}$	$\mathbf{67.16}_{60\%}$	$\mathbf{100.0}_{100\%}$	$\mathbf{83.90}_{87\%}$
IWDAN-O	$87.23_{100\%}$	$88.88_{100\%}$	$69.92_{100\%}$	$98.09_{100\%}$	$67.96_{80\%}$	$99.92_{100\%}$	$85.33_{97\%}$
CDAN	**89.56**	93.01	71.25	99.24	70.32	100.0	87.23
IWCDAN	$88.91_{60\%}$	$\mathbf{93.23}_{60\%}$	$\mathbf{71.90}_{80\%}$	$\mathbf{99.30}_{80\%}$	$\mathbf{70.43}_{60\%}$	$\mathbf{100.0}_{100\%}$	$\mathbf{87.30}_{73\%}$
IWCDAN-O	$90.08_{60\%}$	$94.52_{100\%}$	$73.11_{100\%}$	$99.30_{80\%}$	$71.83_{100\%}$	$100.0_{100\%}$	$88.14_{90\%}$
JAN	85.94	**85.66**	**70.50**	97.48	**71.5**	99.72	85.13
IWJAN	$\mathbf{87.68}_{100\%}$	$84.86_{0\%}$	$70.36_{60\%}$	$\mathbf{98.98}_{100\%}$	$70.06_{0\%}$	$\mathbf{100.0}_{100\%}$	$\mathbf{85.32}_{60\%}$
IWJAN-O	$89.68_{100\%}$	$89.18_{100\%}$	$71.96_{100\%}$	$99.02_{100\%}$	$73.0_{100\%}$	$100.0_{100\%}$	$87.14_{100\%}$

B.5 Losses

B.5.1 DANN

For batches of data (x_S^i, y_S^i) and (x_T^i) of size s, the DANN losses are:

$$\mathcal{L}_{DA}(x_S^i, y_S^i, x_T^i; \theta, \psi) = -\frac{1}{s}\sum_{i=1}^{s} \log(d_\psi(g_\theta(x_S^i))) + \log(1 - d_\psi(g_\theta(x_T^i))), \tag{15}$$

$$\mathcal{L}_C(x_S^i, y_S^i; \theta, \phi) = -\frac{1}{s}\sum_{i=1}^{s} \log(h_\phi(g_\theta(x_S^i)_{y_S^i})). \tag{16}$$

B.5.2 CDAN

Similarly, the CDAN losses are:

$$\mathcal{L}_{DA}(x_S^i, y_S^i, x_T^i; \theta, \psi) = -\frac{1}{s}\sum_{i=1}^{s} \log(d_\psi(h_\phi(g_\theta(x_S^i)) \otimes g_\theta(x_S^i))) \tag{17}$$

$$+ \log(1 - d_\psi(h_\phi(g_\theta(x_T^i)) \otimes g_\theta(x_T^i))), \tag{18}$$

$$\mathcal{L}_C(x_S^i, y_S^i; \theta, \phi) = -\frac{1}{s}\sum_{i=1}^{s} \log(h_\phi(g_\theta(x_S^i)_{y_S^i})), \tag{19}$$

where $h_\phi(g_\theta(x_S^i)) \otimes g_\theta(x_S^i) := (h_1(g(x_S^i))g(x_S^i), \ldots, h_k(g(x_S^i))g(x_S^i))$ and $h_1(g(x_S^i))$ is the i-th element of vector $h(g(x_S^i))$.

Table 8: Results on the Subsampled Office dataset.

METHOD	sA → D	sA → W	sD → A	sD → W	sW → A	sW → D	AVG.
NO DA	75.82	70.69	56.82	95.32	58.35	97.31	75.72
DANN	75.46	77.66	56.58	93.76	57.51	96.02	76.17
IWDAN	**81.61**$_{100\%}$	**88.43**$_{100\%}$	**65.00**$_{100\%}$	**96.98**$_{100\%}$	**64.86**$_{100\%}$	**98.72**$_{100\%}$	**82.60**$_{100\%}$
IWDAN-O	84.94$_{100\%}$	91.17$_{100\%}$	68.44$_{100\%}$	97.74$_{100\%}$	64.57$_{100\%}$	99.60$_{100\%}$	84.41$_{100\%}$
CDAN	82.45	84.60	62.54	96.83	65.01	98.31	81.62
IWCDAN	**86.59**$_{100\%}$	**87.30**$_{100\%}$	**66.45**$_{100\%}$	**97.69**$_{100\%}$	**66.34**$_{100\%}$	**98.92**$_{100\%}$	**83.88**$_{100\%}$
IWCDAN-O	87.39$_{100\%}$	91.47$_{100\%}$	69.69$_{100\%}$	97.91$_{100\%}$	67.50$_{100\%}$	98.88$_{100\%}$	85.47$_{100\%}$
JAN	77.74	77.64	64.48	91.68	92.60	65.10	78.21
IWJAN	**84.62**$_{100\%}$	**83.28**$_{100\%}$	**65.30**$_{80\%}$	**96.30**$_{100\%}$	**98.80**$_{100\%}$	**67.38**$_{100\%}$	**82.61**$_{97\%}$
IWJAN-O	88.42$_{100\%}$	89.44$_{100\%}$	72.06$_{100\%}$	97.26$_{100\%}$	98.96$_{100\%}$	71.30$_{100\%}$	86.24$_{100\%}$

Table 9: Results on the Office-Home dataset.

