LTF: A Label Transformation Framework for Correcting Target Shift

Jiaxian Guo [1] **Mingming Gong** [2] **Tongliang Liu** [1] **Kun Zhang** [3] **Dacheng Tao** [1]

Abstract

Distribution shift is a major obstacle to the deployment of current deep learning models on real-world problems. Let Y be the target (label) and X the predictors (features). We focus on one type of distribution shift, *target shift*, where the marginal distribution of the target variable P_Y changes but the conditional distribution $P_{X|Y}$ does not. Existing methods estimate the density ratio between the source- and target-domain label distributions by density matching. However, these methods are either computationally infeasible for large-scale data or restricted to shift correction for discrete labels. In this paper, we propose an end-to-end Label Transformation Framework (LTF) for correcting target shift, which implicitly models the shift of P_Y and the conditional distribution $P_{X|Y}$ using neural networks. Thanks to the flexibility of deep networks, our framework can handle continuous, discrete, and even multi-dimensional labels in a unified way and is scalable to large data. Moreover, for high dimensional X, such as images, we find that the redundant information in X severely degrades the estimation accuracy. To remedy this issue, we propose to match the distribution implied by our generative model and the target-domain distribution in a low-dimensional feature space that discards information irrelevant to Y. Both theoretical and empirical studies demonstrate the superiority of our method over previous approaches.

[1]UBTECH Sydney AI Centre, School of Computer Science, Faculty of Engineering, The University of Sydney, Darlington, NSW 2008, Australia [2]School of Mathematics and Statistics, The University of Melbourne [3]Department of Philosophy, Carnegie Mellon University. Correspondence to: Jiaxian Guo <jguo5934@uni.sydney.edu.au>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

1. Introduction

Standard supervised learning methods typically assume that the training set (source domain) and the test set (target domain) have the same distribution. However, the data available for training is always limited and may not represent and reflect the statistics of the test data. As such, the source-domain distribution P^S_{XY} is often different from the target-domain distribution P^T_{XY}, degrading the performance of the models learned on the training set. This phenomenon is called *distribution shift*, which has become a major obstacle to the deployment of deep learning models in the real world.

To overcome distribution shift and improve the prediction on test data, existing methods have studied various distribution shift settings, among which *covariate shift* and *target shift* have been widely considered. Covariate shift assumes that the marginal P_X changes across training and test sets, whereas the conditional distribution $P_{Y|X}$ is invariant (Shimodaira, 2000; Sugiyama et al., 2008; Gretton et al., 2009; Long et al., 2015; 2017; 2018; Liu et al., 2019). Target shift assumes that the label distribution P_Y changes but the conditional distribution $P_{X|Y}$ stays the same (Zhang et al., 2013; Iyer et al., 2014; Lipton et al., 2018; Azizzadenesheli et al., 2019).

Here we focus on the target shift problem, since it appears in a wide range of real-world learning problems. For example, in disease prediction, where our goal is to predict disease Y from symptoms X, the distribution of the disease can change over location and time, while the mechanism of symptoms $P_{X|Y}$ is rather stable. Consider the flu prediction task, the data available for flu prediction is always has a regular morbidity rate, but if a model is trained on these data, the performance of this model will decrease when it is used to detect flu in a location or over a period with a high morbidity rate (Tasche, 2017). In addition, target shift also exists in many computer vision applications, such as predicting object locations (Yang et al., 2018) ands direction and human poses (Martinez et al., 2017). The distribution of object locations or human poses often changes across training and test sets.

Despite being a natural phenomenon in many real applications, target shift is relatively understudied compared to covariate shift. Chan & Ng (2005) proposed an expectation–maximization algorithm that requires estimation of the

conditional distribution $P_{X|Y}$. Unfortunately, estimating $P_{X|Y}$ is difficult for high-dimensional X and moreover, it does not apply to regression problems. Zhang et al. (2013) proposed a nonparametric method to estimate the density ratio P_Y^T/P_Y^S by kernel mean matching of distributions, which is applicable to both regression and classification problems. However, this approach is not compatible with large data as the computational cost is quadratic in the sample size. Recently, Lipton et al. (2018); Azizzadenesheli et al. (2019) proposed efficient and sample size-independent methods that make use of the confusion matrix of a classifier learned on the training set. These methods have shown promising performance on large-scale data but are only applicable to classification problems.

In this paper, we aim to propose a new framework that can correct target shift for both discrete and continuous Y. Compared to existing methods, we make the following contributions. First, instead of estimating the density ratio P_Y^T/P_Y^S, we model the change in the distribution P_Y by a neural label transformation T, which transforms the training label distribution P_Y^S to a new label distribution P_Y^R that can approximate the unknown P_Y^T in the test set. Thanks to the flexibility of neural nets, we can design different transform models T to deal with different types of Y, including discrete, continuous, and even multi-dimensional labels. Second, because of the absence of labels in the test set, we model the invariant conditional distribution $P_{X|Y}$ using a conditional generator G on the training set. By concatenating the label transformation model T with the conditional generator G, we can generate corresponding sample distribution P_X^R, which is then matched with P_X^T to estimate the parameters in T. Third, for high dimensional X, such as images, we observe that the redundant information significantly degrades the estimation accuracy. To remedy this issue, we theoretically analyze this phenomenon and propose to match the distributions of a feature representation of X that discards the information irrelevant to Y.

To demonstrate the advantage of our framework in practical applications, we apply our method to a range of label types, including classification (discrete label), regression (continuous label) and objects 2D object position prediction (multi-dimension label), in various target shift settings, such as random target shift, high probability label quantification and low probability label quantification). The empirical results demonstrate the generality, flexibility and superiority of our framework compared to previous methods.

2. Related Work

Covariate shift and *target shift* are two common types of *distribution shift*. The former one assumes that the feature distribution P_X changes over training set and test set, but the conditional distribution $P_{Y|X}$ from label to data remains unchanged, while the latter one assumes that the label distribution P_Y changes but $P_{X|Y}$ is invariant.

