On Target Shift in Adversarial Domain Adaptation

Yitong Li[1], **Michael Murias**[2], **Samantha Major**[3], **Geraldine Dawson**[4], **David E. Carlson**[1,4,5]
[1]Electrical and Computer Engineering, [2]Duke Institute for Brain Sciences, [3]Psychiatry and Behavioral Sciences,
[4]Civil and Environmental Engineering, [5]Biostatistics and Bioinformatics, Duke University
{yitong.li, michael.murias, samantha.major, geraldine.dawson, david.carlson}@duke.edu

Abstract

Discrepancy between training and testing domains is a fundamental problem in the generalization of machine learning techniques. Recently, several approaches have been proposed to learn domain invariant feature representations through adversarial deep learning. However, label shift, where the percentage of data in each class is different between domains, has received less attention. Label shift naturally arises in many contexts, especially in behavioral studies where the behaviors are freely chosen. In this work, we propose a method called Domain Adversarial nets for Target Shift (DATS) to address label shift while learning a domain invariant representation. This is accomplished by using distribution matching to estimate label proportions in a blind test set. We extend this framework to handle multiple domains by developing a scheme to upweight source domains most similar to the target domain. Empirical results show that this framework performs well under large label shift in synthetic and real experiments, demonstrating the practical importance.

1 Introduction

In supervised learning, the goal is to be able to make predictions on newly collected data (the target domain) by training on previously labeled data (the source domain). However, a gap between the source and target domains is often inevitable, due to either the changes in the data, differing data collection processes, or differing applications. Domain adaptation aims to bridge these distribution gaps to enhance generalization [25, 9, 21, 38]. In this manuscript, we focus on unsupervised domain adaptation, where the target samples have no labels available during training. A common approach for this scenario is to match the marginal distribution of the features without using labels [12, 31, 11]. This is motivated by the problem of "covariate shift," where the distribution of features may change, but the relationship between features and the associated outcome is constant.

In order to solve the problem of covariate shift, most existing algorithms implicitly assume that the label proportions remain unchanged [7]. However, a common case in the real world is that the percentage of samples from each class are highly variant between domains. Consider a case where we model patients in a study as separate domains. When data is collected, the label proportions can be drastically different between patients due to many reasons, such as free behavioral choice, missing data, or differing outcomes or progression from a disease. We will show empirically that in such a situation, these existing approaches do not help generalization due to this incorrect assumption. Similar problems also arise in anomaly rejection [28, 36] and remote sensing image classification [33]. This kind of problem is called class-prior change [7] or target shift [26]. If an algorithm cannot account for such a shift, it can be provably suboptimal in deployment, and an overfit classifier can incorrectly remember the label proportions [32]. Previous methods have addressed this problem by adding regularization terms [11, 12, 19]. In this manuscript, we show how the label proportions in the target domain is estimated and appropriately weight samples to correct adversarial domain adaptation methods for target shift.

Additionally, the number of source domains is not limited to one in practice. This necessitates explicitly accounting for multiple sources instead of treating the data as one large source domain. An unfortunate issue in multiple domain adaptation is that adding more domains is not always better. Adding irrelevant (or

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

less relevant) domains can hurt generalization performance [21]. There has been some recent works to address choosing appropriate source domains for use in domain adaptation [38, 26]. In a similar vein, we propose a scheme to weight source domains by how similar they are to the target, allowing the domain adaptation to use only the most relevant information. This weighting can be naturally included with our previous scheme to address label imbalance.

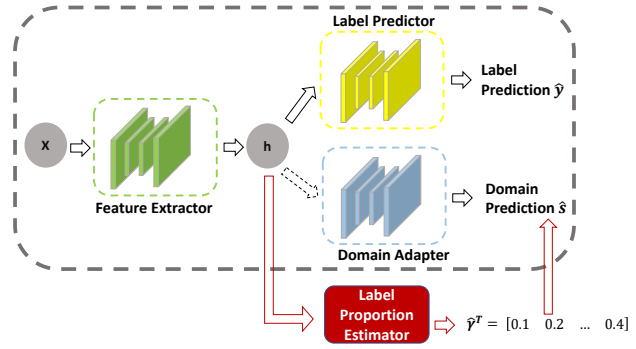
In this work, we propose an approach called Domain Adversarial nets for Target Shift (DATS) to address unsupervised multiple source domain adaptation with target shift. Our model is implemented in a neural network framework. First, we extend an adversarial learning scheme to get domain-invariant features [9] to account for label imbalance. In these extracted features, the target label proportion is estimated by minimizing the marginal distribution gap between source and target after accounting for the known or estimated label proportions. To jointly deal with multiple sources, a weighting vector is learned to determine how much each source domain should be used. This model is trained end-to-end in an iterative way. The proposed model captures strength from related source domains while eliminating the influence from less correlated domains. Experimentally, we demonstrate on real-world data that the proposed model improves performance over numerous baselines in the presence of target shift.

2 Notation, Background, and an Illustrative Problem

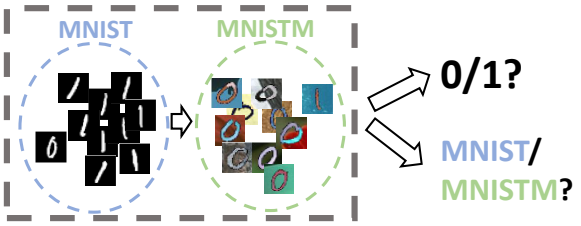
Before introducing the proposed model, the Domain Adversarial Neural Network (DANN) [9] framework is introduced. We will then show a simple example of how this approach does not naturally handle label shift, motivating the extensions to solve these situations.