METHOD	A → C	A → P	A → R	C → A	C → P	C → R	
NO DA	41.02	62.97	71.26	48.66	58.86	60.91	
DANN	46.03	62.23	70.57	49.06	63.05	64.14	
IWDAN	**48.65**$_{100\%}$	**69.19**$_{100\%}$	**73.60**$_{100\%}$	**53.59**$_{100\%}$	**66.25**$_{100\%}$	**66.09**$_{100\%}$	
IWDAN-O	50.19$_{100\%}$	70.53$_{100\%}$	75.44$_{100\%}$	56.69$_{100\%}$	67.40$_{100\%}$	67.98$_{100\%}$	
CDAN	49.00	69.23	74.55	54.46	68.23	68.9	
IWCDAN	**49.81**$_{100\%}$	**73.41**$_{100\%}$	**77.56**$_{100\%}$	**56.5**$_{100\%}$	**69.64**$_{80\%}$	**70.33**$_{100\%}$	
IWCDAN-O	52.31$_{100\%}$	74.54$_{100\%}$	78.46$_{100\%}$	60.33$_{100\%}$	70.78$_{100\%}$	71.47$_{100\%}$	
JAN	**41.64**	67.20	73.12	51.02	62.52	64.46	
IWJAN	41.12$_{0\%}$	**67.56**$_{80\%}$	**73.14**$_{60\%}$	**51.70**$_{100\%}$	**63.42**$_{100\%}$	**65.22**$_{100\%}$	
IWJAN-O	41.88$_{80\%}$	68.72$_{100\%}$	73.62$_{100\%}$	53.04$_{100\%}$	63.88$_{100\%}$	66.48$_{100\%}$	
METHOD	P → A	P → C	P → R	R → A	R → C	R → P	AVG.
NO DA	47.1	35.94	68.27	61.79	44.42	75.5	56.39
DANN	48.29	44.06	72.62	63.81	53.93	77.64	59.62
IWDAN	**52.81**$_{100\%}$	**46.24**$_{80\%}$	**73.97**$_{100\%}$	**64.90**$_{100\%}$	**54.02**$_{80\%}$	**77.96**$_{100\%}$	**62.27**$_{97\%}$
IWDAN-O	59.33$_{100\%}$	48.28$_{100\%}$	76.37$_{100\%}$	69.42$_{100\%}$	56.09$_{100\%}$	78.45$_{100\%}$	64.68$_{100\%}$
CDAN	56.77	**48.8**	76.83	**71.27**	**55.72**	**81.27**	64.59
IWCDAN	**58.99**$_{100\%}$	48.41$_{0\%}$	**77.94**$_{100\%}$	69.48$_{0\%}$	54.73$_{0\%}$	81.07$_{60\%}$	**65.66**$_{70\%}$
IWCDAN-O	62.60$_{100\%}$	50.73$_{100\%}$	78.88$_{100\%}$	72.44$_{100\%}$	57.79$_{100\%}$	81.31$_{80\%}$	67.64$_{98\%}$
JAN	54.5	40.36	**73.10**	**64.54**	**45.98**	**76.58**	59.59
IWJAN	**55.26**$_{80\%}$	**40.38**$_{60\%}$	73.08$_{80\%}$	64.40$_{60\%}$	45.68$_{0\%}$	76.36$_{40\%}$	**59.78**$_{63\%}$
IWJAN-O	57.78$_{100\%}$	41.32$_{100\%}$	73.66$_{100\%}$	65.40$_{100\%}$	46.68$_{100\%}$	76.36$_{20\%}$	60.73$_{92\%}$

CDAN is particularly well-suited for conditional alignment. As described in Section 2, the CDAN discriminator seeks to match $\mathcal{D}_S(\widehat{Y} \otimes Z)$ with $\mathcal{D}_T(\widehat{Y} \otimes Z)$. This objective is very aligned with GLS: let us first assume for argument's sake that $\widehat{Y}$ is a perfect classifier on both domains. For any sample (x, y), $\hat{y} \otimes z$ is thus a matrix of 0s except on the y-th row, which contains z. When label distributions match, the effect of fooling the discriminator will result in representations such that the matrices $\widehat{Y} \otimes Z$ are equal on the source and target domains. In other words, the model is such that $Z \mid Y$ match: it verifies GLS (see Th. 3.4 below with $\mathbf{w} = 1$). On the other hand, if the label distributions differ, fooling the discriminator actually requires mislabelling certain samples (a fact quantified in Th. 2.1).

Table 10: Results on the subsampled Office-Home dataset.

