UNSUPERVISED VISUAL DOMAIN ADAPTATION: A PROBABILISTIC APPROACH

by

BEHNAM BABAGHOLAMI MOHAMADABADI

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Vladimir Pavlovic

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

JANUARY, 2020

ABSTRACT OF THE DISSERTATION

Unsupervised Visual Domain Adaptation: A Probabilistic Approach

by Behnam Babagholami Mohamadabadi

Dissertation Director: Vladimir Pavlovic

Artificial intelligent and machine learning technologies have already achieved significant success in various applications (computer vision, natural language processing, speech recognition, etc.). Such methods work well only under a common assumption that training and test data are drawn from the same distribution. However, the curse of domain mismatch arises when the test data and the training data come from different distributions. In such distribution changes, most statistical models need to be rebuilt, using newly collected training data. In many real world applications, it is expensive or even impossible to collect the required training data and rebuild the models. One of the ultimate goals of the open ended learning systems is to take advantage of previous experience/ knowledge in dealing with similar future problems. Two levels of learning can be identified in such scenarios. One draws on the data by capturing the pattern and regularities which enables reliable predictions on new samples. The other starts from an acquired source of knowledge and focuses on how to generalise it to a new target concept; this is also known as transfer learning which is going to be the main focus of this thesis.

This thesis will focus on a family of transfer learning methods applied to the task of visual object recognition, specifically image classification. The visual recognition

problem is central to computer vision research: many desired applications, from robotics to information retrieval, demand the ability to correctly identify categories, places, and objects. Transfer learning is a general term, and specific settings have been given specific names: when the learner has access to only unlabeled data from the target domain (where the model should perform) and labeled data from a different domain (the source), the problem is called unsupervised domain adaptation (DA).

The thesis focuses on four methods for this setting. The first one proposes a probabilistic latent variable model by learning projections from each domain to a latent (shared) space jointly with the classifier in the latent space, which simultaneously minimizes the domain disparity while maximizing the classifier's discriminative power. Furthermore, the non-parametric nature of our adaptation model makes it possible to infer the latent space dimension automatically from data.

The second method is based on the Gaussian Process (GP): The GP allows us to induce a hypothesis space of classifiers from the posterior distribution of the latent random functions, turning the learning into a large-margin posterior separation problem.

The Third method is based on GANs: We introduce an adversarial discriminative discrepancy measure which takes advantage of auxiliary information available in the source and the target domains to better align the source and target distributions. Specifically, we leverage the cohesive clustering structure within individual data manifolds, associated with different tasks, to improve the alignment.

The last one addresses domain adaptation for multiple target domains. We propose an information theoretic approach for domain adaptation in the novel context of multiple target domains with unlabeled instances and one source domain with labeled instances. Our model aims to find a shared latent space common to all domains, while simultaneously accounting for the remaining private, domain-specific factors. Disentanglement of shared and private information is accomplished using a unified information-theoretic approach, which also serves to establish a stronger link between the latent representations and the observed data.

We conduct experiments on a wide range of image classification tasks. We test our proposed methods and show that, in all cases, leveraging knowledge from a related

domain can improve performance when there are no labels available for direct training on the new target data.

Acknowledgements

First and foremost, I would like to thank my advisor, Professor Vladimir Pavlovic, for his guidance, immense knowledge, encouragement, and patience throughout my PhD study. All these years of working under his supervision truly were valuable and rewarding experience. Many thanks, as well, to the thesis committee. I want to thank my alumni lab mates, Sejong, Jongpil, Hai, and the current lab mates Pritish, Fangda, and Gang for the inspiring discussions and the fun we had throughout these years. I would also like to thank my friends and fellow PhD candidates for their companionship and support. Without them, I would be another lonely PhD student. I received valuable experience as a teaching assistant for both undergraduate and graduate courses throughout my PhD study. My students came from diverse backgrounds and featured different strengths and interests. They brought many points of view to the discussions and investigations, as they posed different questions for the same problem. Last, but not least, I would like to thank my family for their love, advice and encouragement.

Dedication

To my parents: without their encouragement, I could not have started my study.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

In the context of Computer Vision (CV), an image in the most basic representation is defined through a matrix of its pixels intensity values and the semantic organisation of an image database is known as classification where an ideal image classifier should be able to exploit complex high dimensional feature representations even when only a few labelled training samples are available. In most classification scenarios, it is expensive to acquire vast amounts of labelled training samples in order to provide classifiers with a good coverage of the feature space. One possible solution to tackle this problem is to synthesise images of training data using computer graphics techniques (e.g. [102]), however, their appearance may not be realistic and it is not possible to model all possible backgrounds. Using crowd sourcing [16], but the annotations obtained are either costly or unreliable. Ideally, an image classifier should be initially capable of detecting similarities between data distributions and subsequently facilitates the exploitation of the required knowledge from all the previously trained reliable models, just as human can exploit previous experience when learning some similar concepts.