The existing methods solve *covariate shift* and *target shift* using re-weighting methods, which are also used in a wide range of problems, *e.g.*, label-noise (Liu & Tao, 2015; Yu et al., 2017b; Cheng et al., 2017; Fang et al., 2020) . We firtstly introduce methods dealing with *covariate shift* problems shortly, where many methods estimate importance sample weights P_X^T/P_X^S (Zadrozny, 2004; Huang et al., 2007; Sugiyama et al., 2008; Gretton et al., 2009) via kernel methods (Huang et al., 2007; Gretton et al., 2009; Zhang et al., 2013) or using a discriminative classifier (Lopez-Paz & Oquab, 2016; Liu et al., 2017). Then they correct models by retraining a new model with re-weighted training samples using estimated P_X^T/P_X^S under the ERM framework (Shimodaira, 2000). More recent works learn domain-invariant representations $X' = h(X)$ that have similar marginal distributions across domains ($P_{X'}^T \approx P_{X'}^S$) (Si et al., 2009; Pan et al., 2010; Baktashmotlagh et al., 2013; Tzeng et al., 2014; Ganin et al., 2016; Long et al., 2015).

Similar to the correction of *Covariate shift*, there are two major steps to solve *target shift* problems. The first step is to estimate the label distribution P_Y^T in the target domain or the ratio P_Y^T/P_Y^S. The second step is to construct an unbiased estimate of the target domain risk based on the results from the first step. Zhang et al. (2013); Iyer et al. (2014); Nguyen et al. (2016); Gong et al. (2016) proposed to estimate P_Y^T or P_Y^T/P_Y^S by matching a weighted combination of conditionals $P_{X|Y}^S$ in the source domain the marginal distribution P_X^T in the target domain. The matching of distributions is achieved by minimizing suitable divergence measures (Gretton et al., 2012; Sugiyama et al., 2012) w.r.t. the weights on $P_{X|Y}^S$. In the discrete Y scenario, Lipton et al. (2018) proposed a method which estimates the importance weight (P_Y^T / P_Y^S) by matching the output of trained classifier on the training set (confusion matrix), and then Azizzadenesheli et al. (2019) turned this problem as a linear programming problem and iteratively minimized the error of label distributions between the training set and the test set, improving the accuracy of estimated target label distribution P_Y^T. In addition, Azizzadenesheli et al. (2019) added a regularization term to make the algorithm compatible with the situation where the target sample size is small.

3. Methodology

Given training data $\mathcal{D}_s = \{x_i^s, y_i^s\}_{i=1}^{n_s} \subseteq \mathcal{X} \times \mathcal{Y}$ independently drawn from an unknown joint distribution P_{XY}^S, denoted as the source domain distribution, and test data $\mathcal{D}_t = \{x_i^t, y_i^t\}_{i=1}^{n_t}$ drawn from the target-domain distribution P_{XY}^T, where y_i^t is unknown, target shift assumes that $P_{X|Y}^S = P_{X|Y}^T = P_{X|Y}$ and $P_Y^S \neq P_Y^T$. Our goal is to build

a model to estimate the label distribution P_Y^T in the target domain such that we can correct the label shift between the training and test data and thus improve the prediction performance on the test set. We consider both continuous Y, i.e., $\mathcal{Y} = \mathbb{R}^d$, and discrete Y, i.e., $\mathcal{Y} = \{1, \ldots, K\}$.

3.1. Review of Previous Methods

To estimate the label distribution P_Y^T, existing methods use the relation between source and target distributions:

$$P_X^T(x) = \int_y P_{X|Y}(x|y) P_Y^T(y) dy \tag{1}$$

$$= \underbrace{\int_y P_{XY}^S(x,y) \frac{P_Y^T(y)}{P_Y^S(y)} dy}_{P_X^{\text{new}}}. \tag{2}$$

Because P_{XY}^S and P_X^T can be estimated from $\mathcal{D}_s$ and $\mathcal{D}_t$, previous methods (Zhang et al., 2013; Gong et al., 2016) estimate the density ratio $\beta^*(y) = \frac{P_Y^T(y)}{P_Y^S(y)}$ by minimizing the empirical Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) between P_X^T and P_X^{new}:

$$\Big|\Big| \frac{1}{n_t} \sum_{i=1}^{n_t} \psi(x_i^t) - \frac{1}{n_s} \sum_{i=1}^{n_s} \beta(y_i^s) \psi(x_i^s) \Big|\Big|_{\mathcal{H}}^2, \tag{3}$$

$$s.t.\ \beta(y_i^s) \geq 0, \text{ and } \sum_{i=1}^{n_s} \beta(y_i^s) = n_s, \tag{4}$$

where ψ is the feature mapping from $\mathcal{X}$ to a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ associated with a kernel function $k(x, x') = \langle \psi(x), \psi(x') \rangle_{\mathcal{H}}$. For kernel functions that have no explicit ψ, for example, RBF kernels, we need to use kernel trick to calculate (3). The computational cost is quadratic in the sample size and thus the algorithm is not scalable to large datasets.

When Y is discrete, recent works (Lipton et al., 2018; Azizzadenesheli et al., 2019) proposed to estimate $\boldsymbol{\beta} = [\beta(y = 1), \ldots, \beta(y = K)]^T$ by using the confusion matrix of a classifier f:

$$\hat{\mathbf{q}} = \hat{\mathbf{C}} \hat{\boldsymbol{\beta}}, \tag{5}$$

where $\hat{\mathbf{C}}$ is the confusion matrix with each element $\hat{C}_{ij} = \frac{1}{n_s} \sum_{k=1}^{n_s} \mathbb{1}\{f(x_k^s) = i, y_k^s = j\}$ and $\hat{\mathbf{q}}_i = \frac{1}{n_t} \sum_{j=1}^{n_t} \mathbb{1}\{f(x_j^t) = i\}$. It can be seen that (5) corresponds to a specific form of (3) in which the feature mapping ψ is set to $\psi(x) = one\_hot(f(x))$, where one_hot is a function mapping $\hat{y} = f(x)$ to its corresponding one-hot vector. Because the dimensionality of $\psi(x)$ is simply the number of classes K, which is usually much smaller than the sample size, $\hat{\boldsymbol{\beta}}$ can be obtained efficiently. However, this type of methods only work for discrete labels.