METHOD	A → C	A → P	A → R	C → A	C → P	C → R	
NO DA	35.70	54.72	62.61	43.71	52.54	56.62	
DANN	36.14	54.16	61.72	44.33	52.56	56.37	
IWDAN	**39.81**$_{100\%}$	**63.01**$_{100\%}$	**68.67**$_{100\%}$	**47.39**$_{100\%}$	**61.05**$_{100\%}$	**60.44**$_{100\%}$	
IWDAN-O	42.79$_{100\%}$	66.22$_{100\%}$	71.40$_{100\%}$	53.39$_{100\%}$	61.47$_{100\%}$	64.97$_{100\%}$	
CDAN	38.90	56.80	64.77	48.02	60.07	61.17	
IWCDAN	**42.96**$_{100\%}$	**65.01**$_{100\%}$	**71.34**$_{100\%}$	**52.89**$_{100\%}$	**64.65**$_{100\%}$	**66.48**$_{100\%}$	
IWCDAN-O	45.76$_{100\%}$	68.61$_{100\%}$	73.18$_{100\%}$	56.88$_{100\%}$	66.61$_{100\%}$	68.48$_{100\%}$	
JAN	34.52	56.86	64.54	46.18	56.84	59.06	
IWJAN	**36.24**$_{100\%}$	**61.00**$_{100\%}$	**66.34**$_{100\%}$	**48.66**$_{100\%}$	**59.92**$_{100\%}$	**61.88**$_{100\%}$	
IWJAN-O	37.46$_{100\%}$	62.68$_{100\%}$	66.88$_{100\%}$	49.82$_{100\%}$	60.22$_{100\%}$	62.54$_{100\%}$	

METHOD	P → A	P → C	P → R	R → A	R → C	R → P	AVG.
NO DA	44.29	33.05	65.20	57.12	40.46	70.0	
DANN	44.58	37.14	65.21	56.70	43.16	69.86	51.83
IWDAN	**50.44**$_{100\%}$	**41.63**$_{100\%}$	**72.46**$_{100\%}$	**61.00**$_{100\%}$	**49.40**$_{100\%}$	**76.07**$_{100\%}$	**57.61**$_{100\%}$
IWDAN-O	56.05$_{100\%}$	43.39$_{100\%}$	74.87$_{100\%}$	66.73$_{100\%}$	51.72$_{100\%}$	77.46$_{100\%}$	60.87$_{100\%}$
CDAN	49.65	41.36	70.24	62.35	46.98	74.69	56.25
IWCDAN	**54.87**$_{100\%}$	**44.80**$_{100\%}$	**75.91**$_{100\%}$	**67.02**$_{100\%}$	**50.45**$_{100\%}$	**78.55**$_{100\%}$	**61.24**$_{100\%}$
IWCDAN-O	59.63$_{100\%}$	46.98$_{100\%}$	77.54$_{100\%}$	69.24$_{100\%}$	53.77$_{100\%}$	78.11$_{100\%}$	63.73$_{100\%}$
JAN	50.64	37.24	69.98	58.72	40.64	72.00	53.94
IWJAN	**52.92**$_{100\%}$	**37.68**$_{100\%}$	**70.88**$_{100\%}$	**60.32**$_{100\%}$	**41.54**$_{100\%}$	**73.26**$_{100\%}$	**55.89**$_{100\%}$
IWJAN-O	56.54$_{100\%}$	39.66$_{100\%}$	71.78$_{100\%}$	62.36$_{100\%}$	44.56$_{100\%}$	73.76$_{100\%}$	57.36$_{100\%}$

Table 11: Jensen-Shannon divergence between the label distributions of the Digits and Visda tasks.

(A) FULL DATASET

	MNIST	USPS	REAL
MNIST	0	6.64e−3	-
USPS	6.64e−3	0	-
SYNTH.	-	-	2.61e−2

(B) SUBSAMPLED

	MNIST	USPS	REAL
MNIST	0	6.52e−2	-
USPS	2.75e−2	0	-
SYNTH.	-	-	6.81e−2

B.5.3 JAN

The JAN losses [40] are :

$$\mathcal{L}_{DA}(x_S^i, y_S^i, x_T^i; \theta, \psi) = -\frac{1}{s^2}\sum_{i,j=1}^{s} k(x_S^i, x_S^j) - \frac{1}{s^2}\sum_{i,j=1}^{s} k(x_T^i, x_T^j) + \frac{2}{s^2}\sum_{i,j=1}^{s} k(x_S^i, x_T^j) \tag{20}$$

$$\mathcal{L}_C(x_S^i, y_S^i; \theta, \phi) = -\frac{1}{s}\sum_{i=1}^{s} \log(h_\phi(g_\theta(x_S^i)_{y_S^i})), \tag{21}$$

where k corresponds to the kernel of the RKHS $\mathcal{H}$ used to measure the discrepancy between distributions. Exactly as in Long et al. [40], it is the product of kernels on various layers of the network $k(x_S^i, x_S^j) = \prod_{l\in\mathcal{L}} k^l(x_S^i, x_S^j)$. Each individual kernel k^l is computed as the dot-product between

Table 12: Jensen-Shannon divergence between the label distributions of the Office-31 tasks.