Assuming the training/source data is given as $\{\boldsymbol{x}_i, y_i, s_i\}$ for $i = 1.\cdots, N_S$, where $\boldsymbol{x}_i$ is the input, y_i is the label with values from $\{1, \cdots, L\}$, and $s_i \in \{1, \cdots, S\}$ indicates which domain the data comes from. For domain $\mathcal{D}_s$, it contains a total of n_s samples and $\sum_{s=1}^{S} n_s = N_S$. S is the total number of source domains. The testing/target samples are given as $\{\boldsymbol{x}_i, s_i = T\}$ for $i = N_S + 1, \cdots, N_S + N_T$ without label y given. $\boldsymbol{h}_i = f(\boldsymbol{x}_i; \boldsymbol{\theta}_h)$ is the encoded feature of $\boldsymbol{x}_i$ generated by the feature extractor. The entries of $\boldsymbol{\gamma}^s \in \Delta^{L-1}$ are the label proportions of domain $\mathcal{D}_s$, which lies on the simplex. The target domain label proportion $\boldsymbol{\gamma}^T$ is unknown, which we will later estimate. The superscript s and T indicate the source and target domain indexes.

Compared to the proposed model DATS, the framework of DANN is given in the gray dotted box in



(a) DATS model framework.



(b) A toy example.

Figure 1: (Top) The framework of the proposed model contains a shared feature extractor, a label predictor, a domain predictor and a label proportion estimator. The domain weighting scheme is not visualized. (Bottom) An unbalanced toy example. The proportion of digit 0 and 1 hugely differs in the two domains.

Figure 1(a) (everything except the red box). The intuition is to learn encoded features that can correctly predict the label while being unable to accurately predict the domain, thereby requiring that the features are domain invariant. DANN contains three components: a feature extractor, a label predictor, and a domain adapter. The black dotted arrow (in contrast to the solid arrows) between the extracted features and the domain adapter marks an adversarial relationship; in other words, that features $\boldsymbol{h}$ are expected to give low domain classification accuracy. All three components are implemented as neural networks. The feature extractor outputs features $\boldsymbol{h} = f(\boldsymbol{x}; \boldsymbol{\theta}_h)$ with parameters denoted as $\boldsymbol{\theta}_h$. $\boldsymbol{h}$ is then input to the label predictor with loss $\mathcal{L}_Y(y, f(\boldsymbol{x}; \boldsymbol{\theta}_h); \boldsymbol{\theta}_Y)$. For a classification task, $\mathcal{L}_Y$ is the cross entropy loss between the predicted and the true label pairs. For regression, this can be the mean squared error. Similarly, the domain adapter has loss $\mathcal{L}_D(s, f(\boldsymbol{x}; \boldsymbol{\theta}_h); \boldsymbol{\theta}_D)$ to predict which domain the data belongs to with the cross-entropy classification loss. This is adversarial as the learned features minimize the discrepancy between different domains while simultaneously maximizing label prediction accuracy. The objective function can be written as

$$\min_{\boldsymbol{\theta}_h, \boldsymbol{\theta}_Y} \max_{\boldsymbol{\theta}_D} \mathbb{E}_{p(\mathbf{x}, y, s)} \left[\mathcal{L}_Y(y, f(\boldsymbol{x}; \boldsymbol{\theta}_h); \boldsymbol{\theta}_Y) - \alpha_D \mathcal{L}_D(s, f(\boldsymbol{x}; \boldsymbol{\theta}_h); \boldsymbol{\theta}_D) \right]. \quad (1)$$

Note that target samples are not included in the first term since the label y is unknown in the target domain. α_D controls the relative strength of the adversary.

DANN assumes that the domain adapter should contain no information about the label by learning features that maximize the domain loss $\mathcal{L}_D$. However, the domain adapter ***must*** contain information about the label under target shift. Consider the example in Figure 1(b) for digit image classification, where we consider domain transfer from the well-known MNIST (source) to MNISTM (target) dataset. MNISTM is a colorized version of the MNIST dataset that is used for demonstrating domain adaptation [9]. Suppose that the source domain contains 10% of digit 1 and 90% of digit 0 while the target domain has 90% of digit 1 and 10% of digit 0. If a label classifier can achieve 100% accuracy, then an optimal domain predictor must be at least 90% accurate. This can be seen because the label itself is 90% accurate for predicting domain, so this information must exist in the feature set. However, the domain classifier in DANN is aims to achieve 50% accuracy, which means that the learned features cannot distinguish between the domains. This contradicts the result from the naive classifier, and enforcing this condition destroys performance, which we detail empirically in Section 5.1.

In order to solve this problem, we propose the DATS model, which estimates the label distribution in the target domain to reweight data samples. This approach follows from the similar idea as balancing classes in logistic regression [11].

3 Domain Adaptation under Target Shift

To address the target shift, a label proportion estimator is proposed. This is visualized as the red box in Figure 1(a). The technique for estimating the target label proportions $\boldsymbol{\gamma}^T$ is introduced in Section 3.1, which is used to reweight data samples in the adversary. The red arrow in Figure 1(a) illustrates the usage of the label weight. The proposed method to weight multiple source domains is introduced in Section 3.2. After that, a distribution matching technique is introduced to further improve the weighting accuracy in Section 3.3. Finally, the complete loss function and pseudo-code is covered in Section 3.4. In the following, the superscript $s = 1, \cdots, S$ is the index of the source domain. For clarity, T means the target domain, while $\intercal$ means vector/matrix transpose. The label is denoted by a subscript $l \in \{1, \cdots, L\}$.

3.1 Label Weighting Scheme

The label proportions in the source domains are known simply by counting examples, with γ_l^s representing the proportion of label l in source $\mathcal{D}_s$. For the target domain, we propose to estimate the proportion of each label over the whole set, rather than estimating the label of each individual sample [3]. Our empirical results demonstrate that this enhances robustness.

A common assumption in target shift is that the conditional distributions from the label to the features are constant, such that $P^s(\boldsymbol{x}|\boldsymbol{y}) = P^T(\boldsymbol{x}|\boldsymbol{y}) = P(\boldsymbol{x}|\boldsymbol{y})$ for $s = 1, \ldots, S$, and the variability in the joint distribution $p(\boldsymbol{x}, \boldsymbol{y})$ is due to the shift in label proportions $p(\boldsymbol{y})$ [23]. Such an assumption is obviously untrue in the raw data for cases such as MNIST to MNISTM (see Figure 1(b), where the color differences in the raw data break this assumption). After correcting for the target shift and with the adversarial framework, the assumption that the feature extractor $\boldsymbol{h} = f(\boldsymbol{x}; \boldsymbol{\theta}_h)$ provides domain-invariant features is much more reasonable, so the assumption $P^s(\boldsymbol{h}|\boldsymbol{y}) = P^T(\boldsymbol{h}|\boldsymbol{y}) = P(\boldsymbol{h}|\boldsymbol{y})$ is better aligned with reality.