3.2. Our Framework

Instead of estimating the density ratio $\beta(y)$, our framework estimates the target-domain marginal distribution using a constructed distribution P_X^R defined as follows:

$$P_X^R = \int_{y^r} P_{X|Y}(x|y^r) P_Y^R(y^r) dy^r$$

$$= \int_{y^r} P_{X|Y}(x|y^r) \int_{y^s} P_{Y^R|Y^S}(y^r|y^s) P_Y^S(y^s) dy^s dy^r, \tag{6}$$

where we build a new label distribution P_Y^R by transforming the training label distribution P_Y^S using the transition model $\int_{y^s} P_{Y^R|Y^S}(y^r|y^s) P_Y^S(y^s) dy^s$. Because Y is not observed in the test domain, we need to estimate the label transition model by comparing P_X^R and P_X^T. In the following sections, we will show how the transformation between P_Y^S and P_Y^T can be estimated from the labeled training set and unlabeled test set.

Figure 1 displays the flowchart of our framework. First, we transform the samples drawn from P_Y^S using the **Label Transformation** network LT which maps P_Y^S to a distribution P_Y^R. Because there are only unlabeled data in the target domain, we cannot directly match P_Y^R with the target-domain label distribution P_Y^T. Therefore, we then pass the transformed labels into the **Label Influence Recovery** network G, which models the conditional distribution $P_{X|Y}$ implicitly, to generate samples with distribution P_X^R. Finally, we match the generated distribution P_X^R with the target domain P_X^T to estimate the parameters in the label transformation network, such that P_Y^R can approximate the target-domain label distribution P_Y^T. After estimating P_Y^R, we can train an unbiased classifier for prediction in the target domain. In the following, we present the details of each component in our framework.

3.2.1. Label Transformation Network

Here we use a neural network LT to transform the training label distribution P_Y^S to a new label distribution P_Y^R, such that we can directly generate the corresponding sample distribution P_X^R together with one generator G that models $P_{X|Y}$. Specifically, we use the following functional model:

$$Y^R = LT(Y^S, Z), \tag{7}$$

where LT is modeled by a neural net and Z is a random variable with distribution P_Z. (7) models the conditional distribution $P_{Y^R|Y^S}$ implicitly. Because $P_Y^R = \int_{y^s} P_{Y^R|Y^S}(y^r|y^s) P_Y^S(y^s) dy^s$, we can sample $y_i^r \sim P_Y^R$ by first sampling y_i^s from the source-domain labels, and then generate the corresponding $y_i^r = LT(y_i^s, z_i)$, where $z_i \sim P_Z$. Note that in some situations, such as discrete Y, it might be more convenient to directly use the parametric form of $P_{Y^R|Y^S}$.

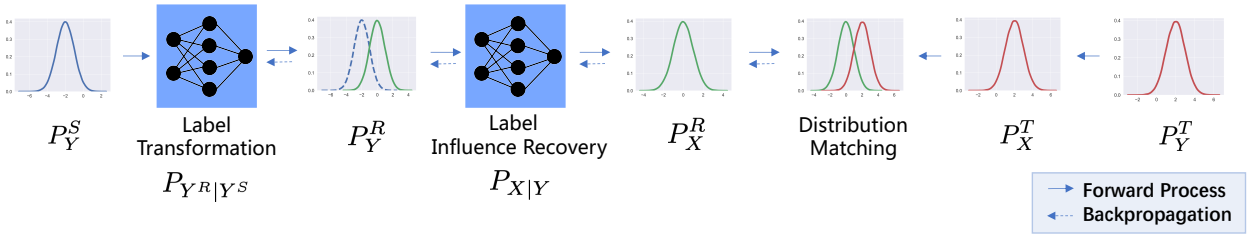


Figure 1. The illustration of the LTF framework. Here we make an example which assumes that $X = Y + \epsilon$. Firstly, the **Label Transformation Model** LT transforms the training label distribution P_Y^S (blue one) to a new label distribution P_Y^R (green one), and then the **Label Influence Recovery Model** G generates the sample distribution P_X^R from the data generated from P_Y^R. By matching the target sample distribution P_X^T (red one) and P_X^R and fixing the G, the P_Y^R from LT is expected to be close to P_Y^T. As such, the target label distribution P_Y^T can be approximated by P_Y^R.

If labeled data were available in the target domain, we can then simply match the empirical P_Y^R and P_Y^T to learn LT. Unfortunately, target-domain labels are not available in unsupervised domain adaptation, but still, in the target domain we have unlabeled data $\{x_i^t\}_{i=1}^{n_t}$, which can be used to estimate LT. To this end, we need to transform P_Y^R to a distribution P_X^R in the $\mathcal{X}$ space. Because P_X^R captures the influence of P_Y^R, we can possibly estimate P_Y^R (or LT) by matching P_X^R and P_X^T, from which we can sample data points to estimate and minimize their distance.

3.2.2. Label Influence Network

In order to transform P_Y^R to P_X^R, we make use of the following model:

$$X^R = G(Y^R, E), \tag{8}$$

where G is a neural generator, and E is a random variable with distribution P_E, which is set to normal distribution. We can use (8) to implicitly model $P_{X|Y}$. Due to $P_X^R = \int_{y^r} P_{X^R|Y^R}(x^r|y^r)P_Y^R(y^r)dy^r$, we can sample $x_i^r \sim P_X^R$ by first sampling y_i^r using (7), and then generate the corresponding $x_i^r = G(y_i^r, e_i)$, where $e_i \sim P_E$.