(A) FULL DATASET

	AMAZON	DSLR	WEBCAM
AMAZON	0	1.76e−2	9.52e−3
DSLR	1.76e−2	0	2.11e−2
WEBCAM	9.52e−3	2.11e−2	0

(B) SUBSAMPLED

	AMAZON	DSLR	WEBCAM
AMAZON	0	6.25e−2	4.61e−2
DSLR	5.44e−2	0	5.67e−2
WEBCAM	5.15e−2	7.05e−2	0

Table 13: Jensen-Shannon divergence between the label distributions of the Office-Home tasks.

(A) FULL DATASET

	ART	CLIPART	PRODUCT	REAL WORLD
ART	0	3.85e−2	4.49e−2	2.40e−2
CLIPART	3.85e−2	0	2.33e−2	2.14e−2
PRODUCT	4.49e−2	2.33e−2	0	1.61e−2
REAL WORLD	2.40e−2	2.14e−2	1.61e−2	0

(B) SUBSAMPLED

	ART	CLIPART	PRODUCT	REAL WORLD
ART	0	8.41e−2	8.86e−2	6.69e−2
CLIPART	7.07e−2	0	5.86e−2	5.68e−2
PRODUCT	7.85e−2	6.24e−2	0	5.33e−2
REAL WORLD	6.09e−2	6.52e−2	5.77e−2	0

two transformations of the representation: $k^l(x_S^i, x_S^j) = \langle d_\psi^l(g_\theta^l(x_S^i)), d_\psi^l(g_\theta^l(x_S^j))\rangle$ (in this case, d_ψ^l outputs vectors in a high-dimensional space). See Section B.7 for more details.

The IWJAN losses are:

$$\mathcal{L}_{DA}^{\mathbf{w}}(x_S^i, y_S^i, x_T^i; \theta, \psi) = -\frac{1}{s^2}\sum_{i,j=1}^{s} \mathbf{w}_{y_S^i}\mathbf{w}_{y_S^j} k(x_S^i, x_S^j) - \frac{1}{s^2}\sum_{i,j=1}^{s} k(x_T^i, x_T^j) + \frac{2}{s^2}\sum_{i,j=1}^{s} \mathbf{w}_{y_S^i} k(x_S^i, x_T^j) \tag{22}$$

$$\mathcal{L}_{C}^{\mathbf{w}}(x_S^i, y_S^i; \theta, \phi) = -\frac{1}{s}\sum_{i=1}^{s} \frac{\mathbf{w}_{y_S^i}}{k\mathcal{D}_S(Y=y)} \log(h_\phi(g_\theta(x_S^i))_{y_S^i}). \tag{23}$$

B.6 Generation of domain adaptation tasks with varying $D_{\mathbf{JS}}(\mathcal{D}_S(Z) \parallel \mathcal{D}_T(Z))$

We consider the MNIST → USPS task and generate a set $\mathcal{V}$ of 50 vectors in $[0.1, 1]^{10}$. Each vector corresponds to the fraction of each class to be trained on, either in the source or the target domain (to assess the impact of both). The left bound is chosen as 0.1 to ensure that classes all contain some samples.

This methodology creates 100 domain adaptation tasks, 50 for *subsampled*-MNIST → USPS and 50 for MNIST → *subsampled*-USPS, with Jensen-Shannon divergences varying from 6.1e−3 to

9.53e$-$2[2]. They are then used to evaluate our algorithms, see Section 4 and Figures 1 and 3. They show the performance of the 6 algorithms we consider. We see the sharp decrease in performance of the base versions DANN and CDAN. Comparatively, our importance-weighted algorithms maintain good performance even for large divergences between the marginal label distributions.

B.7 Implementation details

All the values reported below are the default ones in the implementations of DANN, CDAN and JAN released with the respective papers (see links to the github repos in the footnotes). We did not perform any search on them, assuming they had already been optimized by the authors of those papers. To ensure a fair comparison and showcase the simplicity of our approach, we simply plugged the weight estimation on top of those baselines and used their original hyperparameters.

For MNIST and USPS, the architecture is akin to LeNet [31], with two convolutional layers, ReLU and MaxPooling, followed by two fully connected layers. The representation is also taken as the last hidden layer, and has 500 neurons. The optimizer for those tasks is SGD with a learning rate of 0.02, annealed by 0.5 every five training epochs for M $\rightarrow$ U and 6 for U $\rightarrow$ M. The weight decay is also 5e$-$4 and the momentum 0.9.

For the Office and Visda experiments with IWDAN and IWCDAN, we train a ResNet-50, optimized using SGD with momentum. The weight decay is also 5e$-$4 and the momentum 0.9. The learning rate is 3e$-$4 for the Office-31 tasks A $\rightarrow$ D and D $\rightarrow$ W, 1e$-$3 otherwise (default learning rates from the CDAN implementation[3]).