This assumption can be used to estimate the label proportions in the target domain via marginal distribution matching [37]; however, unlike previous approaches this estimation proceeds on the extracted feature space. Using known properties from the source domains and the weights on the target domain, we can reweight a source domain by labels to match the target distribution under the assumption. For domain $\mathcal{D}_s$, this weighted distribution is given as

$$Q^s(\boldsymbol{h}) = \textstyle\sum_{l=1}^{L} P^s(\boldsymbol{h}|y = l)\gamma_l^T. \tag{2}$$

If the above assumption holds and $\boldsymbol{\gamma}^T$ is correct, then $Q^s(\boldsymbol{h})$ is identical to the target distribution with $Q^s(\boldsymbol{h}) = P^T(\boldsymbol{h})$. Therefore, one estimation strategy is to estimate $\boldsymbol{\gamma}^T$ to minimize a distance metric $d(Q^s(\boldsymbol{h}), P^T(\boldsymbol{h}))$ by jointly considering all source domains, where $d(\cdot)$ is a distance metric.

In the literature, mean matching has proven to be a simple and effective approach to these types of problems [11, 12]. In contrast to prior work, we will perform mean matching in the extracted feature space rather than in the raw data. Eq. (2) can be estimated by using sample means of the data points by $\boldsymbol{M}^s\boldsymbol{\gamma}^T$, where $\boldsymbol{M}^s = \boldsymbol{M}^s(\boldsymbol{h}|\boldsymbol{y})$ is the concatenation of $[\boldsymbol{\mu}^s(\boldsymbol{h}|y = 1), \boldsymbol{\mu}^s(\boldsymbol{h}|y = 2), \cdots, \boldsymbol{\mu}^s(\boldsymbol{h}|y = L)]$, the empirical sample means from the source domain $\mathcal{D}_s$. The target label proportion $\boldsymbol{\gamma}^T$ is estimated by restricting to the simplex and minimizing the loss function,

$$\mathcal{L}_{r_M}(\boldsymbol{\gamma}^T) = \textstyle\sum_{s=1}^{S} \lambda^s ||\boldsymbol{M}^s\boldsymbol{\gamma}^T - \boldsymbol{\mu}^T||_2^2 \tag{3}$$

λ^s is defined as the domain weight that controls which

source domains are used more (or less) for domain adaptation, described in Section 3.2. $\boldsymbol{\mu}^T$ is the encoded feature mean of the target. The L_2 loss in (3) can be replaced with a distribution loss such as the Wasserstein loss [26] or Maximum Mean Discrepancy loss [11], which we expand upon in Section 3.3. Note that (3) is a standard linearly constrained quadratic problem, yielding estimated target label proportions $\hat{\boldsymbol{\gamma}}^T$. In practice, this is updated by gradient descent in each minibatch.

Given the label proportions, it remains to correct the cross-entropy loss in the domain adversary defined in (1) for the target shift. To do this, define $\beta^s(y=l) = \frac{P^T(y=l)}{P^s(y=l)} = \frac{\gamma_l^T}{\gamma_l^s}$ as an unnormalized probability ratio of the target domain to domain $\mathcal{D}_s$, and $\boldsymbol{\beta}^s$ is the vector form across all labels in domain $\mathcal{D}_s$. $\hat{\gamma}^T$ is plugged in to get an empirical estimate $\hat{\boldsymbol{\beta}}^s$.

By introducing the additional label weight, the domain adapter in Figure 1(a) is mathematically akin to a weighted classifier. The loss function of the domain adapter is given as

$$\mathcal{L}_D(\boldsymbol{\theta}_D, \boldsymbol{\theta}_h) = \underbrace{\sum_{i=1}^{N_S} \frac{\lambda^{s_i} \hat{\beta}_{y_i}^{s_i}}{n_{s_i} ||\hat{\boldsymbol{\beta}}^s||_1} \mathcal{C}\left(\hat{s}_i, s_i; \boldsymbol{\theta}_D, \boldsymbol{\theta}_h\right)}_{\text{Source Samples}} + \underbrace{\frac{1}{LN_T} \sum_{i=N_S+1}^{N_S+N_T} \mathcal{C}\left(\hat{s}_i, s_i; \boldsymbol{\theta}_D, \boldsymbol{\theta}_h\right).}_{\text{Target Samples}}$$

$\mathcal{C}(\cdot)$ is used as the cross-entropy loss between the estimated domain index and the ground truth. The label weight $\boldsymbol{\beta}$ is used for each source domain sample. $\hat{\beta}_{y_i}^{s_i}$ is the estimated label weight for sample $\boldsymbol{x}_i$ in domain s_i with label y_i. λ^s determines the importance of source domain $\mathcal{D}_s$, which will be introduced in the next section. The weighted version of domain loss increases the robustness of the algorithm under target shift.

We note that if the stated assumptions are true, then the proportion estimation scheme is asymptotically consistent. This is stated formally below.

Theorem 3.1. *Assume that $P^s(\boldsymbol{h}|\boldsymbol{y}) = P^T(\boldsymbol{h}|\boldsymbol{y}) = P(\boldsymbol{h}|\boldsymbol{y})$, the variance in the feature space is finite, and the label proportions are all non-zero. When the number of training and testing samples goes to infinity, $\hat{\boldsymbol{\gamma}}^T$ is asymptotically consistent for $\boldsymbol{\gamma}^T$ if $(\boldsymbol{M}^s)^\intercal \boldsymbol{M}^s$ is invertible for all s.*

Note that the superscript T means target, while $\intercal$ means transpose. The proof sketch of Theorem 3.1 is given in the supplemental material section B. This theorem strictly considers a single source domain; it is straightforward to be extended to multiple domains by the same arguments. When it is generalized to multiple source domains, the optimum values of the estimation $\hat{\boldsymbol{\gamma}}^T$ estimated from different domains are equal because the assumption $P(\boldsymbol{h}|y)$ is domain-invariant. Succinctly, a linear combination of asymptotically unbiased estimator is still asymptotically unbiased.