Since G corresponds to the generator in a conditional generative adversarial network (Mirza & Osindero, 2014; Miyato & Koyama, 2018; Gong et al., 2019), we can learn it from the source domain data $\mathcal{D}_s = \{x_i^s, y_i^s\}_{i=1}^{n_s}$ by adversarial training. Let $Q_{X|Y}$ denote the conditional distribution specified by G. If the input of G is drawn from P_Y^S, the joint distribution of the generated data will be $Q_{XY} = Q_{X|Y}P_Y^S$. We can estimate G by minimizing the Jensen-Shannon Divergence (JSD) between Q_{XY} and P_{XY}^S (Mirza & Osindero, 2014):

$$\min_G \max_{D_G} \mathop{\mathbb{E}}_{(X,Y)\sim P_{XY}^S}[\log(D_G(X,Y))] + \mathop{\mathbb{E}}_{E\sim P_E, Y\sim P_Y^S}[\log(1 - D_G(G(Y,E),Y))], \tag{9}$$

where D_G is an introduced discriminator (Goodfellow et al., 2014) to play the mini-max game together with G. (9) is the negative cross entropy loss, and in some real experiments, we need to replace (9) by negative hinge-loss because it is more stable in image generation, as demonstrated in (Miyato et al., 2018; Brock et al., 2018).

3.2.3. Distribution Matching

As described above, we can then construct a new data distribution P_X^R with the label transformation network LT and the label influence network G. To estimate LT, we fix G and minimize the JSD between P_X^R and P_X^T w.r.t. LT by the following objective

$$\min_{LT} \max_{D_{LT}} \mathop{\mathbb{E}}_{X\sim P_X^T}[\log(D_{LT}(X))] + \mathop{\mathbb{E}}_{Z\sim P_Z, E\sim P_E, Y^S\sim P_Y^S}[\log(1 - D_{LT}(G(LT(Y^S, Z), E)))]. \tag{10}$$

where D_{LT} is an introduced discriminator (Goodfellow et al., 2014) to perform adversarial training with LT. In detail, when the label is continuous, the whole network is totally differentiable, so we can simply estimate LT by using backpropagation. However, in the case of discrete labels, we cannot backpropagate through the label y_i^r. Fortunately, we can assume a parametric form of P_Y^R, i.e., the categorical distribution, in case of discrete Y. Thus, we can make use of the Gumbel-softmax trick (**??**) or the REINFORCE trick (Williams, 1992) to backpropagate through the discrete labels y_i^r. The two tricks have been successfully employed in various problems such as text generation (Yu et al., 2017a) and neural architecture search (**?**).

Gumbel-softmax Trick Let Y^{Ro} and Y^{So} denote the one-hot representations of Y^R and Y^S, respectively. We can use a special LT function to sample from $P_{Y^R|Y^S}$:

$$\tilde{Y}_k^{Ro} = \frac{\exp((\log M_k Y^{So} + Z_k)/\tau)}{\sum_{i=1}^K \exp((\log M_i Y^{So} + Z_i)/\tau)}, \tag{11}$$

where $\tilde{Y}_k^{Ro}$ is the kth element of $\tilde{Y}^{Ro}$, $Z_k \sim Gumbel(0,1)$, τ is the temperature, and M_k is the kth row of the transition matrix $\mathbf{M}$, whose ijth element is $P(Y^R = i|Y^S = j)$. As $\tau \to 0$, $\tilde{Y}^{Ro}$ provides a good approximation of the one-hot Y^{Ro}. Since the softmax function is differentiable, it

enables end-to-end learning of LT, which only contains $\mathbf{M}$ as parameters.

REINFORCE Trick Because $P_{Y^R|Y^S}$ involves learnable parameters $\mathbf{M}$, we rewrite it as $P_{\mathbf{M}}(Y^R|Y^S)$ and reformulate (10) as

$$\min_{\mathbf{M}} \max_{D_{LT}} \mathop{\mathbb{E}}_{X\sim P_X^T}[\log(D_{LT}(X))]+ \tag{12}$$
$$\mathop{\mathbb{E}}_{E\sim P_E, Y^R\sim P_{\mathbf{M}}(Y^R|Y^S), Y^S\sim P_Y^S}[\log(1-D_{LT}(G(Y^R,E)))].$$

The gradient w.r.t. LT can be written as:

$$\mathop{\mathbb{E}}_{E\sim P_E, Y^R\sim P_{\mathbf{M}}(Y^R|Y^S), Y^S\sim P_Y^S}[\log(1-D_{LT}(G(Y^R,E)))$$
$$\nabla_{\mathbf{M}} \log\ P_{\mathbf{M}}(Y^R|Y^S)].$$

Feature Matching Generally speaking, we estimate the prior distribution in the target domain P_Y^T by comparing the marginal distributions of X in the target domain and the transformed source domain. However, for some high dimensional data such as images, X might contain many redundant features X_R that are unrelated to Y, causing unnecessary estimation errors of P_Y^T. Intuitively, this is because the conditional distributions of these redundant features X_R satisfy $P_{X_R|Y} = P_{X_R}$, which are not helpful in identification of P_Y^T but will cause additional estimation error. To improve the estimation accuracy, we propose to estimate LT by matching $P_{h(X)}^R$ and $P_{h(X)}^T$ instead, where h is a pre-trained network that extracts compact representations from raw X data. Therefore, we replace (10) by

$$\min_{T} \max_{D_{LT}} \mathop{\mathbb{E}}_{X\sim P_X^T}[\log(D_{LT}(h(X)))] + \mathop{\mathbb{E}}_{Z\sim P_Z, E\sim P_E, Y^S\sim P_Y^S}$$
$$[\log\ (1-D_{LT}(h(G(LT(Y^S,Z),E))))]. \tag{13}$$