For the IWJAN experiments, we use the default implementation of Xlearn codebase[4] and simply add the weigths estimation and reweighted objectives to it, as described in Section B.5. Parameters, configuration and networks remain the same.

Finally, for the Office experiments, we update the importance weights $\mathbf{w}$ every 15 passes on the dataset (in order to improve their estimation on small datasets). On Digits and Visda, the importance weights are updated every pass on the source dataset. Here too, fine-tuning that value might lead to a better estimation of $\mathbf{w}$ and help bridge the gap with the oracle versions of the algorithms.

We use the cvxopt package[5] to solve the quadratic programm 5.

We trained our models on single-GPU machines (P40s and P100s). The runtime of our algorithms is undistinguishable from the the runtime of their base versions.

B.8 Weight Estimation

We estimate importance weights using Lemma 3.2, which relies on the GLS assumption. However, there is no guarantee that GLS is verified at any point during training, so the exact dynamics of $\mathbf{w}$ are unclear. Below we discuss those dynamics and provide some intuition about them.

In Fig. 4b, we plot the Euclidian distance between the moving average of weights estimated using the equation $\mathbf{w} = \mathbf{C}^{-1}\boldsymbol{\mu}$ and the true weights (note that this can be done for any algorithm). As can be seen in the figure, the distance between the estimated and true weights is highly correlated with the performance of the algorithm (see Fig.4). In particular, we see that the estimations for IWDAN is more accurate than for DANN. The estimation for DANN exhibits an interesting shape, improving at first, and then getting worse. At the same time, the estimation for IWDAN improves monotonously. The weights for IWDAN-O get very close to the true weights which is in line with our theoretical results: IWDAN-O gets close to zero error on the target error, Th. 3.4 thus guarantees that GLS is verified, which in turns implies that the weight estimation is accurate (Lemma 3.2). Finally, without domain adaptation, the estimation is very poor. The following two lemmas shed some light on the phenomena observed for DANN and IWDAN:

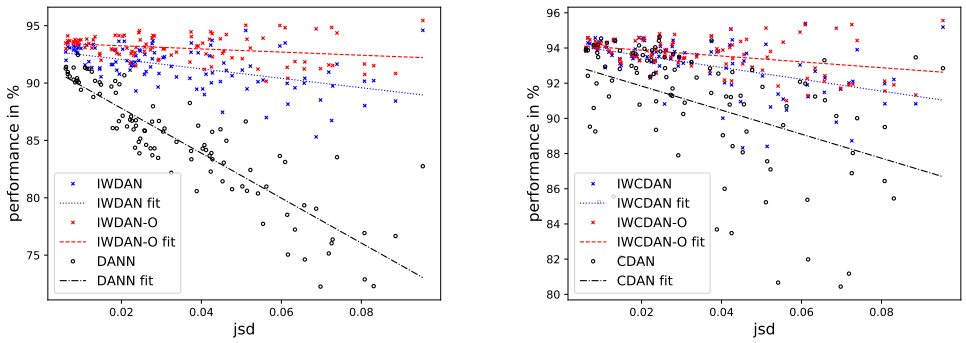
Lemma 3.3. If the source error $\varepsilon_S(h \circ g)$ is zero and the source and target marginals verify $D_{\text{JS}}(\mathcal{D}_S^{\tilde{\mathbf{w}}}(Z), \mathcal{D}_T(Z)) = 0$, then the estimated weight vector $\mathbf{w}$ is equal to $\tilde{\mathbf{w}}$.

[2]We manually rejected some samples to guarantee a rather uniform set of divergences.

[3]https://github.com/thuml/CDAN/tree/master/pytorch

[4]https://github.com/thuml/Xlearn/tree/master/pytorch

[5]http://cvxopt.org/



(a) Performance of DANN, IWDAN and IWDAN-O. (b) Performance of CDAN, CDAN and IWCDAN.

Figure 3: Performance in % of our algorithms and their base versions. The x-axis represents $D_{\text{JS}}(\mathcal{D}_S^Y, \mathcal{D}_T^Y)$, the Jensen-Shannon distance between label distributions. Lines represent linear fits to the data. For both sets of algorithms, the larger the jsd, the larger the improvement.