3.2 Domain Weighting Scheme

Because irrelevant domains can harm adaptation performance [21], multiple domain adaptation should primarily use information from the most similar domains. However, which domains are relevant is unknown *a priori*, so a weighting scheme was developed to determine the most relevant domains. The weight for source domain $\mathcal{D}_s$ is denoted as λ^s in (2). This weighting scheme allows us to create a single network to perform multiple domain adaptation, rather than using a separate network for each domain (e.g. MDANs [38]).

We determine the closest domains by finding the features with the best match in the domain adapter. To define this, the last hidden layer of the domain adapter is given as $\boldsymbol{z} = f_D(\boldsymbol{h}; \boldsymbol{\theta}_D)$, where $f_D()$ is a neural network with parameter $\boldsymbol{\theta}_D$. Note that this is *not* the standard feature space. Then the weights are

$$\boldsymbol{\lambda} = \text{softmax}([-||\mathbb{E}[\boldsymbol{z}^1] - \mathbb{E}[\boldsymbol{z}^T]||_2^2, -||\mathbb{E}[\boldsymbol{z}^2] - \mathbb{E}[\boldsymbol{z}^T]||_2^2, \cdots, -||\mathbb{E}[\boldsymbol{z}^S] - \mathbb{E}[\boldsymbol{z}^T]||_2^2]),$$

where the softmax is taken over this distance for each domain. $\boldsymbol{z}^s$ and $\boldsymbol{z}^T$ is the source and target features, respectively. Note that the distances can be scaled to determine the peakiness of the softmax function, but in practice the scale of 1 worked well.

We would like to note three important properties of this approach. First, the choice of z is important, because there is only a softmax function between z and the prediction on domains. Therefore, if two domains are similar, then they are on average indistinguishable and appear the same to the domain adversary. Second, it is unnecessary to correct for the label imbalance. Because the label proportions re-weight the domain loss, the feature space at this stage has already accounted for the label imbalance. As an alternative approach, this weight can be estimated by the average probability that a sample in $\mathcal{D}_s$ is confused for a target sample; empirically, both strategies gave similar performance. Third, there is a positive feedback loop in this weighting scheme, which could potentially pose an issue if it is focused on unrelated domains. However, this feedback can be beneficial to narrow the focus to relevant domains. Empirically, we have only observed increased performance from this weighting, so this feedback loop does not appear to be a practical issue.

3.3 Extending to Distribution Matching

Mean matching is an effective way of estimating label proportions; however, in many situations it is beneficial to match more than the first moment. This can be done with by matching the estimated target distribution $Q^s(\boldsymbol{h})$ and the ground truth $P^T(\boldsymbol{h})$ with an f-divergence [1]:

$$d_F(P^T(\boldsymbol{h}), Q^s(\boldsymbol{h})) = \int P^T(\boldsymbol{h}) F\left(\frac{Q^s(\boldsymbol{h})}{P^T(\boldsymbol{h})}\right) d\boldsymbol{h}. \quad (4)$$

While there are many forms of f-divergences, we choose $F(v) = (v-1)^2$ to match prior studies [7], which can be effectively estimated using kernel functions. In this form, a lower bound of (4) is $d_F(P^T(\boldsymbol{h}), Q^s(\boldsymbol{h})) = \max_{r^s} \int Q^s(\boldsymbol{h}) r^s(\boldsymbol{h}) d\boldsymbol{h} - \int P^T(\boldsymbol{h}) \left(\frac{r^s(\boldsymbol{h})^2}{2} + r^s(\boldsymbol{h})\right) d\boldsymbol{h}$ using the Legendre-Fenchel convex duality [24]. This lower bound is maximized when function $r^s(\boldsymbol{h})$ equals the density ratio $\frac{Q^s(\boldsymbol{h})}{P^T(\boldsymbol{h})}$ [13]. The lower bound of the f-divergence in (4) requires the maximum over all possible functions for $r^s(\cdot)$, which is not achievable in practice. As a surrogate, we limit $r^s(\boldsymbol{h})$ to a kernel space defined by grid points as

$$r^s(\boldsymbol{h}) = (\boldsymbol{\alpha}^s)^\intercal \boldsymbol{\phi}^s(\boldsymbol{h}). \quad (5)$$

$r^s(\boldsymbol{h})$ is defined as a weighted combination of kernel functions $\boldsymbol{\phi}^s(\boldsymbol{h})$ with parameters $\boldsymbol{\alpha}^s$ that will be learned. The kernel is evaluated as a radial basis function with respect to anchor or grid points. In previous works [7, 37, 26], all training samples are taken as grid points. However, it is impracticable to include all training samples in the kernel of a large dataset due to the complexity scaling of kernel methods. Computational efficiency can be accomplished through a variety of methods, such as pre-defining fixed grid points or randomly sampling a subset of the data points [30]. For simplicity, we used grid points at the mean of conditional functions for labels and domains, which worked well empirically.

If we substitute $r^s(\boldsymbol{h})$ in (5) into a lower bound of (4), the f-divergence between $Q^s(\boldsymbol{h})$ and $P^T(\boldsymbol{h})$ can be approximated as

$$\max_{\alpha^s} -\frac{1}{2}(\boldsymbol{\alpha}^s)^\intercal \left[\int P^T(\boldsymbol{h}) \boldsymbol{\phi}^s(\boldsymbol{h}) (\boldsymbol{\phi}^s(\boldsymbol{h}))^\intercal d\boldsymbol{h}\right] \boldsymbol{\alpha}^s + (\boldsymbol{\alpha}^s)^\intercal \left[\int P^s(\boldsymbol{h}|\boldsymbol{y}) \boldsymbol{\phi}^s(\boldsymbol{h}) d\boldsymbol{h}\right] \boldsymbol{\gamma}^T - 1, \quad (6)$$

