Re-weighted Adversarial Adaptation Network for Unsupervised Domain Adaptation

Qingchao Chen *[1] Yang Liu *[2] Zhaowen Wang [3] Ian Wassell [2] Kevin Chetty[1]
[1] University College London [2] University of Cambridge [3] Adobe Research
{qingchao.chen.13, k.chetty}@ucl.ac.uk {yl504, ijw24}@cam.ac.uk zhawang@adobe.com

Abstract

Unsupervised Domain Adaptation (UDA) aims to transfer domain knowledge from existing well-defined tasks to new ones where labels are unavailable. In the real-world applications, as the domain (task) discrepancies are usually uncontrollable, it is significantly motivated to match the feature distributions even if the domain discrepancies are disparate. Additionally, as no label is available in the target domain, how to successfully adapt the classifier from the source to the target domain still remains an open question. In this paper, we propose the Re-weighted Adversarial Adaptation Network (RAAN) to reduce the feature distribution divergence and adapt the classifier when domain discrepancies are disparate. Specifically, to alleviate the need of common supports in matching the feature distribution, we choose to minimize optimal transport (OT) based Earth-Mover (EM) distance and reformulate it to a minimax objective function. Utilizing this, RAAN can be trained in an end-to-end and adversarial manner. To further adapt the classifier, we propose to match the label distribution and embed it into the adversarial training. Finally, after extensive evaluation of our method using UDA datasets of varying difficulty, RAAN achieved the state-of-the-art results and outperformed other methods by a large margin when the domain shifts are disparate.

1. Introduction

Recent developments in Deep Neural Networks (DNN) have yielded state-of-the-art results from supervised learning applications in computer vision [11][35][19]. However, the success of DNN requires a large amount of well annotated training data which is not always feasible to perform manually. Therefore, this has acted as a driver to transfer knowledge from datasets for which labels are well-defined. The Domain Adaptation (DA) problem [33] was proposed in this context where the data distribution between the target domain (a few of labels are available) and the source domain (well-annotated labels) varies so that the discriminative features and the classifiers in the source domain cannot be transferred to the target domain[33][41]. Under this regime, unsupervised domain adaptation (UDA) is the most challenging problem where no label information in the target domain is available. To successfully conduct adaptation between domains in UDA, two essential problems are required to be addressed, including matching the feature distribution and adapting the classifier from source to target domains.

Since DNN based methods exhibit strong capacity to extract transferable feature representations among datasets, research has been conducted investigating measurements to estimate distribution divergence of deep features among domains and the relevant methods to minimize them. As an un-biased estimate of distribution divergence, Maximum Mean Discrepancy (MMD) [16] has been employed in various DNN based methods for UDA [23][26][27][38][42][39]. More recently, inspired by the best-performing adversarial training in generative models, the state-of-the-art UDA methods utilize the Jenson-Shannon (JS) divergence or the more generalized f-divergence [32] implemented using DNN to estimate the distribution divergence[12] [13] [37] [22] [4].

However, both the MMD and f-divergence based methods require that feature distributions of the source and target domain share a common support. We argue that this is an unrealistic condition which can rarely be met in the real-world adaptation tasks, since the domain discrepancies are caused by a variety of factors that are difficult to control [8], such as light conditions, acquisition devices or even from different image formats e.g., RGB and HHA. From this point of view, these methods fail to adapt between domains once their distributions do not have significant overlap. More recently, to alleviate the need of a common support in UDA, optimal transport (OT) based methods have been proposed to match the source and target feature distributions by minimizing the global transportation efforts [9] [8]. However, OT based methods have not been formalized

*indicates equal contributions.

and embedded into an end-to-end pipeline to train DNNs, which limits its application to large-scale UDA problems.

In UDA, besides selecting a good divergence measure of the marginal feature distribution, it is essential to adapt the classifier between domains. Long et.al [25][26] and Courty.et.al both [8] proposed to match the joint distribution of feature and label by regarding the transductive features from the final layer's activation map of the DNN as an approximation of the target domain labels. In fact, how to match the feature distribution and meanwhile adapt the classifier is still an open question in UDA.

In this paper, we propose a Re-weighted Adversarial Adaptation Network (RAAN) for UDA to reduce disparate domain discrepancies and adapt the classifier. More specifically, there are two main contributions:

1. To match feature distributions when domains discrepancies are disparate, we train a domain discriminator network together with the conventional deep convolutional neural network (DCNN) in an adversarial manner to minimize the OT based EM distance. Compared with other methods adopting geometry-oblivious measures, RAAN can better reduce large feature distribution divergence.

2. To help adapt the classifier in UDA, we propose to match the label distribution by estimating a re-weighted source domain label distribution so that it can be similar to the unknown target label distribution. In addition, we embed it into the procedure of minimizing the EM distance during the end-to-end adversarial training procedure. This not only adapts the classifier but also helps match the marginal feature distribution.

Finally, our proposed RAAN is evaluated by conducting a series of experiments using datasets with different domain distribution divergence.

2. Related Work

In this section, we review the state-of-the-art methods in reducing the domain distribution divergence for the UDA problem.

2.1. Matching Feature Distribution using Adversarial Training

JS-divergence based methods are the best-performing techniques for measuring the divergence of feature distributions in deep adaptation networks [22] [37] [4]. Although it is not a new statistical measure, the JS divergence or the f-divergence loss is implemented by a mini-batch approach in the DNN trained in an adversarial manner [32] [15]. DANN [13] may be the first to add a domain classifier, with the aim of extracting not only discriminative features for the main classification task, but also indistinguishable ones for the domain classifiers. The adversarial loss of