Ideally, we aim to find $h(X)$ such that $Y \perp\!\!\!\perp X|h(X)$ by using the source-domain labeled data. This conditional independence property implies that $h(X)$ contains all information in X that is relevant to Y. Learning conditional invariant representation has been shown to be effective in correcting covariate shift (Stojanov et al., 2019). However, since Stojanov et al. (2019) set h as a linear transformation and measure conditional dependency using kernel measures (Fukumizu et al., 2004), the method cannot learn compact representations for images and is computationally expensive. Here we use a convolutional network as h to extract feature representations and measure the dependency by assuming a (generalized) linear model for $P_{Y|h(X)}$. This is sensible because the features extracted by nonlinear neural networks are usually linearly separable. Proposition 1 shows how h can be learned to satisfy the conditional independence property. (The proof can be found at the supplementary material A.1)

Proposition 1 *Assuming $P_{Y|h(X)}$ can be modeled by a (generalized) linear model, i.e., linear regression model for continuous Y and multinomial logistic regression model for discrete Y. Let sample size $n \to \infty$, h learned by minimizing the mean squared error (for continuous Y) or the cross-entropy loss (for discrete Y) satisfies $Y \perp\!\!\!\perp X|h(X)$.*

3.2.4. Shift Correction

After quantifying the target label distribution P_Y^T, the model with target shift problems should be corrected and adapted to the target domain. The previous work choose to re-train the model under the importance-weighted ERM framework (Gretton et al., 2009; Shimodaira, 2000; Sugiyama et al., 2008). In our framework, we can retrain the source-domain model with new data drawn from our model. As it is time-consuming to retrain a new model, a quick adaptation method is also provided in our framework. As described by Proposition 1, if h learned at the uniform Training set satisfies the conditional independence property with Y, the output layer of a neural network is the only module needed to be adapted to the new label distribution P_Y^R given the feature extractor h. In our framework, we fine-tune the output layer several epochs using the samples generated by our Label Influence Recovery network G with the label distribution P_Y^R learned by Label Transformation Network LT. As such, the output layer will be quickly adapted to the target domain.

4. Experiments

To verify the effectiveness and universality of the proposed framework, we design experiments for three target shift scenarios, i.e., the discrete, continuous, and multi-dimensional target shift, on various datasets.

4.1. Discrete Target Shift Experiments

We compare our method with the competitors on three datasets, e.g., MNIST, FASHION-MNIST, and CIFAR10 (Krizhevsky & Hinton, 2009). We follow the same setting of BBSE (Lipton et al., 2018) and RLLS (Azizzadenesheli et al., 2019). Specifically, for MNIST, we use a simple two-layer neural network; the Resnet-18 (He et al., 2016) and CNN in DCGAN (Radford et al., 2015) are chosen for CIFAR 10 and FASHION-MNIST, respectively. The learning rate is set to 0.01. Moreover, we use the network architecture of BigGAN (Brock et al., 2018) and the loss of TAC-GAN (Gong et al., 2019) to model the invariant conditional distribution $P_{X|Y}$. The original training sets given in the datasets are used as the training set for the proposed method and the baselines. The test set is sampled to have a specific label distribution P_Y^T and is of size 10,000. For the quantification of P_Y^T, we use the REINFORCE trick instead of Gumbel-softmax trick, as the temperature τ in

the Gumbel-softmax trick is hard to choose.

4.1.1. Shift Settings

In our paper, the label distribution P_Y^S in the training set is a **uniform** distribution over all classes. For the test set, we consider three types of shifts: Tweak-One shift, Minority-Class shift, and Random Dirichlet shift. These settings are designed to capture diverse label probability changes, i.e., large label probability change, small label probability change, and random label distribution change. We repeat the experiments 10 times to verify the effectiveness and robustness of the proposed method.

Tweak-One Shift To evaluate the performance on the large label probability quantification. In our experiments, the ratio of one class is set to $[0.5, 0.6, 0.7, 0.8, 0.9]$, respectively, while ratios of other classes are uniform.

Minority-Class Shift To evaluate the performance on the small label probability quantification. In our experiments, $[20\%, 30\%, 40\%, 50\%]$ classes are set to 0.001, respectively, while ratios of other classes are uniform.

Random Dirichlet Shift In this shift, we randomly generate a label distribution P_Y^T by employing the Dirichlet distribution with different values of the concentration parameter α. Then, we re-sample the test set according to the generated distribution P_Y^T. In our experiments, α are set to $10, 1, 0.1, 0.01$. Note that the generated label distribution P_Y^T tends to be smoother for bigger α.

4.1.2. Evaluation metrics and Results

As done in BBSE (Lipton et al., 2018) and RLLS (Azizzadenesheli et al., 2019), the accuracy and F1 score (Goutte & Gaussier, 2005) are used as evaluation metrics, allowing us to compare the performance of different methods more comprehensively (Azizzadenesheli et al., 2019). We also evaluate the estimation error of the estimated label weights (P_Y^T / P_Y^S) by using mean square error (MSE).

We compare our method with the two recent methods: BBSE (Lipton et al., 2018) and RLLS (Azizzadenesheli et al., 2019), which estimate label weights $\hat{\boldsymbol{\beta}}$ using the confusion matrix of a classifier f trained on the training set. To verify Proposition 1, we consider a variant of RLLS called RLLS(feature), which matches distributions on the feature space $h(X)$. RLLS(feature) can also be considered as setting $\psi(X)$ to $h(X)$ in (3). For the evaluation of the shift correction, we evaluate the performance of classifiers trained on the training set without adaptation (denoted as Baseline) and the classifiers trained on weighted training sets, where the weights are estimated by using target domain labels (denoted as BEST(ERM)). Similarly, we also test the classifiers trained on weighted training data, where the weights are obtained by RLLS, BBSE and RLLS(feature). For our method, we have two ways to utilize the label distributions estimated by our framework. The first one is to re-train a new classifier using the the weighted training set (Ours(ERM)). The second one is the fine-tuning method described in 3.2.4, which is denoted as Ours(Fine-tune). Specifically, we fine-tune the output layer of the pretrained classifier on the source domain by 10 epochs, using the data generated from our model.