where $P^s(\boldsymbol{h}|\boldsymbol{y})$ is the concatenation of $[P^s(\boldsymbol{h}|y=1), \cdots, P^s(\boldsymbol{h}|y=L)]$. The derivation of (6) is given in Supplemental Section A. To simplify the notation, define $\boldsymbol{A} = \int P^T(\boldsymbol{h}) \boldsymbol{\phi}^s(\boldsymbol{h}) (\boldsymbol{\phi}^s(\boldsymbol{h}))^\intercal d\boldsymbol{h}$ and $\boldsymbol{B} = \int P^s(\boldsymbol{h}|\boldsymbol{y}) \boldsymbol{\phi}^s(\boldsymbol{h}) d\boldsymbol{h}$, where the superscript domain index is omitted. The optimum $\boldsymbol{\alpha}^s$ in (6) is $\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{\gamma}^T$. Remember that the goal is to minimize the f-divergence with respect to $\boldsymbol{\gamma}^T$, i.e. match distribution $Q^s(\boldsymbol{h})$ and $P^T(\boldsymbol{h})$. Substituting the optimum value of $\boldsymbol{\alpha}^s$ into (6), the objective of $\min_{\gamma^T} d_F(Q^s(\boldsymbol{h}), P^T(\boldsymbol{h}))$ becomes

$$\min_{\gamma^T, \gamma_l^T \geq 0, ||\gamma^T||_1 = 1} -\frac{1}{2}(\boldsymbol{\gamma}^T)^\intercal \boldsymbol{B}^\intercal \boldsymbol{A}^{-1} \boldsymbol{B} \boldsymbol{A}^{-1} \boldsymbol{B} \boldsymbol{\gamma}^T + \boldsymbol{\gamma}^T \boldsymbol{B}^\intercal \boldsymbol{A}^{-1} \boldsymbol{B} \boldsymbol{\gamma}^T. \quad (7)$$

Next we sill give how to estimate the integral with finite samples. By using kernel methods, $\boldsymbol{A}$ and $\boldsymbol{B}$ can be approximated as

$$\hat{\boldsymbol{A}} = \frac{1}{n^T} \sum_{j:s_j=T} \boldsymbol{\phi}^s(\boldsymbol{h}_j) (\boldsymbol{\phi}^s(\boldsymbol{h}_j))^\intercal$$
$$\hat{\boldsymbol{B}} = [\frac{1}{n_1^s} \sum_{j:y_j=l, s_j=s} \boldsymbol{\phi}(\boldsymbol{h}_j), \cdots, \frac{1}{n_L^s} \sum_{j:y_j=L, s_j=s} \boldsymbol{\phi}(\boldsymbol{h}_j)]. \quad (8)$$

Note that $\intercal$ is matrix transpose (different from T). If we have a total of S domains, there will be a total of $S \times (L+1)$ parameter α's to be learned. The total number of grid point is $L+1$, because we choose to use the label center in each domain. Since each $\boldsymbol{\alpha}$ is independent, the optimal $\boldsymbol{\alpha}^s$ in $\mathcal{D}_s$ can be written as

$$\hat{\alpha}^s = (\hat{\boldsymbol{A}} + \delta \boldsymbol{I})^{-1} \hat{\boldsymbol{B}} \boldsymbol{\gamma}^T, \quad (9)$$

where the identity matrix is added to ensure invertability. With this optimal $\hat{\alpha}^s$, the only parameter to be optimized is $\boldsymbol{\gamma}^T$. Thus (7) can be approximated as

$$\min_{\gamma^T, \gamma_l^T \geq 0, ||\gamma^T||_1 = 1} -\frac{1}{2}(\hat{\boldsymbol{B}}\gamma^T)^\intercal (\hat{\boldsymbol{A}} + \delta \boldsymbol{I})^{-1} \hat{\boldsymbol{A}} (\hat{\boldsymbol{B}} + \delta \boldsymbol{I})^{-1} \hat{\boldsymbol{B}} \gamma^T + (\hat{\boldsymbol{B}}\gamma^T)^\intercal (\hat{\boldsymbol{A}} + \delta \boldsymbol{I})^{-1} (\hat{\boldsymbol{B}}\gamma^T) \quad (10)$$

Here we omit the superscript 's' of $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{B}}$ for simplicity. Strictly, each domain $\mathcal{D}_s$ should have its own $\hat{\boldsymbol{A}}^s$ and $\hat{\boldsymbol{B}}^s$. When combining all source domains, the total matching loss function can be written as

$$\mathcal{L}_{r_F}(\boldsymbol{\gamma}^T) = \textstyle\sum_{s=1}^S \lambda^s d_F(P^T(\boldsymbol{h}), Q^s(\boldsymbol{h})), \quad (11)$$

where $d_F(P^T(\boldsymbol{h}), Q^s(\boldsymbol{h}))$ is approximated by the function in (10).

3.4 Algorithm Outline

Combining all loss terms together, we need to jointly optimize neural network parameters $\boldsymbol{\theta}_h$, $\boldsymbol{\theta}_D$, $\boldsymbol{\theta}_Y$ and the target label proportion $\boldsymbol{\gamma}^T$. The objective function of the proposed model is given as

$$\min_{\theta_h, \theta_Y, \gamma^T} \max_{\theta_D} \mathcal{L}_Y(\boldsymbol{\theta}_h, \boldsymbol{\theta}_Y) + \alpha_\gamma \mathcal{L}_\gamma(\boldsymbol{\gamma}^T) - \alpha_D \mathcal{L}_D(\boldsymbol{\theta}_h, \boldsymbol{\theta}_D). \quad (12)$$

Here, $\mathcal{L}_Y(\boldsymbol{\theta}_h, \boldsymbol{\theta}_Y)$ is the standard cross-entropy label prediction loss. For purposes of optimization, the label estimation $\boldsymbol{\gamma}^T$ is considered a variable *only* in $\mathcal{L}_\gamma(\boldsymbol{\gamma}^T) = \alpha_{\gamma,1}\mathcal{L}_{r_M}(\boldsymbol{\gamma}^T) + \alpha_{\gamma,2}\mathcal{L}_{r_F}(\boldsymbol{\gamma}^T)$, where $\mathcal{L}_{r_M}(\boldsymbol{\gamma}^T)$ is defined in (3) and $\mathcal{L}_{r_F}(\boldsymbol{\gamma}^T)$ in (11). The constraint on $\boldsymbol{\gamma}^T$ is satisfied by linking through a softmax function. For the other loss terms, $\boldsymbol{\gamma}^T$ is considered a constant. The label proportion estimator is also not used to update the feature extractor.