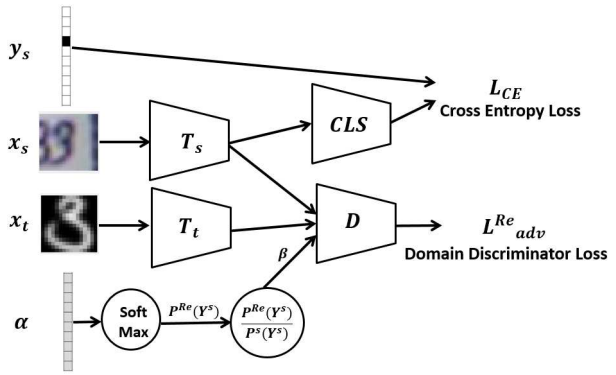

Figure 1. RAAN's architecture: first, in the source domain, the DCNN $\mathcal{T}_s$ and the classifier CLS are trained to extract discriminative features from images $\boldsymbol{x}_s$ labeled by y_s by minimizing the cross entropy loss $\mathcal{L}_{CE}$. Second, to adapt the classifier by matching the label distribution between domains, the re-weighted source domain label distribution $P^{Re}(\boldsymbol{Y}^s)$ is computed by transforming a variable $\boldsymbol{\alpha}$ using the soft-max function. Then it is straightforward to obtain the ratio vector as follows: $\boldsymbol{\beta} = \frac{P^{Re}(\boldsymbol{Y}^s)}{P^s(\boldsymbol{Y}^s)}$. To extract transferable features for target domain images $\boldsymbol{x}^t$, the target domain DCNN $\mathcal{T}_t$, domain discriminator $\mathcal{D}$ and the estimated density ratio vector $\boldsymbol{\beta}$ play the following adversarial game: $\boldsymbol{\beta}$ and $\mathcal{D}$ tries to discriminate whether features is from the target or source domain, while $\mathcal{T}_t$ tries to confuse $\mathcal{D}$ and $\boldsymbol{\beta}$.

DANN is implemented by directly maximizing the domain classification loss and reversing the gradient in the back-propagation. DRCN [14] utilized the same approach but added another loss function to minimize the reconstruction error of the data samples between domains. More recently, ADDA [37] designed two separate discriminative networks (one for each domain) to extract useful features for the main classification task. The domain discriminator network is added so that the target network and the domain discriminator network can compete with each other until the target and source domain features cannot be distinguished.

Inspired by the good performance of adversarial training in generative models, some methods generate new images that are transferable in both domains. Co-GAN [22] may be the first to design two Generative Adversarial Nets (GANs) to generate diverse images for both source and target domain. Although Co-GAN achieved good performance in adapting domains having a small discrepancy, it cannot work well when the domain shifts are disparate [37]. In contrast to Co-GAN, the pixel-level domain adaptation network (pixelDA) proposed in [4] uses one generative network to generate images indistinguishable from source and target domains. In addition, constraints on pixel level similarity between the generated and source images are utilized. In fact, the ability of generative model based methods for UDA having large discrepancy is still under investigation.

2.2. Matching Feature Distribution using OT

The most closely related approach to ours for reducing the distribution divergence is through solving the OT directly, as described in [9][8]. However, this implementation has not been included into the end-to-end learning framework and only the stand-alone De-Caffe features [11] from the DANN network are used. Instead, RAAN utilizes the domain discriminator network with the objective of minimizing the dual formulation of the Earth-Mover(EM) distance. From this perspective, the Wasserstein GAN [2] [17] is a special case to minimize the dual of EM distance, however, their ultimate goal is to generate the images. To the best of author's knowledge, RAAN may be the first to learn domain invariant features for UDA utilizing the OT based EM distance in a DNN. Note that the concurrent work [34] also adopted the EM distance as the divergence measurement in UDA, however, we are handling the more generalized scenario with unbalanced datasets.

2.3. Instance Re-weighting Scheme

The instance re-weighting scheme is well documented in the literature [7] [43], for example in the instance re-weighting of the bias in the discriminative models [42], or in the causal inference regime [44]. In DNN based methods for UDA, Yan.et.al [42] recently proposed to learn the bias of the source domain instances by the classification expectation maximization (CEM) algorithms using the MMD as the divergence measure[6].

In contrast, RAAN differs from [42] as the instance re-weighting is achieved by estimating the density ratio vector of label distributions between domains. Specifically, the estimation of the density ratio vector is embedded into the adversarial training via back-propagation. Finally, we also argue and explain why matching the label distribution helps to adapt the classifier in UDA.

3. Model

First, we introduce the notation and formulate our problem. Suppose we are given a n_{cls}-class source domain set $\boldsymbol{D}_s = \{(\boldsymbol{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ including n_s images $\boldsymbol{x}_i^s$ labeled by y_i^s and an unlabelled n_{cls}-class target domain set $\boldsymbol{D}_t = \{(\boldsymbol{x}_j^t)\}_{j=1}^{n_t}$ composed of n_t images $\boldsymbol{x}_j^t$. The random variables representing the image and label in general are denoted as $\boldsymbol{X}$ and $\boldsymbol{Y}$. As illustrated in Figure 1, RAAN is composed of three networks, specifically two conventional L-layer DCNNs $\mathcal{T}_s$ and $\mathcal{T}_t$ and a domain discriminator network $\mathcal{D}$.

The first objective of RAAN is to adapt the classifier, which is difficult without the target domain labels. However, as the label is a low-dimensional and discrete variable, it is fairly straightforward to match between domains and we argue that this can assist with the adaptation of classifiers (see reasons in section 3.2). With this intuition, a re-weighted source domain label distribution $P^{Re}(\boldsymbol{Y}^s)$ is obtained by mapping a variable $\boldsymbol{\alpha} \in \mathbb{R}^{n_{cls}}$ by the softmax function. Then estimation of $\boldsymbol{\alpha}$ aims to ensure that $P^{Re}(\boldsymbol{Y}^s)$ is similar to the unknown target one $P^t(\boldsymbol{Y}^t)$. Consequently, the density ratio vector can be denoted as $\boldsymbol{\beta} \in \mathbb{R}^{n_{cls}}$, with its $(y^s)^{th}$ element $\boldsymbol{\beta}(y^s)$ calculated by $\boldsymbol{\beta}(y^s) = \frac{P^{Re}(\boldsymbol{Y}^s=y^s)}{P^s(\boldsymbol{Y}^s=y^s)}$. As $\boldsymbol{\beta}$ can be directly computed based on $\boldsymbol{\alpha}$, in the following paper, we regard $\boldsymbol{\beta}$ as the variable under estimate.