Proof. If $\varepsilon_S(h \circ g) = 0$, then the confusion matrix $\mathbf{C}$ is diagonal and its y-th line is $\mathcal{D}_S(Y = y)$. Additionally, if $D_{\text{JS}}(\mathcal{D}_S^{\tilde{\mathbf{w}}}(Z), \mathcal{D}_T(Z)) = 0$, then from a straightforward extension of Eq. 12, we have $D_{\text{JS}}(\mathcal{D}_S^{\tilde{\mathbf{w}}}(\hat{Y}), \mathcal{D}_T(\hat{Y})) = 0$. In other words, the distribution of predictions on the source and target domains match, *i.e.* $\boldsymbol{\mu}_y = \mathcal{D}_T(\hat{Y} = y) = \sum_{y'} \tilde{\mathbf{w}}_{y'} \mathcal{D}_S(\hat{Y} = y, Y = y') = \tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \forall y$ (where the last equality comes from $\varepsilon_S(h \circ g) = 0$). Finally, we get that $\mathbf{w} = \mathbf{C}^{-1}\boldsymbol{\mu} = \tilde{\mathbf{w}}$. ■

In particular, applying this lemma to DANN (*i.e.* with $\tilde{\mathbf{w}}_{y'} = \mathbf{1}$) suggests that at convergence, the estimated weights should tend to $\mathbf{1}$. Empirically, Fig. 4b shows that as the marginals get matched, the estimation for DANN does get closer to $\mathbf{1}$ ($\mathbf{1}$ corresponds to a distance of 2.16)[6]. We now attempt to provide some intuition on the behavior of IWDAN, with the following lemma:

Lemma B.1. If $\varepsilon_S(h \circ g) = 0$ and if for a given y:

$$\min(\tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \mathcal{D}_T(Y = y)) \leq \boldsymbol{\mu}_y \leq \max(\tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \mathcal{D}_T(Y = y)), \tag{24}$$

then, letting $\mathbf{w} = \mathbf{C}^{-1}\boldsymbol{\mu}$ be the estimated weight:

$$|\mathbf{w}_y - \mathbf{w}_y^*| \leq |\tilde{\mathbf{w}}_y - \mathbf{w}_y^*|.$$

Applying this lemma to $\tilde{\mathbf{w}}_y = \mathbf{w}_t$, and assuming that (24) holds for all the classes y (we discuss what the assumption implies below), we get that:

$$\|\mathbf{w}_{t+1} - \mathbf{w}_y^*\| \leq \|\mathbf{w}_t - \mathbf{w}_y^*\|, \tag{25}$$

or in other words, the estimation improves monotonously. Combining this with Lemma B.1 suggests an explanation for the shape of the IWDAN estimated weights on Fig. 4b: the monotonous improvement of the estimation is counter-balanced by the matching of weighted marginals which, when reached, makes $\mathbf{w}_t$ constant (Lemma 3.3 applied to $\tilde{\mathbf{w}} = \mathbf{w}_t$). However, we wish to emphasize that the exact dynamics of $\mathbf{w}$ are complex, and we do not claim understand them fully. In all likelihood, they are the by-product of regularity in the data, properties of deep neural networks and their interaction with stochastic gradient descent. Additionally, the dynamics are also inherently linked to the success of domain adaptation, which to this day remains an open problem.

As a matter of fact, assumption (24) itself relates to successful domain adaptation. Setting aside $\tilde{\mathbf{w}}$, which simply corresponds to a class reweighting of the source domain, (24) states that predictions on the target domain fall between a successful prediction (corresponding to $\mathcal{D}_T(Y = y)$) and the prediction of a model with matched marginals (corresponding to $\mathcal{D}_S(Y = y)$). In other words, we

[6] It does not reach it as the learning rate is decayed to 0.

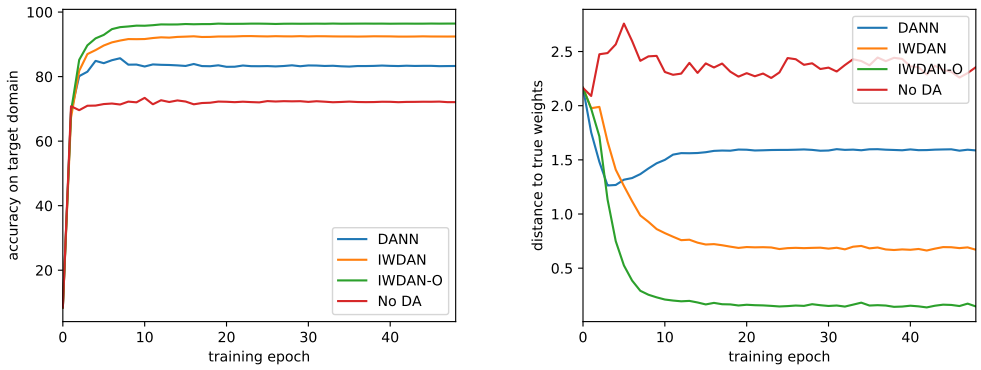
assume that the model is naturally in between successful domain adaptation and successful marginal matching. Empirically, we observed that it holds true for most classes (with $\tilde{\mathbf{w}} = \tilde{\mathbf{w}}_t$ for IWDAN and with $\tilde{\mathbf{w}} = \mathbf{1}$ for DANN), but not all early in training[7].

To conclude this section, we prove Lemma B.1.