By setting α_γ to zero, the model loss in Eq. (12) is equivalent to DANN if the label proportions do not update. (Note Eq. (1) is given in expectations while Eq. (12) is over observed samples.) In our experiments, we compare two distinct strategies, the first only using mean matching, and the second using mean and distribution matching. The pseudo-code of the proposed algorithm is given in Algorithm 1.

Algorithm 1 Multiple Source Domain Adaptation for Target Shift

Input: Source samples $\{\boldsymbol{x}_i, y_i, s_i\}_{i=1}^{N_S}$ and target samples $\{\boldsymbol{x}_i, s_i\}_{i=N_S+1}^{N_T}$.
Output: Classifier parameters $\boldsymbol{\theta}_h, \boldsymbol{\theta}_Y, \boldsymbol{\theta}_D$ and target label proportion $\boldsymbol{\gamma}^T$

Calculate source label proportions $\boldsymbol{\gamma}^s$ for $s = 1, \cdots, S$.
Initialize $\boldsymbol{\gamma}^T = [\frac{1}{L}, \cdots, \frac{1}{L}]$ and $\lambda^s = \frac{1}{S}$.
for $iter = 1$ to max_iter **do**
  Sample a mini-batch training set.
  % Update Label Predictor and Feature Extractor
  Fix $\boldsymbol{\gamma}^T$. Compute $\nabla\boldsymbol{\theta}_Y = \frac{\partial\mathcal{L}_Y}{\partial\boldsymbol{\theta}_Y}$ and $\nabla\boldsymbol{\theta}_h = \frac{\partial\mathcal{L}_Y}{\partial\boldsymbol{\theta}_h} - \alpha_D\frac{\partial\mathcal{L}_D}{\partial\boldsymbol{\theta}_h}$ using source samples. Update $\boldsymbol{\theta}_Y$ and $\boldsymbol{\theta}_h$ by gradient methods.
  % Update Domain Adapter
  Update estimate of $\boldsymbol{\lambda}$ by exponential smoothing.
  Calculate $\boldsymbol{\beta}^s$ from current estimate of $\boldsymbol{\gamma}^T$.
  Compute $\nabla\boldsymbol{\theta}_D = \frac{\partial\mathcal{L}_D}{\partial\boldsymbol{\theta}_D}$ using weighted source and target samples. Update $\boldsymbol{\theta}_D$ by gradient methods.
  % Update Target Label Proportion
  Compute $\nabla\boldsymbol{\gamma}^T = \frac{\partial\mathcal{L}_\gamma(\boldsymbol{\gamma}^T)}{\partial\boldsymbol{\gamma}^T}$ using (3) and (11) on the mini-batch. Update $\boldsymbol{\gamma}^T$ by gradient methods.
end for

4 Related Works

First, we discuss previous works to estimate the proportion of labels in a blind test set. The most commonly used technique is based on marginal distribution matching [37, 7, 23]. A key idea is that the marginal target domain sample distribution, $P^T(\boldsymbol{x})$, should match the distribution of a source domain weighted by the target label proportions. This can be estimated by integrating the joint of the source domain, $P^s(\boldsymbol{x}, y)$, with respect to estimated label proportions. Kernel mean matching [11] is proved to be an effective technique to solve this problem, which has been extended in numerous ways [37, 7, 23]. However, using a RKHS to estimate $P^s(\boldsymbol{x}|y)$ suffers from the curse of dimensionality, reducing the utility in high dimensional feature space. Finally, the concept of Fisher consistency has been used to analyze several algorithms theoretically [32].

The covariate shift issue has an abundance of historical literature [29, 37, 31, 12, 36]. This literature focuses on solving the discrepancy in conditional probability of $p(y|\boldsymbol{x})$, while implicitly assuming the label distribution is the same in the source and target. In order to deal with target shift, people tend to use re-weight training samples in a given feature space [23]. Kernel methods can be used to learn weighting for each individual data point [19], but is not feasible on big data. Domain adaptation aims to learn domain-invariant features, such as Transfer Component Analysis (TCA) [25] and Subspace Alignment (SA) [8]. Recently, many works have explored how to learn a domain-invariant neural network feature extractor [20, 19], including via adversarial learning [9, 38, 2, 14]. They can achieve domain-invariant features by playing a min-max game between a label classifier and a domain classifier. Compared with TCA and SA, neural network more naturally extends to a large scale dataset. [4] proposes a partial domain adaptation in two domains under the an adversarial framework. However, generalizing their work to our situation is not trivial. Based upon Generative Adversarial Networks (GANs) [10], many recent approaches have proposed to learn domain invariant features by transferring samples from source domain to the target [27, 18, 17, 22]. To the extent of our knowledge, these GAN-based frameworks have not considered target shift or multiple source domains for the domain adaptation task.

Recently, optimal transport has been used to analyze the problem of label shift in domain adaptation [26], but did not consider learning a feature extractor in conjunction with their framework. Notably, estimating terms in optimal transport is computationally expensive; accuracy of fast neural network based approximations is not guaranteed [5]. The target shift problem has also been addressed by using conditional properties via confusion matrix consistency [16]. This approach has not been extended to multiple domains or adapted to learn domain-invariant feature. To the extent of our knowledge, this is the first work that learns domain-invariant features while adjusting for target shift.