The second objective of RAAN is to learn the domain invariant transformations $\mathcal{T}_s$ and $\mathcal{T}_t$ so that the disparate divergence between marginal feature distributions $P^t(\mathcal{T}_t^{l=L}(\boldsymbol{X}^t))$ and $P^s(\mathcal{T}_s^{l=L}(\boldsymbol{X}^s))$ is reduced. For brevity, we denote $\mathcal{T}_t$ and $\mathcal{T}_s$ to replace $\mathcal{T}_t^{l=L}$ and $\mathcal{T}_s^{l=L}$ respectively in the following. Given the images and labels in the source domain $\{\boldsymbol{x}^s, y^s\} \in \boldsymbol{D}_s$, with the aim of extracting discriminative features $\mathcal{T}_s(\boldsymbol{x}^s)$ for the classification, it is straight-forward to train the classifier CLS and $\mathcal{T}_s$ by minimizing the cross-entropy loss $\mathcal{L}_{CE}$ as follows:

$$\min_{\mathcal{T}_s, CLS} \mathcal{L}_{CE}. \tag{1}$$

To obtain transferable features $\mathcal{T}_t(\boldsymbol{x}^t)$ without labels, $\mathcal{T}_t$ is trained by playing an adversarial game with the domain discriminator network $\mathcal{D}$ and the ratio vector $\boldsymbol{\beta}$ so that the divergence between the re-weighted feature distribution in the source domain $\boldsymbol{\beta}(y^s)P^s(\mathcal{T}_s(\boldsymbol{x}^s))$ and the target domain $P^t(\mathcal{T}_t(\boldsymbol{x}^t))$ is reduced. Additionally, to better reduce the divergence between disparate domains, the OT-based EM distance is reformulated in the adversarial manner, with more details shown in section 3.1. Specifically, RAAN is trained in the following adversarial manner, where $\mathcal{D}$ with the help of $\boldsymbol{\beta}$ aims to discriminate whether features are from source or target domain, while $\mathcal{T}_t$ tries to confuse them. Based on the discriminator loss $\mathcal{L}_{adv}^{Re}$, the following objective function can be obtained:

$$\min_{\mathcal{T}_t} \max_{\mathcal{D}, \boldsymbol{\beta}} \mathcal{L}_{adv}^{Re}. \tag{2}$$

In fact, besides helping the adaptation of the classifier, matching the label distributions also eases the difficulty of matching the marginal feature distribution. The possible reason may be: if we assume that the feature generation processes are the same between domains, that is $P^s(\mathcal{T}^s(\boldsymbol{X}^s)|\boldsymbol{Y}^s) = P^t(\mathcal{T}^t(\boldsymbol{X}^t)|\boldsymbol{Y}^t)$, then $P^{Re}(\boldsymbol{Y}^s) = P^t(\boldsymbol{Y}^t)$ helps match the marginal feature distributions $P^s(\mathcal{T}^s(\boldsymbol{X}^s)) = P^t(\mathcal{T}^t(\boldsymbol{X}^t))$.

In section 3.1, to match the marginal feature distributions between domains, an OT based EM distance is introduced and implemented in an adversarial manner in RAAN. Then in section 3.2, we propose to match label distributions between domains and embed it in the adversarial training.

We also explain why this helps to adapt the classifier and meanwhile to match marginal feature distributions. Finally in section 3.3, we formulate the final objective function of RAAN.

3.1. Optimal Transport in Adversarial Training

Suppose that the empirical distributions of source and target domain features are denoted as $\boldsymbol{\mu}^s$ and $\boldsymbol{\mu}^t$ respectively as follows:

$$\boldsymbol{\mu}^s = \sum_i^{n_s} p_i^s \delta_{\mathcal{T}_s(\boldsymbol{x}_i^s)}, \boldsymbol{\mu}^t = \sum_j^{n_t} p_j^t \delta_{\mathcal{T}_t(\boldsymbol{x}_j^t)} \tag{3}$$

where $\delta_{\mathcal{T}_s(\boldsymbol{x}_i^s)}$ and $\delta_{\mathcal{T}_t(\boldsymbol{x}_j^t)}$ are the Dirac functions at location $\mathcal{T}_s(\boldsymbol{x}_i^s)$ and $\mathcal{T}_t(\boldsymbol{x}_j^t)$ and p_i^s and p_j^t are their probability masses. Then, the joint probabilistic coupling, or the transportation plan between feature distributions in source and target domains can be defined as $\boldsymbol{\gamma}$ with the marginals $\boldsymbol{\mu}^s$ and $\boldsymbol{\mu}^t$. In the discrete version, the set of probabilistic couplings $\boldsymbol{B}$ can be defined as the following:

$$\boldsymbol{B} = \left\{\boldsymbol{\gamma} \in (\mathbb{R}^+)^{n_s \times n_t} | \boldsymbol{\gamma} \mathbf{1}_{n_t} = \boldsymbol{\mu}^s, \boldsymbol{\gamma}^T \mathbf{1}_{n_s} = \boldsymbol{\mu}^t\right\} [9]. \tag{4}$$

In general, to reduce feature distribution divergence, OT based methods first estimate the optimal transportation plan between two distributions and then learn the feature transformation to minimize the cost of such a plan. Therefore, we first define the metric $J(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t)$ in equation (5) to measure the total cost of transporting probability masses from target to source domains,

$$J(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t) = \langle \boldsymbol{\gamma}, \boldsymbol{C} \rangle_F, \ with \ \boldsymbol{\gamma} \in \boldsymbol{B}, \tag{5}$$

where $\boldsymbol{C}$ is the distance matrix whose $(i,j)^{th}$ element is defined by the distance cost function $c(\mathcal{T}_s(\boldsymbol{x}_i^s), \mathcal{T}_t(\boldsymbol{x}_j^t))$ between features. The $(i,j)^{th}$ element $\boldsymbol{\gamma}(i,j)$ indicates how much mass is moved from $\mathcal{T}_t(\boldsymbol{x}_j^t)$ to $\mathcal{T}_s(\boldsymbol{x}_i^s)$, and F is the Frobenius dot product. Subsequently for brevity, we drop the index i, j to represent $\boldsymbol{x}_i^s, \boldsymbol{x}_j^t$ as $\boldsymbol{x}^s, \boldsymbol{x}^t$. After that, the OT $\boldsymbol{\gamma}_0$ can be estimated by minimizing the cost $J(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t)$ in equation (6), with the optimal transportation cost or the well-known EM distance defined by $W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t)$ in equation (7) [20]. Finally, assuming the ideal source domain features $\mathcal{T}_s(\boldsymbol{x}^s)$ are available, to learn the transferable features in target domains, it is intuitive to train the DCNN transformation $\mathcal{T}_t$ under the objective of minimizing the EM distance $W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t)$, as shown in (8).