Proof. From $\varepsilon_S(h \circ g) = 0$, we know that $\mathbf{C}$ is diagonal and that its y-th line is $\mathcal{D}_S(Y = y)$. This gives us: $\mathbf{w}_y = (\mathbf{C}^{-1}\boldsymbol{\mu})_y = \frac{\boldsymbol{\mu}_y}{\mathcal{D}_S(Y=y)}$. Hence:

$$
\begin{aligned}
& \min(\tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \mathcal{D}_T(Y = y)) \leq \boldsymbol{\mu}_y \leq \max(\tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \mathcal{D}_T(Y = y)) \\
\iff \quad & \frac{\min(\tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \mathcal{D}_T(Y = y))}{\mathcal{D}_S(Y = y)} \leq \frac{\boldsymbol{\mu}_y}{\mathcal{D}_S(Y = y)} \leq \frac{\max(\tilde{\mathbf{w}}_y \mathcal{D}_S(Y = y), \mathcal{D}_T(Y = y))}{\mathcal{D}_S(Y = y)} \\
\iff \quad & \min(\tilde{\mathbf{w}}_y, \mathbf{w}_y^*) \leq \mathbf{w}_y \leq \max(\tilde{\mathbf{w}}_y, \mathbf{w}_y^*) \\
\iff \quad & \min(\tilde{\mathbf{w}}_y, \mathbf{w}_y^*) - \mathbf{w}_y^* \leq \mathbf{w}_y - \mathbf{w}_y^* \leq \max(\tilde{\mathbf{w}}_y, \mathbf{w}_y^*) - \mathbf{w}_y^* \\
\iff \quad & |\mathbf{w}_y - \mathbf{w}_y^*| \leq |\tilde{\mathbf{w}}_y - \mathbf{w}_y^*|,
\end{aligned}
$$

which conludes the proof. ■



(a) Transfer accuracy during training.

(b) Distance to true weights during training.

Figure 4: *Left* Accuracy of various algorithms during training. *Right* Euclidian distance between the weights estimated using Lemma 3.2 and the true weights. Those plots correspond to averages over 5 seeds.

B.9 Per-class predictions and estimated weights

In this section, we display the per-class predictions of various algorithms on the sU $\rightarrow$ M task. In Table 16, we see that without domain adaptation, performance on classes is rather random, the digit 9 for instance is very poorly predicted and confused with 4 and 8.

Table 17 shows an interesting pattern for DANN. In line with the limitations described by Theorem 2.1, the model performs very poorly on the subsampled classes (we recall that the subsampling is done in the source domain): the neural network tries to match the unweighted marginals. To do so, it projects representations of classes that are over-represented in the target domain (digits 0 to 4) on representations of the under-represented classes (digits 5 to 9). In doing so, it heavily degrades its performance on those classes (it is worth noting that digit 0 has an importance weight close to 1 which probably explains why DANN still performs well on it, see Table 14).

[7] In particular at initialization, one class usually dominates the others.

As far as IWDAN is concerned, Table 18 shows that the model perfoms rather well on all classes, at the exception of the digit 7 confused with 9. IWDAN-O is shown in Table 19 and as expected outperforms the other algorithms on all classes.

Finally, Table 14 shows the estimated weights of all the algorithms, at the training epoch displayed in Tables 16, 17, 18 and 19. We see a rather strong correlation between errors on the estimated weight for a given class, and errors in the predictions for that class (see for instance digit 3 for DANN or digit 7 for IWDAN).

Table 14: Estimated weights and their euclidian distance to the true weights, taken at the training epoch for the confusion matrices in Tables 16, 17, 18 and 19. The first row contains the true weights. The last column gives the euclidian distance from the true weights.

	CLASS										
	0	1	2	3	4	5	6	7	8	9	DISTANCE
TRUE	1.19	1.61	1.96	2.24	2.16	0.70	0.64	0.70	0.78	0.66	0
DANN	1.06	1.15	1.66	1.33	1.95	0.86	0.72	0.70	1.02	0.92	1.15
IWDAN	1.19	1.61	1.92	1.96	2.31	0.70	0.63	0.55	0.78	0.78	0.38
IWDAN-O	1.19	1.60	2.01	2.14	2.1	0.73	0.64	0.65	0.78	0.66	0.12
NO DA	1.14	1.4	2.42	1.49	4.21	0.94	0.38	0.82	0.62	0.29	2.31

Table 15: Ablation study on the Digits tasks, with weights learnt during training.

Method	Digits	sDigits	Method	Digits	sDigits
DANN	93.15	83.24	CDAN	95.72	88.23
DANN + $\mathcal{L}_C^{\mathbf{w}}$	93.18	84.20	CDAN + $\mathcal{L}_C^{\mathbf{w}}$	95.30	91.14
DANN + $\mathcal{L}_{DA}^{\mathbf{w}}$	**94.35**	**92.48**	CDAN + $\mathcal{L}_{DA}^{\mathbf{w}}$	95.42	92.35
IWDAN	**94.90**	**92.54**	IWCDAN	**95.90**	**93.22**

Table 16: Per-class predictions without domain adaptation on the sU $\rightarrow$ M task. Average accuracy: 74.49%. The table M below verifies $M_{ij} = \mathcal{D}_T(\hat{Y} = j|Y = i)$.