Due to the page limit, we only show the results of CIFAR10 dataset in the paper and the results of MNIST and FASHION-MNIST can be found in the supplemental materials A.2. In terms of the estimation error of the target label distribution P_Y^T, the subfigure (a) of Figure 2, 3, 4 demonstrate that the label weights estimated by our framework are more accurate and stable than previous methods. In addition, the RLLS (feature) algorithm that matches label distribution on feature space of classifier trained on the training set also achieves better performance than BBSE and RLLS in most settings. For the accuracy and F1 score of the corrected classifiers, subfigures (b) and (c) of Figure 2, 3, 4 show that the classifier corrected by our framework can achieve better performance in both two evaluation metrics in most settings. Also, our fast fine-tune method achieves comparable performance compared with re-weighting methods.

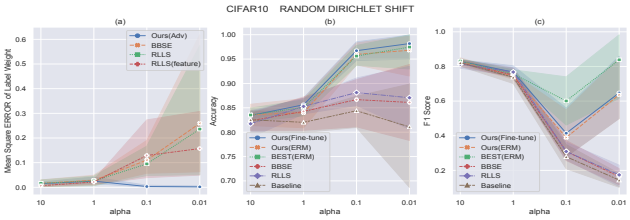


Figure 2. (a) Mean squared errors of estimated label weights (lower is better), (b) accuracy, and (c) F-1 score (higher is better) on CIFAR10 for uniform training set and random Dirichlet shifted test set, where the smaller *alpha* corresponds to larger shift.

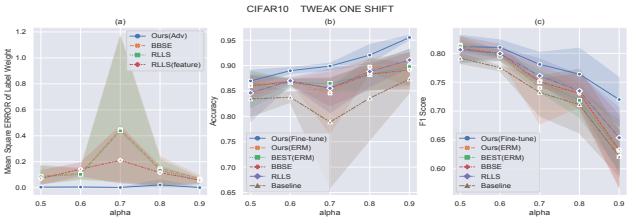


Figure 3. (a) Mean squared errors of estimated label weights (lower is better), (b) accuracy, and (c) F-1 score (higher is better) on CIFAR10 for uniform training set and Tweak-One shifted test set, where *alpha* is the probability of tweaked class.

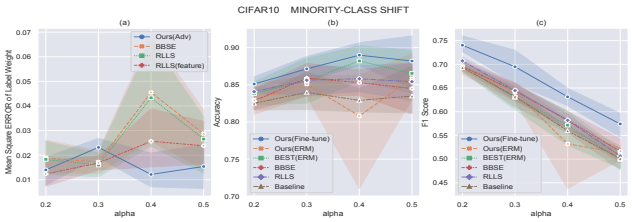


Figure 4. (a) Mean squared errors of estimated label weights (lower is better), (b) accuracy, and (c) F-1 score (higher is better) on CIFAR10 for uniform training set and minority-class shifted test set, where *alpha* is the ratio of minority classes.

4.2. Continuous Target Shift Experiments

In this section, we design two experiments to verify the effectiveness of our model on continuous target shift problems. Firstly, we conduct experiments on a synthetic data that evaluates the performance of our framework on simple continuous target shift problems. Then we apply our model on a real data application: Object 1D position prediction (Matthey et al., 2017), which evaluates the performance of our model on the continuous target shift problem in the high-dimensional X situation.

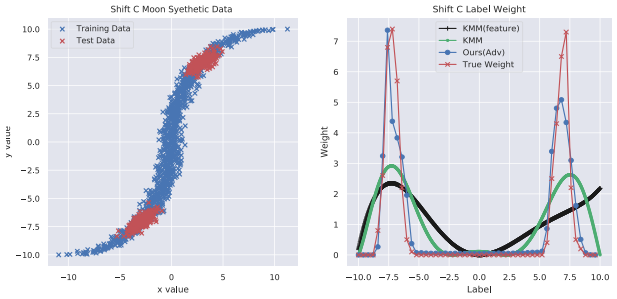
4.2.1. Synthetic Data Experiment

In this experiment, we design a toy dataset by modifying a classic and popular synthetic data experiment (MOON dataset (Ganin et al., 2017)) in *covariate shift*. We first generate two quarter circles with radius R 10 and sample size 1000 as the training set, which is shown in Figure 5(a). The range of a single continuous label is from -10 to 10 and the values are uniformly distributed. Then we generate the test set with 500 samples in the same way according to several target label distributions. Here we consider 4 types of target shift to evaluate model performance and robustness.

Experimental Setting In this experiment, the architectures of all modules in our framework are three-hidden layers neural networks with 10 hidden neurons. In the distribution matching module, the baseline KMM (Zhang et al., 2013) uses MMD with the median kernel width to match the built data distribution P_X^{new} and target data distribution P_X^T. To fairly compare the methods, we also use MMD to do distribution matching.

Shift Settings To evaluate the model's label quantification performance, we set 4 target shift situations.
Shift A: Set the target label distribution P_Y^T as a Gaussian distribution with the mean of $\frac{\sqrt{2}}{2} * R$ and variance of 1.
Shift B: Set the target label distribution P_Y^T as a Gaussian distribution with the mean of $-\frac{\sqrt{2}}{2} * R$ and variance of 1.
Shift C: The target label distribution is a mixture Gaussian distribution with Shift A and Shift B, with a mixture proportion 0.5.
Shift D: The target label distribution is a random label distribution generated by a randomly parameterized neural network.