$$\boldsymbol{\gamma}_0 = \underset{\boldsymbol{\gamma} \in \boldsymbol{B}}{\operatorname{argmin}} J(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t)[9] \tag{6}$$

$$W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t) = \min_{\boldsymbol{\gamma} \in \boldsymbol{B}} J(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t) \tag{7}$$

$$\min_{T_t} W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t) \tag{8}$$

To avoid using linear programs or iterative algorithms to compute the constraint of $\boldsymbol{\gamma}$ in equation (4), the dual formulation of $W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t)$ is utilized in equation (9) and (10) (following equation(5.3) in [40]), considering the capability of batch-wise back-propagation in DNN. More specifically, we use the domain discriminator network $\mathcal{D}$ and its variant $\hat{\mathcal{D}}$ as two dual functions in the following:

$$W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t) = \max_{\mathcal{D}\,\hat{\mathcal{D}}} \mathcal{L}_{adv}, \ where$$
$$\mathcal{L}_{adv} = \mathop{\mathbb{E}}_{\boldsymbol{x}^s \sim P^s(\boldsymbol{X}^s)} \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s)) + \mathop{\mathbb{E}}_{\boldsymbol{x}^t \sim P^t(\boldsymbol{X}^t)} \hat{\mathcal{D}}(\mathcal{T}_t(\boldsymbol{x}^t)) \tag{9}$$

$$s.t. \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s)) + \hat{\mathcal{D}}(\mathcal{T}_t(\boldsymbol{x}^t)) \leq c(\mathcal{T}_s(\boldsymbol{x}^s), \mathcal{T}_t(\boldsymbol{x}^t)). \tag{10}$$

In this paper, we choose the following distance cost function $c(\mathcal{T}_s(\boldsymbol{x}^s), \mathcal{T}_t(\boldsymbol{x}^t)) = \|\mathcal{T}_s(\boldsymbol{x}^s) - \mathcal{T}_t(\boldsymbol{x}^t)\|$ for reasons of computational efficiency and permitting gradient measurements, however, this does not infer that it is the only function that could be selected. According to the constraint (10), the best function that $\hat{\mathcal{D}}$ has to be is $-\mathcal{D}$, as $c(\mathcal{T}_s(\boldsymbol{x}^s), \mathcal{T}_t(\boldsymbol{x}^t))$ is defined to be non-negative. In this way, the constraint in (10) is equivalent to ensuring that $\mathcal{D}$ is a 1-Lipschitz function, or alternatively its gradient norm is smaller than 1. Therefore, if we use (9) and (10) in (8) to replace the EM distance, the DCNN transformation $\mathcal{T}_t$ and the domain discriminator network $\mathcal{D}$ can be trained based on the mini-max objective function in (11),

$$\min_{\mathcal{T}_t} W(\boldsymbol{\mu}^s, \boldsymbol{\mu}^t) = \min_{\mathcal{T}_t} \max_{\mathcal{D}} \mathcal{L}_{adv}, where$$
$$\mathcal{L}_{adv} = \sum_{(\boldsymbol{x}^s, y^s) \sim P^s(\boldsymbol{X}, \boldsymbol{Y}^s)} \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s)) P^s(\mathcal{T}_s(\boldsymbol{x}^s)|y^s) P^s(y^s)$$
$$- \mathop{\mathbb{E}}_{\boldsymbol{x}^t \sim P^t(\boldsymbol{X}^t)} \mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t))$$
$$s.t. \| \bigtriangledown_{\mathcal{T}_t(\boldsymbol{x}^t)} \mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t))\|_2 \leq 1,$$
$$\| \bigtriangledown_{\mathcal{T}_s(\boldsymbol{x}^s)} \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s))\|_2 \leq 1. \tag{11}$$

3.2. Adapting the Classifier by Label Distribution Matching

Although OT based EM distance is utilized to match feature distributions $P^s(\mathcal{T}_s(\boldsymbol{X}^s))$ and $P^t(\mathcal{T}_t(\boldsymbol{X}^t))$, we argue that it is not enough to successfully adapt the classifier from source to target domain, since $P^s(\mathcal{T}_s(\boldsymbol{X}^s)) = P^t(\mathcal{T}_t(\boldsymbol{X}^t))$ does not infer $P^s(\boldsymbol{Y}^s|\mathcal{T}_s(\boldsymbol{X}^s)) = P^t(\boldsymbol{Y}^t|\mathcal{T}_t(\boldsymbol{X}^t))$. However, according to Bayes rule in (12), instead of matching $P^t(\boldsymbol{Y}^t|\mathcal{T}_t(\boldsymbol{X}^t))$ and $P^s(\boldsymbol{Y}^s|\mathcal{T}_s(\boldsymbol{X}^s))$ directly, we can learn $\mathcal{T}_t$ under the objective of matching $P^s(\mathcal{T}_s(\boldsymbol{X}^s)|\boldsymbol{Y}^s)P^s(\boldsymbol{Y}^s)$ and $P^t(\mathcal{T}_t(\boldsymbol{X}^t)|\boldsymbol{Y}^t)P^t(\boldsymbol{Y}^t)$.

$$P(\mathcal{T}(\boldsymbol{X})|\boldsymbol{Y})P(\boldsymbol{Y}) \propto P(\boldsymbol{Y}|\mathcal{T}(\boldsymbol{X})). \tag{12}$$

In fact, as no label information in the target domain $P^t(\boldsymbol{Y}^t)$ is available, it is non-trivial to directly match

$P^t(\mathcal{T}_t(\boldsymbol{X}^t)|\boldsymbol{Y}^t)P^t(\boldsymbol{Y}^t)$ and $P^s(\mathcal{T}_s(\boldsymbol{X}^s)|\boldsymbol{Y}^s)P^s(\boldsymbol{Y}^s)$. However, as the label is a low-dimensional and discrete variable whose distribution is well-defined, it is more straightforward to match label distributions between domains compared with its conditional variant. Therefore, we take a step back and propose to estimate the re-weighted source domain label distribution $P^{Re}(\boldsymbol{Y}^s)$ so that it is similar to the unknown $P^t(\boldsymbol{Y}^t)$ in the target domain. In fact, we argue that matching the label distributions between two domains can also help the adaptation of the classifier, because at least part of $P^t(\mathcal{T}_t(\boldsymbol{X}^t)|\boldsymbol{Y}^t)P^t(\boldsymbol{Y}^t)$ and $P^s(\mathcal{T}_s(\boldsymbol{X}^s)|\boldsymbol{Y}^s)P^{Re}(\boldsymbol{Y}^s)$ is matched. Based on such an assumption, since the EM-distance has been adopted to match $P^t(\mathcal{T}_t(\boldsymbol{X}^t)$ and $P^s(\mathcal{T}_s(\boldsymbol{X}^s)$, we propose to embed the re-weighted label distribution $P^{Re}(\boldsymbol{Y}^s)$ into the procedure of matching the marginal feature distribution $P^s(\mathcal{T}_s(\boldsymbol{X}^s))$ and $P^t(\mathcal{T}_t(\boldsymbol{X}^t))$ in the adversarial training.