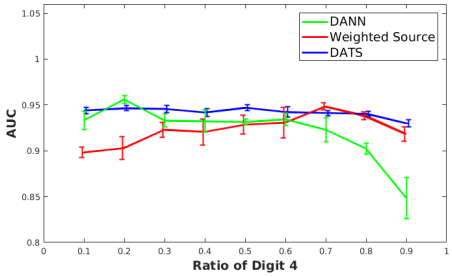
5 Experiments

In this section, we test the proposed algorithm (DATS) on image and neural datasets. Most of the comparison methods are based on neural networks. For standard optimization based methods [7, 37], the required matrix inversion hinders their application to large-scale data. In the following, all benchmarked algorithms share the same feature extractor structure as the baseline model to ensure a fair comparison. Both 'mean matching' and 'DATS' are our proposed models for target shift. 'Mean Matching' only has mean difference loss $\mathcal{L}_{r_M}$, while DATS contains both the label matching losses $\mathcal{L}_{r_M}$ and $\mathcal{L}_{r_F}$. Note that DANN [9] or MDANs [38] can be viewed as similar models without label matching losses ($\alpha_\gamma = 0$), allowing close examination of the impact of the label matching.

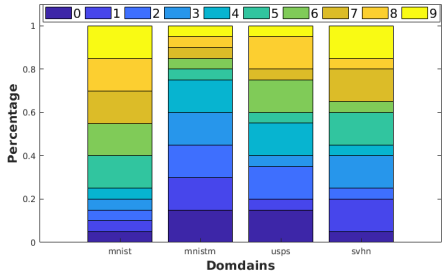
5.1 Synthetically Setting Properties on Toy Datasets

We first test our model on domain adaptation in handwritten digits where we synthetically alter the target shift between the source and target domains. The training set is MNIST, which is composed of digit '4' and '9', with label proportion of 20% and 80%, respectively. The test set is MNISTM, which also contains digit '4' and '9' from, while the proportion of digit '4' changes from 10% to 90% with 10% increments. These two digits are chosen intentionally because they are similar in shape. The feature extractor is composed of two convolutional layers. Deeper networks overfit in this problem [34]. Both the domain adapter and label predictor are two-layer MLPs with softmax output. ReLU non-linearities are used. The result is given in Figure 2(a).

When the target label proportion is similar to the source, the baseline DANN model performs well, because there is minimal target shift. As the proportion of digit '4' increases in the target set, the amount of the target shift increases. Weighting the classes in the source set to match a uniform target label distribution, as the red line in Figure 2(a), the performance trend is positive as the target domain becomes uniform. This is caused by the up-weighting of digit '4' and down-weighting of '9' without using any prior knowledge about the target label proportion. In comparison, the proposed algorithm robustly has high performance regardless of the label proportions. α_γ and α_D are all set as 1.0 in this experiment. The proposed model is not overly sensitive to these tuning parameters. Note if the parameter α_D of (1) in DANN is too large, the domain adversary becomes too powerful and predictive performance tanks due to label imbalance. Specifically, the strength of the adversary in DANN and 'source



(a) AUC on MNISTM



(b) Label proportion in each domain.

Figure 2: (Top) Model performance comparison with different label proportion on test set. (Bottom) Label proportion in each domains for MNIST, MNISTM, USPS and SVHN.

weighted' is tuned to maximize performance. As a result, the maximum AUC in DANN is above .5 because the discriminator was weakened to maximize performance (note that in practice it is not feasible to tune this parameter on an unlabeled target domain). For our proposed models, the estimated $\hat{\gamma}^T$ has at most 0.05 difference compared to the ground truth label proportions.

Next we look at four digit datasets: MNIST, MNISTM, USPS and SVHN. To evaluate the influence of label imbalance, we randomly assign different label proportions for each of the datasets (Figure 2(b)). Each time, one dataset is left out as a target while the other three are treated as training. Table 1 gives the classification accuracy. The top row gives the name of the target domain. Note that the proposed approach robustly adapts to this situation, whereas prior methods do not. For SA [8], the feature input is the encoded feature $\boldsymbol{h}$ from baseline model for a fair comparison.

The proposed model outperforms both DANN and MDANs on all tasks, illustrating the usefulness of the label matching term $\mathcal{L}_\gamma(\boldsymbol{\gamma}^T)$. Since the weighing scheme in MDANs does not jointly considers the label proportion, it is not robust under target shift. Practically, mean matching can stabilize the model, while adding the distribution matching marginally outperforms using only mean matching; however, even our basic strategy

	MNIST	MNISTM	USPS	SVHN
Baseline	94.7	57.3	89.0	41.5
SA [8]	92.5	48.8	85.6	40.3
DAN [19]	95.7	61.7	89.5	42.5
DTN [20]	96.2	61.7	89.6	41.7
Black Box [16]	81.5	42.0	92.4	42.2
ADDA [34]	84.8	54.4	79.5	30.8
DANN [9]	94.8	56.6	89.5	45.0
MDANs [38]	96.3	59.6	91.3	48.0
Mean Matching	96.6	67.1	92.3	47.7
DATS	**97.3**	**68.2**	**94.5**	**48.2**

Table 1: Accuracy on digit image classification.

with minimal tuning parameters performs well compared to competing algorithms.

5.2 Real Datasets

We test our model on a real data composed electrical brain activity recordings using Electroencephalography (EEG) and Local Field Potentials (LFP) signals. These two datasets are described below.

ASD Dataset: The Autism Spectral Disorder (ASD) dataset contains Electroencephalography (EEG) signals from 22 children undergoing treatment for ASD. More details about this dataset can be found at [6]. The target is their treatment stage, which is either before treatment, six months post treatment, or twelve months post treatment. The EEG signal is collected for each child when they are watching three one-minute videos designed to measure their responses to social and non-social stimuli with a standard 124 electrode layout. As is common in real-world data, the label proportions are variable, which is visualized in Appendix C.

The prediction goal for this dataset is to determine when a measurement is taken. This would allow one to track how neural dynamics change as a result of treatment. Towards this end, we use the SyncNet [15] approach, which is a convolutional neural network with domain-specific interpretable features as the feature extractor.

	ASD	LFP
SyncNet [15]	62.1	74.5
SA [8]	62.5	72.4
Black Box [16]	53.6	*
DAN [19]	61.8	69.3
DANN [9]	63.8	75.1
MDANs [38]	63.4	71.4
Mean Matching	65.2	**77.4**
DATS	**67.2**	77.2

Table 2: Classification mean accuracy on EEG datasets. In our experiments, [16] did not converge well on the LFP dataset.