Baselines The classic KMM methods (Zhang et al., 2013) are chosen as our baselines. We consider two variants: KMM that matches the distributions in the raw input space and KMM(feature) that matches the distributions in feature space of the regressor.



(a) (b)

Figure 5. (a) The illustration of Moon Synthetic Data (Shift C), where the generated two quarter circles training set as blue symbols show. (b) The visualization of label weight P_Y^T/P_Y^S of KMM, KMM(feature), our framework and the Ground Truth.

	SHIFT A	SHIFT B	SHIFT C	SHIFT D
Baseline	0.0061 ± 0.0012	0.0059 ± 0.0008	0.0055 ± 0.0009	0.034 ± 0.0114
KMM	0.0048 ± 0.0011	0.0044 ± 0.0005	0.0044 ± 0.0004	0.0275 ± 0.0096
KMM (feature)	0.0045 ± 0.0007	0.0039 ± 0.0006	0.0043 ± 0.0004	0.0276 ± 0.0097
Ours	**0.0036 ± 0.0002**	**0.0024 ± 9e-5**	**0.0036 ± 0.0004**	**0.0251 ± 0.0121**

Table 1. The results of Continuous target shift Synthetic Data Experiments. The value is the mean square error of prediction value and groun truth. The baseline is the original regressor trained on the standard training set, and the KMM is (Zhang et al., 2013)

Results In this section, to compare the performance of estimated target label distribution qualitatively, we visualize the estimated density ratio (P_Y^T/P_Y^S) of Shift C in Figure 5. More figures about other shift settings can be found in supplementary materials A.3.Visually, our model has better label distribution estimation performance compared with other methods.

Then we evaluate the mean square error of the baseline regressor without adaptation and three others corrected

by KMM, KMM(feature), and Ours(Adv) respectively. The results are shown in Table 1. It can be seen that the MSE errors of our framework are significantly lower than those of the other methods in all shift settings, and KMM (feature) achieves slightly better performance than the original KMM method in some settings, which verifies the correctness of Proposition 1.

4.2.2. OBJECT 1D LOCATION PREDICTION EXPERIMENT

In this experiment, we use a popular disentanglement dataset called Sprites [1] (Matthey et al., 2017). This dataset consists of 737,280 2D shapes images, which are generated from 6 ground-truth independent latent factors. Some example images are shown in Figure 6. The factors include color, shape, scale, rotation, x, and y positions of a sprite. We choose the x or y position of sprites as the target variable and consider it as a regression problem.

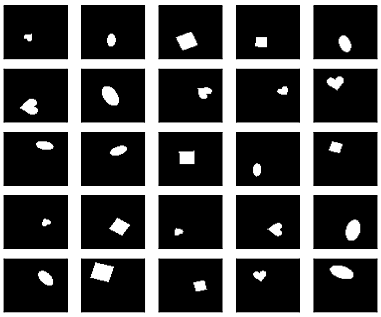


Figure 6. Illustration of the sprites dataset. This sprites in this dataset have 3 shapes(square, ellipse, heart), 6 scales values linearly spaced in [0.5, 1], 40 orientation values in [0, 2 pi], 32 X position values in [0, 10], 32 Y position values in [0, 10]

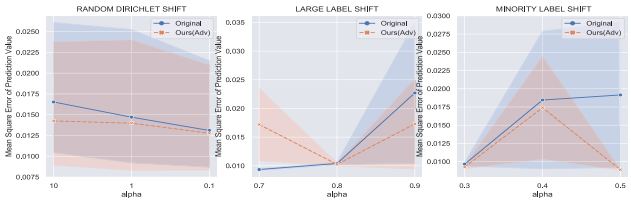


Figure 7. The prediction mean square error of 1D sprite position prediction (lower is better). (a) Random Dirichlet shift, where the smaller *alpha* corresponding to the bigger shift. (b) Large target shift, where *alpha* is the possibility of shifted label. (c) Minority shift, where *alpha* is the ratio of minority classes.

Experimental Setting We use the network architecture in DCGAN (Radford et al., 2015) as feature extractor for the regressor, the learning rate for the regressor is set to 1e-4, which is the best learning rate according to our experiments. The DCGAN (Radford et al., 2015) is used to model the invariant distribution $P_{X|Y}$ and the Transformation Model LT is a simple 3-layer neural network.

For training set, we randomly sample 40000 images with **uniform** x value distribution from overall dataset and the test set consists of 40000 samples (sampled from specified distribution for target shift).

Shift Settings As the x position (or y position) in this dataset has 32 possible values (but we see it as a regression problem), we can use the same shift method with 4.1.1 to evaluate the methods' target shift quantification ability. Specifically, we repeat the experiments 3 times for each setting to evaluate the model performance.

Baselines and Results As the KMM method cannot be applied into large-scale dataset, so the only baseline in this experiment is the baseline regressor without adaptation. We evaluate the mean square error of output value (x or y position) of original regression model and the regressor corrected by our framework. The results are shown as Figure 7, and the model corrected by our framework outperforms the basline model in most settings.

4.3. Multi-Dimensional Target Shift Experiments

In this experiment, we design a simple multi-dimensional target shift experiment, which is object 2D location prediction. We use the same dataset with the object 1D location prediction experiment, but we predict both x and y position values of a sprite. As such, the label Y in this experiment is 2-dimension, increasing the difficulty of detecting and correcting the target shift.

Experimental Setting We use the same network architecture and train/test split as described in 4.2.2 . For Transformation Model LT, two networks are used to model the x and y position target shift respectively as the x and y position value in this dataset are independent. As such, using two networks will reduce the difficulty of quantifying target label distribution P_Y^T.

Shift Settings The settings are also same with 4.2.2 described, but we shift the x and y position value respectively.