To estimate the re-weighted label distribution $P^{Re}(\boldsymbol{Y}^s)$, the following constraint should be considered:

$$\sum_{i=1}^{n_{cls}} P^{Re}(\boldsymbol{Y}^s = y_i) = 1, \tag{13}$$

where, y_i indicates the label of the i^{th} class. However, this constraint has already been considered in the implementation using the softmax function.

Finally, to estimate the re-weighted label distribution, if we directly replace the $P^s(\boldsymbol{Y}^s)$ by $P^{Re}(\boldsymbol{Y}^s)$ in the mini-max objective function $\mathcal{L}_{adv}$ in (11), a new one $\mathcal{L}^{Re}_{adv}$ is obtained in (14), where the network $\mathcal{D}$, $\mathcal{T}_t$ and the ratio vector $\boldsymbol{\beta}$ are trained in the following manner: $\mathcal{D}$ and $\boldsymbol{\beta}$ are trained in a cooperative way to estimate the EM-distance, while $\mathcal{T}_t$ is trained to minimize the EM-distance. From the perspective of implementation, $\boldsymbol{\beta}$ can be regarded as assigning different significance to images $\boldsymbol{x}^s$ in the source domain, so that the mini-batches in the two domains are sampled from similar distributions, which helps $\mathcal{D}$ and $\mathcal{T}_t$ to focus on matching $P^s(\mathcal{T}_s(\boldsymbol{x}^s))$ and $P^t(\mathcal{T}_t(\boldsymbol{x}^t))$.

$$\begin{aligned}
&\min_{\mathcal{T}_t}\max_{\mathcal{D},\boldsymbol{\beta}} \mathcal{L}^{Re}_{adv},\ where\\
\mathcal{L}^{Re}_{adv} =& \sum_{(\boldsymbol{x}^s,y^s)\sim P^s(\boldsymbol{X}^s,\boldsymbol{Y}^s)} \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s))P^s(\mathcal{T}_s(\boldsymbol{x}^s)|y^s)P^{Re}(y^s)\\
&- \mathop{\mathbb{E}}_{\boldsymbol{x}^t\sim P^t(\boldsymbol{X}^t)} \mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t))\\
=& \sum_{(\boldsymbol{x}^s,y^s)\sim P^s(\boldsymbol{X}^s,\boldsymbol{Y}^s)} \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s))P^s(\mathcal{T}_s(\boldsymbol{x}^s)|y^s)\boldsymbol{\beta}(y^s)P^s(y^s)\\
&- \mathop{\mathbb{E}}_{\boldsymbol{x}^t\sim P^t(\boldsymbol{X}^t)} \mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t))\\
=& \mathop{\mathbb{E}}_{(\boldsymbol{x}^s,y^s)\sim P^s(\boldsymbol{X}^s,\boldsymbol{Y}^s)} \boldsymbol{\beta}(y^s)\mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s)) - \mathop{\mathbb{E}}_{\boldsymbol{x}^t\sim P^t(\boldsymbol{X}^t)} \mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t))\\
&s.t. \| \bigtriangledown_{\mathcal{T}_t(\boldsymbol{x}^t)} \mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t))\|_2 \le 1,\\
&\| \bigtriangledown_{T_s(\boldsymbol{x}^s)} \mathcal{D}(\mathcal{T}_s(\boldsymbol{x}^s))\|_2 \le 1.
\end{aligned} \tag{14}$$

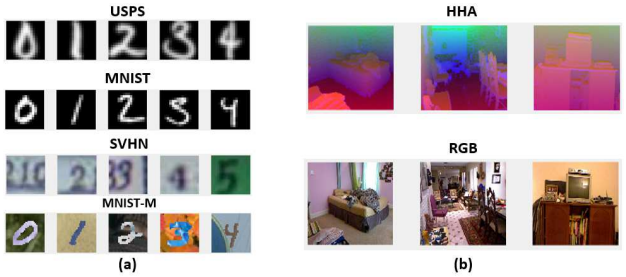

Figure 2. DA datasets: (a) four hand-written digit datasets; (b) cross-modality dataset including RGB and RGB-depth images.

3.3. Optimization in RAAN

As shown in Figure 1, RAAN is proposed to jointly minimize the cross entropy loss of the source domain samples and to reduce the divergence of the extracted feature distributions. First, we define the empirical estimate of the loss function $\mathcal{L}^{Re}_{adv}$ as follows:

$$\mathcal{L}^{Re}_{adv} = \frac{1}{n_s}\sum_{i=1}^{n_s}\mathcal{D}(\boldsymbol{\beta}(y^s_i)\mathcal{T}_s(\boldsymbol{x}^s_i)) - \frac{1}{n_t}\sum_{j=1}^{n_t}\mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t_j)). \tag{15}$$

Following on from the idea of controlling the 1-Lipschitz function of the domain discriminator network $\mathcal{D}$ [17], we explicitly constrain the gradient norm penalty term as follows:

$$\mathcal{L}_{gp} = \| \bigtriangledown_{\hat{\mathcal{T}}(\hat{\boldsymbol{x}})} \mathcal{L}^{Re}_{adv} - 1\|_2, \tag{16}$$

where $\hat{\mathcal{T}}(\hat{\boldsymbol{x}})$ is the weighted interpolation samples of $\mathcal{T}_t(\boldsymbol{x}^t)$ and $\mathcal{T}_s(\boldsymbol{x}^s)$. In summary, the total objective function in RAAN is formulated in the following adversarial manner:

$$\min_{\mathcal{T}_t,\mathcal{D},\boldsymbol{\beta}} -\mathcal{L}^{Re}_{adv} + \lambda_{gp}\mathcal{L}_{gp} + \lambda_{reg}\|\boldsymbol{\beta}\|_2, \tag{17}$$