92.89	0.13	3.24	0.00	2.20	0.01	0.45	0.88	0.18	0.02
0.00	72.54	12.38	0.00	3.40	0.37	7.54	1.50	2.28	0.00
0.31	0.23	93.28	0.09	0.72	0.03	0.34	4.78	0.17	0.05
0.06	0.77	4.81	68.53	1.50	19.91	0.02	2.48	1.61	0.31
0.02	0.62	0.28	0.00	97.19	0.51	0.04	0.17	0.79	0.37
0.75	3.03	0.69	1.01	1.20	88.96	0.39	0.31	2.69	0.96
0.73	1.98	0.42	0.03	23.86	2.74	69.08	0.29	0.12	0.75
1.02	2.01	4.16	0.13	9.32	6.36	0.01	73.48	1.01	2.50
6.01	8.27	2.55	1.35	1.62	3.62	4.98	6.96	64.40	0.24
1.49	3.35	0.55	1.28	38.30	15.36	0.05	20.68	1.34	17.60

Table 17: Per-class predictions for DANN on the sU $\rightarrow$ M task. Average accuracy: 86.71%. The table M below verifies $M_{ij} = \mathcal{D}_T(\hat{Y} = j|Y = i)$. The first 5 classes are under-represented in the source domain compared to the target domain. On those (except 0), DANN does not perform as well as on the over-represented classes (the last 5). In line with Th. 2.1, matching the representation distributions on source and target forced the classifier to confuse the digits "1", "3" and "4" in the target domain with "8", "5" and "9".

95.79	0.01	0.08	0.01	0.12	0.38	2.34	0.36	0.36	0.57
0.14	70.77	0.80	0.01	1.03	1.29	9.46	0.06	16.39	0.06
1.61	0.14	89.82	0.20	0.42	0.48	0.83	3.73	1.37	1.39
0.46	0.08	1.10	63.33	0.04	26.28	0.02	1.78	3.76	3.14
0.11	0.13	0.05	0.00	78.17	0.85	0.16	0.15	1.97	18.41
0.19	0.04	0.02	0.04	0.01	91.30	0.25	0.27	5.97	1.91
0.62	0.12	0.01	0.00	1.98	4.61	91.73	0.05	0.36	0.51
0.14	0.23	1.39	0.13	0.10	0.32	0.02	94.10	1.46	2.10
0.69	0.13	0.11	0.05	0.21	2.12	0.50	0.36	95.19	0.66
0.36	0.31	0.03	0.08	0.46	3.67	0.01	1.64	1.03	92.40

Table 18: Per-class predictions for IWDAN on the sU $\rightarrow$ M task. Average accuracy: 94.38%. The table M below verifies $M_{ij} = \mathcal{D}_T(\hat{Y} = j|Y = i)$.

97.33	0.06	0.23	0.01	0.20	0.43	1.29	0.19	0.18	0.09
0.00	97.71	0.41	0.05	0.56	0.69	0.14	0.03	0.34	0.07
0.70	0.16	96.32	0.08	0.34	0.23	0.23	1.49	0.43	0.01
0.23	0.01	0.97	87.67	0.01	9.25	0.02	0.63	0.87	0.35
0.11	0.25	0.05	0.00	96.93	0.16	0.22	0.02	0.40	1.85
0.15	0.11	0.01	0.16	0.05	95.81	0.69	0.11	2.82	0.08
0.26	0.25	0.00	0.00	2.07	1.49	95.84	0.01	0.07	0.00
0.16	0.42	2.12	0.91	1.15	0.60	0.03	82.07	1.35	11.19
0.44	0.50	0.36	0.18	0.43	0.90	0.91	0.15	95.74	0.40
0.34	0.42	0.06	0.30	1.67	2.46	0.11	0.31	0.85	93.50

Table 19: Per-class predictions for IWDAN-O on the sU $\rightarrow$ M task. Average accuracy: 96.8%. The table M below verifies $M_{ij} = \mathcal{D}_T(\hat{Y} = j | Y = i)$.

98.04	0.01	0.20	0.00	0.27	0.03	1.17	0.11	0.15	0.02
0.00	98.35	0.33	0.15	0.17	0.27	0.19	0.05	0.47	0.02
0.22	0.04	97.48	0.07	0.29	0.08	0.52	1.09	0.19	0.02
0.10	0.00	0.66	95.72	0.01	2.32	0.00	0.35	0.56	0.27
0.01	0.25	0.05	0.00	96.80	0.03	0.18	0.01	0.60	2.06
0.23	0.11	0.00	0.72	0.00	96.09	0.68	0.13	2.01	0.03
0.27	0.31	0.00	0.00	2.07	0.63	96.54	0.00	0.17	0.01
0.26	0.45	2.13	0.29	0.90	0.32	0.01	92.66	1.06	1.92
0.55	0.22	0.30	0.06	0.18	0.22	0.41	0.33	97.11	0.62
0.46	0.37	0.16	0.86	0.82	1.45	0.01	0.77	0.98	94.13