Baselines and Results Similar to the 1D position prediction, the baseline is the baseline regressor without adaptation as our methods is the first method which is compatible with large-scale multi-dimensional target shift problems. The results are shown in Figure 8. It can be seen that the regressor corrected by our framework can

[1] https://github.com/deepmind/dsprites-dataset

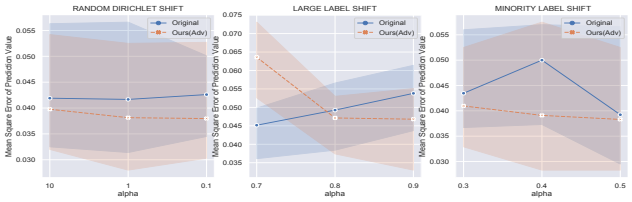


Figure 8. The prediction mean square error of 2D sprite position prediction (Lower is better). (a) Random Dirichlet shift, where the smaller *alpha* corresponds to larger shift. (b) Large target shift, where *alpha*is the possibility of a shifted label (c) Minority shift, where *alpha* is the ratio of minority classes.

achieve lower MSE error than the baseline method in most settings.

5. Conclusion

In this paper, we propose an end-to-end target shift quantification and correction framework called Label Transformation Framework which can deal with discrete, continuous and multi-dimensional target shift problems. Based on this framework, we further find that matching the distributions of a feature representation of X that discards the information irrelevant to Y can have better performance over other methods which quantify the label distribution P_Y^T based on scratch data or biased output. In the experiments, we apply our framework to several classification and regression tasks under various target shift settings. The results show that our framework has better performance and universality than previous methods. Future work will be extending our framework to address conditional shift, where $P_{X|Y}$ also changes across domains.

6. Acknowledgements

This work was supported by Australian Research Council Projects FL-170100117, DP-180103424, IH-180100002, IC-190100031, DE-190101473, LE-200100049 and the United States Air Force under Contract No. FA8650-17-C-7715.

References

Kamyar Azizzadenesheli, Anqi Liu, Fanny Yang, and Animashree Anandkumar. Regularized learning for domain adaptation under label shifts. *arXiv preprint arXiv:1903.09734*, 2019.

Mahsa Baktashmotlagh, Mehrtash T Harandi, Brian C Lovell, and Mathieu Salzmann. Unsupervised domain adaptation by domain invariant projection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 769–776, 2013.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Yee Seng Chan and Hwee Tou Ng. Word sense disambiguation with distribution estimation. In *IJCAI*, volume 5, pp. 1010–5, 2005.

Jiacheng Cheng, Tongliang Liu, Kotagiri Ramamohanarao, and Dacheng Tao. Learning with bounded instance-and label-dependent label noise. *arXiv preprint arXiv:1709.03768*, 2017.

Tongtong Fang, Nan Lu, Gang Niu, and Masashi Sugiyama. Rethinking importance weighting for deep learning under distribution shift. *arXiv preprint arXiv:2006.04662*, 2020.

Kenji Fukumizu, Francis R Bach, and Michael I Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5(Jan):73–99, 2004.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. In *Domain Adaptation in Computer Vision Applications*, pp. 189–209. Springer, 2017.

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*, pp. 2839–2848, 2016.

Mingming Gong, Yanwu Xu, Chunyuan Li, Kun Zhang, and Kayhan Batmanghelich. Twin auxiliary classifiers gan. *arXiv preprint arXiv:1907.02690*, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European Conference on Information Retrieval*, pp. 345–359. Springer, 2005.

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.

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.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pp. 601–608, 2007.

Arun Iyer, Saketha Nath, and Sunita Sarawagi. Maximum mean discrepancy for class ratio estimation: Convergence bounds and kernel selection. In *International Conference on Machine Learning*, pp. 530–538, 2014.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Zachary C. Lipton, Yu-Xiang Wang, and Alex Smola. Detecting and correcting for label shift with black box predictors, 2018.

Hong Liu, Mingsheng Long, Jianmin Wang, and Michael Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In *International Conference on Machine Learning*, pp. 4013–4022, 2019.

Song Liu, Akiko Takeda, Taiji Suzuki, and Kenji Fukumizu. Trimmed density ratio estimation. In *Advances in Neural Information Processing Systems*, pp. 4518–4528, 2017.

Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2015.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pp. 97–105, 2015.

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*, pp. 2208–2217. JMLR. org, 2017.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pp. 1640–1650, 2018.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2640–2649, 2017.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Takeru Miyato and Masanori Koyama. cgans with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Tuan Duong Nguyen, Marthinus Christoffel, and Masashi Sugiyama. Continuous target shift adaptation in supervised learning. In *Asian Conference on Machine Learning*, pp. 285–300, 2016.

Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

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

Si Si, Dacheng Tao, and Bo Geng. Bregman divergence-based regularization for transfer subspace learning. *IEEE Transactions on Knowledge and Data Engineering*, 22 (7):929–942, 2009.

Petar Stojanov, Mingming Gong, Jaime Carbonell, and Kun Zhang. Low-dimensional density ratio estimation for

covariate shift correction. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3449–3458, 2019.

Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul V Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, pp. 1433–1440, 2008.

Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation. *Annals of the Institute of Statistical Mathematics*, 64(5):1009–1044, 2012.

Dirk Tasche. Fisher consistency for prior probability shift. *The Journal of Machine Learning Research*, 18(1):3338–3369, 2017.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Xue Yang, Hao Sun, Xian Sun, Menglong Yan, Zhi Guo, and Kun Fu. Position detection and direction prediction for arbitrary-oriented ships via multitask rotation region convolutional neural network. *IEEE Access*, 6:50839–50849, 2018.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017a.

Xiyu Yu, Tongliang Liu, Mingming Gong, Kun Zhang, Kayhan Batmanghelich, and Dacheng Tao. Transfer learning with label noise. *arXiv preprint arXiv:1707.09724*, 2017b.

Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 114. ACM, 2004.

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