$$\min_{\mathcal{T}_t} -\frac{1}{n_t}\sum_{j=1}^{n_t}\mathcal{D}(\mathcal{T}_t(\boldsymbol{x}^t_j)), \tag{18}$$

$$\min_{\mathcal{T}_s,CLS}\frac{1}{n_s}\sum_{i=1}^{n_s}\mathcal{L}_{CE}(CLS(\mathcal{T}_s(\boldsymbol{x}^s_i)), y^s_i), \tag{19}$$

where $\mathcal{L}_{CE}(CLS(\mathcal{T}_s(\boldsymbol{x}^s_i)), y^s_i)$ indicates the cross-entropy function with classifier CLS, feature vector $\mathcal{T}_s(\boldsymbol{x}^s_i)$ and its label y^s_i. Note that to train the networks stably, the source domain DCNN $\mathcal{T}_s$ is trained first while $\mathcal{T}_t$ and $\mathcal{D}$ are trained to match the feature distributions between $\mathcal{T}_t(\boldsymbol{x}^t)$ and $\mathcal{T}_s(\boldsymbol{x}^s)$ in an adversarial manner. In addition, to stably learn the ratio vector, we add the L2-norm of $\boldsymbol{\beta}$ as the regularization term in (17). λ_{gp} and λ_{reg} indicate the regularization weights of the gradient penalty term and the L2-norm of the ratio vector respectively.

4. Experiment and Results

In this section, RAAN is evaluated in two UDA tasks, specifically one between hand-written digit datasets and

the other between cross-modality datasets. For all the experiments, RAAN achieves competitive results compared with the state-of-the-art methods and outperforms them by a large margin when the distribution divergence is large between domains.

4.1. Adaptation Tasks and Dataset

The first UDA task adapts between four hand written digit datasets including MNIST[21], USPS [10], SVHN [31] and MNIST-M [12]. As shown in Figure 2, adaptation between these four datasets are of varying difficulty. MNIST and USPS are both composed of grey-scale images in a fairly well-controlled environment while images in MNIST-M are synthesized using the patches from BSDS500 dataset [1] as the background and the MNIST images as the foreground. To evaluate RAAN in reducing large domain discrepancies, SVHN is also explored which is composed of RGB images in more complicated real-world scenarios, e.g, misalignment of images and different light conditions. In addition, note that the sub-class instances between SVHN and the others are largely unbalanced.

To continue evaluating RAAN in reducing large domain shifts, the second adaptation task is designed using the NYU-D dataset [30], adapting from the indoor object images in RGB format to the depth variants encoded by the HHA format[18]. The 19-class dataset is extracted following the scheme in [37]. As shown in Figure 2, the domain shifts between images of RGB and HHA format are fairly large, mainly due to the low image resolutions and potential mis-alignments caused by the coarse cropping box. In addition, as shown in the instance number in Table 4, this dataset has unbalanced sub-class instances. Furthermore, it is challenging as the images from the target domain are in a completely different format from those in the source domain.

4.2. Adaptation in Hand-Written Digit Dataset

For the task of adapting between hand written digit datasets, the following four adaptation directions are chosen for the evaluation: from MNIST to USPS, from USPS to MNIST, from SVHN to MNIST and from MNIST to MNIST-M. For the first three adaptation tasks, we adopt a variant of LeNet as network $\mathcal{T}_s$, $\mathcal{T}_t$ and the domain discriminator network $\mathcal{D}$ is composed of three fully-connected layers activated by the rectified linear unit (ReLU) with output activation numbers of 512,512,1 respectively. For adapting from MNIST to MNIST-M, we adopt the basic model architecture of pixelDA but change their domain discriminator network to an OT-based objective function and embed the ratio vector $\boldsymbol{\beta}$. As for the experiment protocols, we utilized the one in [24] for adapting between MNIST and USPS, while for adaptation from SVHN to MNIST, we choose that in [37]. The protocol used for adapting from MNIST to MNIST-M is the same as that in [4] to permit fair comparisons.

To assess the reasons underlying RAAN's performance, we denote RAAN(+) and RAAN(-) as RAAN with and without the re-weighting scheme respectively. As shown in Table 1, when adapting between MNIST and USPS, compared with ADDA and Co-GAN, the proposed RAAN(-) and RAAN(+) achieved competitive results and RAAN(+) slightly outperforms RAAN(-). In the most difficult task, i.e., adapting from SVHN to MNIST, RAAN(-) and RAAN(+) achieved 80.7% and 89.1% respectively, outperforming the state-of-the-art ADDA by 4.7% and 13.1% respectively, while Co-GAN does not converge in this experiment. It seems that the weight-sharing approach utilized in Co-GAN is not capable of generating transferable images between disparate domains such as MNIST and SVHN. As RAAN(-) utilized the same DCNN architecture to ADDA's, RAAN(-)'s superior performance is mainly owing to the OT based objective function. We hypothesize that the OT based objective function is able to better reduce feature distribution divergence when the domains are disparate, e.g., SVHN and MNIST. In addition, based on the fact that RAAN(+) achieves superior performances to both ADDA and RAAN(-), we hypothesize that matching the label distribution helps adapt the classifiers, and embedding it into minimizing the EM distance of feature distributions can be regarded as two cooperative tasks.

For adaption from MNIST to MNIST-M, as shown in Table 2, RAAN achieves slightly better performance than pixelDA. In addition, as expected RAAN(-) and RAAN(+) achieve similar results since the label distribution of the two domains are quite similar. Although it has been argued that domain shift between MNIST and MNIST-M is large for a conventional DCNN based method[13], we argue that reducing the domain shift caused by the background images in MNIST-M is easier than reducing the one between MNIST and SVHN if a generative model is utilized. The possible reason may be that the domain shifts led by background patches in MNIST-M exhibit fewer details and variations than the one caused by the complex real-world variations present in SVHN. In addition, we argue that the adversarial training based generative model is good at generating such background patches. These consideration may suggest and explain the slight outperformance achieved by RAAN in this task, compared with RAAN's large improvement when adapting from SVHN to MNIST.

4.3. Adaptation in Cross-modality Dataset

In this section, RAAN is evaluated in the presence of large domain shifts that confront the adaptation from RGB images to RGB-depth images. To enable a fair comparison, we follow ADDA's experimental set-up [37] and utilized

Table 1. Recognition rates of adapting hand-written digit datasets; RAAN(+) and RAAN(-) indicate with and without the re-weighting scheme.