LFP dataset: Local Field Potential (LFP) signal are collected from implanted electrodes inside the brain. The dataset used to evaluate the proposed method contains 29 mice from two genetic backgrounds (wild-type or CLOCKΔ19), where CLOCKΔ19 is a mouse model of bipolar disorder [35]. During the data recording, each mouse spends five minutes in its home cage, spends five minutes in an open field, and ten minutes in a tail-suspension test. The task is to predict the behavior condition of the mice (home cage, open field or tail suspension). The data is pre-processed to five seconds windows. Because this dataset is controlled, its class labels are balanced. However, current experiments are being recording under freely chosen behaviors, which will result in significant target shift. In order to simulate this issue, the class labels are slightly perturbed. The label proportions for each mouse are shown in Supplemental Figure 3(b).

For both of the datasets, we perform leave-one-subject-out testing, i.e. one subject is picked out as target domain and the remaining ones are treated as source domains. Therefore, the source domain reaches 21 in ASD dataset and 28 in LFP dataset. Mean classification accuracy over the target is given in Table 2. The proposed algorithm performs well when there is clear target shift in the data. In these experiments, the number of domains can increase drastically, while each domain usually contains only a 'small' amount of data. Without adjusting for relevance of the domains, the model tends to over-fit. The proposed model, DATS, can effectively handle adjust for label imbalance and domain weighting to give higher accuracy compared to the other baseline models. The comparative methods can fail or even not converge well when source domain number is large. Again, note that even the basic proposed strategy is effective to improve domain adaptation.

6 Conclusion

In this work, we have addressed the target shift problem under an adversarial domain adaptation framework, and our strategy addresses is easily incorporated into standard frameworks. We have shown that label weighting via mean matching is a simple and effective strategy, and that using distribution matching can often improve performance. Our approach also weights source domains by their relevance, increasing efficacy on multi-domain adaptation. Experiments show that the model performs consistently well in the face of large source and target domain label shift.

Acknowledgements: Funding was provided by the Stylli Translational Neuroscience Award, Marcus Foundation, and NICHD P50-HD093074.

References

[1] S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1966.

[2] S. Ao, X. Li, and C. X. Ling. Fast generalized distillation for semi-supervised domain adaptation. In *AAAI*, 2017.

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

[4] Z. Cao, M. Long, J. Wang, and M. I. Jordan. Partial transfer learning with selective adversarial networks. *CVPR*, 2018.

[5] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 2017.

[6] G. Dawson, J. M. Sun, K. S. Davlantis, M. Murias, L. Franz, J. Troy, R. Simmons, M. Sabatos-DeVito, R. Durham, and J. Kurtzberg. Autologous cord blood infusions are safe and feasible in young children with autism spectrum disorder: Results of a single-center phase i open-label trial. *Stem Cells Translational Medicine*, 2017.

[7] M. C. Du Plessis and M. Sugiyama. Semi-supervised learning of class balance under class-prior change by distribution matching. In *ICML*, 2014.

[8] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013.

[9] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[11] A. Gretton, A. J. Smola, J. Huang, M. Schmittfull, K. M. Borgwardt, and B. Schölkopf. Covariate shift by kernel mean matching, 2009.

[12] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola. Correcting sample selection bias by unlabeled data. In *NIPS*, 2007.

[13] A. Keziou. Dual representation of φ-divergences and applications. *Comptes rendus mathématique*, 2003.

[14] Y. Li, M. Murias, S. Major, G. Dawson, and D. E. Carlson. Extracting relationships by multi-domain matching. In *Advances in Neural Information Processing Systems*, 2018.

[15] Y. Li, M. Murias, S. Major, G. Dawson, K. Dzirasa, L. Carin, and D. E. Carlson. Targeting eeg/lfp synchrony with neural nets. In *NIPS*, 2017.

[16] Z. C. Lipton, Y.-X. Wang, and A. Smola. Detecting and correcting for label shift with black box predictors. *arXiv preprint arXiv:1802.03916*, 2018.

[17] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[18] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *NIPS*, 2016.

[19] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2016.

[20] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017.

[21] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *NIPS*, 2009.

[22] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto. Few-shot adversarial domain adaptation. In *NIPS*, 2017.

[23] T. D. Nguyen, M. Christoffel, and M. Sugiyama. Continuous target shift adaptation in supervised learning. In *Asian Conference on Machine Learning*, 2016.

[24] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 2010.

[25] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 2011.

[26] I. Redko, N. Courty, R. Flamary, and D. Tuia. Optimal transport for multi-source domain adaptation under target shift. *arXiv preprint arXiv:1803.04899*, 2018.

[27] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo. From source to target and back: symmetric bi-directional adaptive gan. *arXiv preprint arXiv:1705.08824*, 2017.

[28] C. Scott, G. Blanchard, and G. Handy. Classification with asymmetric label noise: Consistency and maximal denoising. In *Conference On Learning Theory*, 2013.

[29] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 2000.

[30] E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *NIPS*, 2006.

[31] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe. Direct importance estimation for covariate shift adaptation. *Annals of the Institute of Statistical Mathematics*, 2008.

[32] D. Tasche. Fisher consistency for prior probability shift. *The Journal of Machine Learning Research*, 2017.

[33] D. Tuia, R. Flamary, A. Rakotomamonjy, and N. Courty. Multitemporal classification without new labels: a solution with optimal transport. In *Analysis of Multitemporal Remote Sensing Images (Multi-Temp), 2015 8th International Workshop on the*. IEEE, 2015.

[34] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. *CVPR*, 2017.

[35] J. van Enkhuizen, A. Minassian, and J. W. Young. Further evidence for clockδ19 mice as a model for bipolar disorder mania using cross-species tests of exploration and sensorimotor gating. *Behavioural brain research*, 2013.

[36] J. Wen, C.-N. Yu, and R. Greiner. Robust learning under uncertain test distributions: Relating covariate shift to model misspecification. In *ICML*, 2014.

[37] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang. Domain adaptation under target and conditional shift. In *ICML*, 2013.

[38] H. Zhao, S. Zhang, G. Wu, J. P. Costeira, J. M. Moura, and G. J. Gordon. Multiple source domain adaptation with adversarial training of neural networks. *NeurIPS*, 2018.