Methods	MNIST to USPS	USPS to MNIST	SVHN to MNIST
Source Only	0.725	0.612	0.593
Gradient Reversal[13]	0.771	0.730	0.739
Domain Confusion [38]	0.791	0.665	0.681
Co-GAN[22]	**0.912**	0.891	No Converge
ADDA [37]	0.894	0.901	0.760
RAAN(-)(Ours)	0.883	0.915	0.807
RAAN(+)(Ours)	0.89	**0.921**	**0.892**

Table 2. Recognition rates of adapting from MNIST to MNIST-M; RAAN(+) and RAAN(-) indicate with and without re-weighting scheme

Dataset	Source Only[4]	CORAL[36]	MMD[4]	DANN [13]	DSN [5]	PixelDA[4]	RAAN(+)/(-)(Ours)
MNIST to MNIST-M	0.636	0.577	0.769	0.774	0.832	0.982	**0.985**

Table 3. $\mathcal{A}$-Distance of Adversarial Training Method

Metric	Source Only	ADDA	RAAN(-)	RAAN(+)
$\mathcal{A}$-Distance	1.673	1.548	1.526	1.506

the VGG-16 architecture [35] for DCNNs $\mathcal{T}_s$ and $\mathcal{T}_t$. The domain discriminator network $\mathcal{D}$ is composed of three fully-connected layers activated by the Relu, with 1024,2048,1 outputs respectively.

As shown in Table 4, we report the sub-class classification accuracy achieved by RAAN(-) and RAAN(+), along with the re-weighted label distribution $P^{Re}(\boldsymbol{Y^s})$ yielded by RAAN(+) and the target one $P^t(\boldsymbol{Y^t})$. It can be observed from the overall recognition rates that RAAN(+) achieves an average of 34.3%, outperforming ADDA by 6.7% and RAAN(-) by 3.5%. In addition, RAAN(-) outperforms ADDA by 3.2%. For classes with less samples, RAAN(+) and RAAN(-) achieve better performances than ADDA. In fact, ADDA only achieved better performance in class 'chair' as that class has the largest number of samples. It can also be seen that RAAN(+) outperforms RAAN(-) not only from the overall recognition accuracy but also from how many classes the classifier can recognize (classes with the recognition rates more than 0%). This is potentially due to the fact that the re-weighting scheme increases the significance of instances from the sub-classes with a lower number of instances. This can be verified by comparing the number of sub-class instances with the estimated ratio vector $\boldsymbol{\beta}$ in Table 4.

4.4. Parameter Selection and Implementation

The experiments are conducted on a GPU cluster. For all experiments, we utilize the Adam optimizer, with the learning rate selected from the following set:$\{2e^{-5}, 5e^{-5}, 1e^{-4}, 2e^{-4}, 5e^{-4}, 1e^{-3}\}$. For the regularization weights, λ_{gp} and λ_{reg} are chosen from the following sets $\{1, 10, 50, 100\}$ and $\{0.1, 1, 10, 50, 100, 500\}$ respectively. We used the exponential decay, with decay factor of 0.99 for every 1000 iterations. All experiments are run 10 times, each for 100000 iterations and we report the average results. For adapting from MNIST to MNIST-M, the batch size is 32 while for others, the batch size is 128.

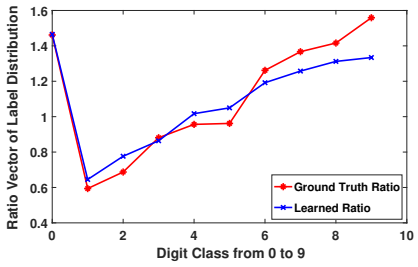

Figure 3. Ratio of label distribution between SVHN and MNIST; red line indicates the ground truth ratio, while blue one indicates the estimated ratio.

5. Analysis

In this section, we analyze the results presented in the previous sections, including the re-weighting scheme in adversarial training and the domain distribution divergence in both quantitative and qualitative ways. The evaluation is in the context of the most challenging scenario, which involves adapting from the SVHN to the MNIST.

5.1. Evaluate the Re-weighting Scheme

In Figure 3, we evaluate the re-weighting scheme by comparing the ground truth label ratio vector (red) and the learned one (blue). It can be seen that some ratios are accurate while others are not. However, the relative ratio trend of the learned ratio vector $\boldsymbol{\beta}$ follows that of the ground truth.

As the label distributions between SVHN and MNIST are largely mismatched, it will confuse the domain discriminator and the feature distributions will be matched in a biased manner. In addition, the mismatch of label distribution will directly give rise to the mismatch of classifiers as well. However, as shown in Figure 3, RAAN(+) successfully matches the distribution of labels by simply learning

Table 4. Adaptation results in cross-modality dataset; RAAN(+) and RAAN(-) indicate with and without re-weighting scheme

class	bathtub	bed	bookshelf	box	chair	counter	desk	door	dresser	garbage bin	lamp	monitor	night stand	pillow	sink	sofa	table	television	toilet	Overall
NO.instances	19	96	87	210	611	103	122	129	25	55	144	37	51	276	47	129	210	33	17	2401
Source Only	0.000	0.760	0.000	0.105	0.131	0.029	0.090	0.457	0.000	0.036	0.125	0.000	0.000	0.549	0.000	0.023	0.148	0.030	0.000	0.189
ADDA	0.000	0.469	0.000	0.005	**0.762**	0.194	0.016	0.519	0.040	0.018	0.007	0.000	0.000	0.083	0.000	0.062	0.138	0.000	0.000	0.276
RAAN(-) (ours)	0.000	**0.500**	**0.299**	0.000	0.629	0.019	0.000	0.271	0.000	0.000	0.000	0.000	0.000	**0.692**	0.000	0.000	0.043	0.000	0.000	0.308
RAAN(+) (ours)	0.000	0.104	0.000	**0.148**	0.703	**0.485**	0.000	**0.612**	0.000	0.000	0.007	0.000	0.000	0.638	**0.106**	0.000	**0.205**	0.000	0.000	**0.343**
$Est.\beta$	2.040	1.395	1.568	0.872	0.492	1.440	1.297	1.206	1.640	1.629	1.057	1.858	1.652	0.805	1.527	1.102	0.682	1.814	1.858	$--$

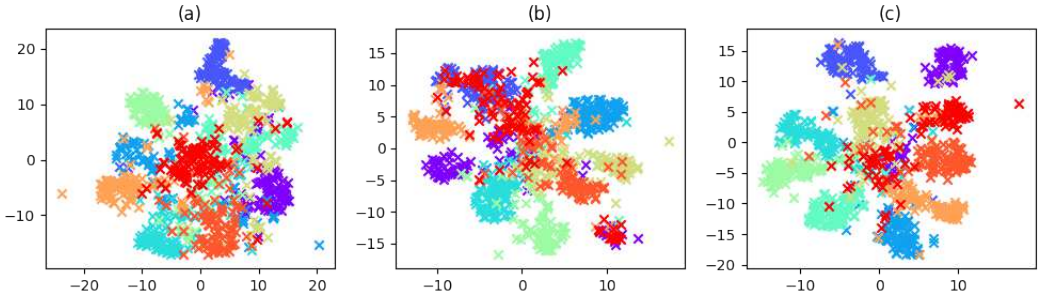

Figure 4. T-SNE plot of features when adapting from SVHN to MNIST; (a) No adaptation (b) Adaptation after ADDA (c) Adaptation after RAAN. We randomly select 1000 features samples from 10 classes, with 100 samples per class.

the ratio vector embedded in the adversarial training. Therefore, this can be regarded as the main reason for the 9% improvement achieved by RAAN(+) compared to RAAN(-) shown in Table 1. To sum up, matching the label distribution can better adapt the classifiers.

To understand the instance re-weighting scheme intuitively, it is implemented by assigning different significance to source domain instances. For example, as shown in Figure 3, the learned ratio of digit "0" is around 1.5, which means that in the adversarial training, each sample from digit "0" in SVHN dataset can be regarded as 1.5 samples.

5.2. Evaluate Distribution Divergence of Feature Embeddings

To analyze the distribution divergence in a quantitative way, we calculate the $\mathcal{A}$ distance suggested by the UDA community [3] [29], taking the input features extracted by various methods. Using the SVM classifier, we first calculate the generalization error θ of classifying the source and target domain features as a binary classification task. Then the $\mathcal{A}$ distance can be calculated as follows: $d = 2(1 - 2\theta)$. As shown in the Table 3, the $\mathcal{A}$ distances of feature embeddings with no adaptation, adapted by ADDA, OT based RAAN(-) and RAAN(+) progressively decrease. In the experiment, since RAAN uses the same DCNN architecture as ADDA's, compared with ADDA, the lower $\mathcal{A}$ distance achieved by RAAN(-) infers that feature distribution between domains can be better matched using RAAN(-). This may be due to the fact that the OT based EM distance is a better measure to reduce large distribution divergence than the geometry-oblivious JS divergence. In addition, compared to RAAN(-), the smaller $\mathcal{A}$ distance achieved by RAAN(+) indicates that matching the label distribution and the feature distribution are two cooperative tasks and this cooperative training may be the main reason for the lower $\mathcal{A}$ distance.

Finally, to measure the feature distribution divergence in a qualitative way, we utilized the T-SNE software package [28] to visualize the 2-D embedding of the extracted features. It can be seen in Figure 4 that the example points from the same class adapted by RAAN in Figure 4(c) are clustered closer than those in Figure 4(b) by ADDA and also those without the adaptation method in Figure 4(a).

6. Conclusions

In this paper, we propose a Re-weighted Adversarial Adaptation Network (RAAN) to reduce disparate domain feature distribution and adapt the classifier. Through an extensive set of experiments using various UDA datasets, RAAN outperforms state-of-the-art methods by a large margin when the domain distribution divergence is large. Therefore we argue that the OT based objective function in the adversarial training exhibits better properties to match distributions when they share less common support. In addition, embedding the estimation of the ratio vector into the adversarial training is capable of matching the label distribution between domains and further adapting the classifier. It is also shown that this scheme can help reduce feature distribution divergence.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011. 6

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017. 3

[3] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010. 8

[4] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *arXiv preprint arXiv:1612.05424*, 2016. 1, 2, 6, 7

[5] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *Advances in Neural Information Processing Systems*, pages 343–351, 2016. 7

[6] G. Celeux and G. Govaert. A classification em algorithm for clustering and two stochastic versions. *Computational statistics & Data analysis*, 14(3):315–332, 1992. 3

[7] C. Cortes, Y. Mansour, and M. Mohri. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pages 442–450, 2010. 3

[8] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. *arXiv preprint arXiv:1705.08848*, 2017. 1, 2, 3

[9] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1853–1865, 2017. 1, 3, 4

[10] J. S. Denker, W. Gardner, H. P. Graf, D. Henderson, R. Howard, W. Hubbard, L. D. Jackel, H. S. Baird, and I. Guyon. Neural network recognizer for hand-written zip code digits. In *Advances in neural information processing systems*, pages 323–331, 1989. 6

[11] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014. 1, 3

[12] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015. 1, 6

[13] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016. 1, 2, 6, 7

[14] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016. 2

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[16] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012. 1

[17] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017. 3, 5

[18] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014. 6

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[20] L. Kantorovitch. On the translocation of masses. *Management Science*, 5(1):1–4, 1958. 4

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6

[22] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016. 1, 2, 7

[23] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015. 1

[24] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, pages 2200–2207, 2013. 6

[25] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu. Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1410–1417, 2014. 2

[26] M. Long, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. *arXiv preprint arXiv:1605.06636*, 2016. 1, 2

[27] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016. 1

[28] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008. 8

[29] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009. 8

[30] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 6

[31] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011. 6

[32] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016. 1, 2

[33] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010. 1

[34] J. Shen, Y. Qu, W. Zhang, and Y. Yu. Adversarial representation learning for domain adaptation. *arXiv preprint arXiv:1707.01217*, 2017. 3

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1, 7

[36] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops*, pages 443–450. Springer, 2016. 7

[37] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017. 1, 2, 6, 7

[38] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 1, 7

[39] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *(IEEE) Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[40] C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008. 4

[41] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016. 1

[42] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. *arXiv preprint arXiv:1705.00609*, 2017. 1, 3

[43] Y. Yu and C. Szepesvári. Analysis of kernel mean matching under covariate shift. *arXiv preprint arXiv:1206.4650*, 2012. 3

[44] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang. Domain adaptation under target and conditional shift. In *International Conference on Machine Learning*, pages 819–827, 2013. 3