Another major challenge is the sampling bias problem [115]. Conventional statistical machine learning revolves on an simplified assumption that the training data, from which the algorithms learn, are drawn i.i.d. from the same distribution as the test data, to which the learned models are applied. This assumption and the corresponding algorithms are fundamentally restrictive, being frequently challenged in numerous situations. For example, a pedestrian detection system on automobiles encounters very different data when weather patterns change, when cameras age, or simply when people drive to new locations. In other words, the training and test data are often mismatched.

Figure 1.1: Sample images of certain classes (stapler, water bottle, cellphone, spray can) as seen in a wed dataset, Imagenet [93], on the left, and in a real-life like dataset (JHUIT-50 [67], HelloiCubWorld [31]), on the right. Note that while they should be representing the same things, the images have very little in common

As a result, practical autonomous systems inevitably suffer from the sampling bias. The systems are often deployed to new target environments for which it is unrealistic to attempt to reproduce all sorts of the target environment when one develops the systems, not to mention that real life environments are often not lab-reproducible. Hence, it is highly desirable to have a new statistical machine learning paradigm to explicitly deal with the mismatches in data.

A simple intuition of this problem is shown in Figure. 1.1. The images on the left belong to a dataset which was mined from the web, the images on the right were captured by a robot: the framing, the lightning conditions, the resolution, the background clutter are all different. If our model is trained with the images on the left it is easy to understand why it will perform poorly in the real world. This is a pretty typical setup: we wanted to perform recognition on a set of classes, we used the web to download some training data (we will call it the *source*) and found out that the model did not work well on real world data. During the deployment of our system we gathered some unlabeled data (our *target*) for free. We know that the labeled source and unlabeled target share the same classes and we would like for our model to perform well on both, ignoring their specific biases. This problem is formally known as that of *unsupervised domain adaptation* (we will define it more rigorously in next Chapter).

This thesis concentrates on addressing this challenge in the framework of unsupervised

domain adaptation. Domain Adaptation is at its core the quest for principled algorithms enabling the generalization of visual recognition methods. Given at least a source domain for training, the goal is to achieve recognition results as good as those achievable on source test data on any other target domain, in principle belonging to a different probability distribution, without having prior access to labeled images. Solving this problem will represent a major step towards one of the key goals of computer vision, i.e. having machines able to answer the fundamental question 'what do you see?' in the wild.

1.2 Contributions

Working in the context of visual image recognition, we contribute to the field of unsupervised domain adaptation with four novel and distinct techniques. We then perform a qualitative and quantitative experimental evaluation of the proposed solutions, to quantify their effectiveness and robustness demonstrating that employing the proposed solutions assures better performances than simply training on the available data.

Specifically we present:

- **A probabilistic latent variable model to address unsupervised domain adaptation** [39]. Specifically, we tackle the task of categorization of visual input from different domains by learning projections from each domain to a latent (shared) space jointly with the classifier in the latent space, which simultaneously minimizes the domain disparity while maximizing the classifier's discriminative power. Furthermore, the non-parametric nature of our adaptation model makes it possible to infer the latent space dimension automatically from data. We also develop a novel regularized Variational Bayes (VB) algorithm for efficient estimation of the model parameters.

- **A systematic and effective way to achieve hypothesis (classifier) consistency over both source and target domains using Gaussian processes (GP)** [61]. The GP allows us to induce a hypothesis space of classifiers from the posterior distribution of the latent random functions, turning the learning into a large-margin posterior separation problem, significantly easier to solve than previous approaches. We formulate a learning objective that effectively influences the posterior to minimize the maximum discrepancy. This is shown to be equivalent to maximizing margins and minimizing uncertainty of the class predictions in the target domain.

- **A discriminative discrepancy measure which takes advantage of auxiliary information available in the source and the target domains to**

better align the source and target distributions [40]. Specifically, we leverage the cohesive clustering structure within individual data manifolds, associated with different tasks, to improve the alignment. This structure is explicit in the source, where the task labels are available, but is implicit in the target, making the problem challenging. We address the challenge by devising a deep DA framework, which combines a new task-driven domain alignment discriminator with domain regularizers that encourage the shared features as task-specific and domain invariant, and prompt the task model to be data structure preserving, guiding its decision boundaries through the low density data regions.

- **An information theoretic approach for domain adaptation in the novel context of multiple target domains with unlabeled instances and one source domain with labeled instances** [41]. Our model aims to find a shared latent space common to all domains, while simultaneously accounting for the remaining private, domain-specific factors. Disentanglement of shared and private information is accomplished using a unified information-theoretic approach, which also serves to establish a stronger link between the latent representations and the observed data. The resulting model, accompanied by an efficient optimization algorithm, allows simultaneous adaptation from a single source to multiple target domains.

1.3 Outline

Reflecting the structure of this thesis, **Chapter 2** will provide a formal definition of domain adaptation in the context of visual object recognition, present an overview of relevant works and introduce the datasets we will use for the experimental evaluation. The domain adaption literature (Section 2.1) will present shallow, deep and adversarial methods.

Chapter 3 will delve into the details of the four unsupervised domain adaptation methods we propose in this thesis.

The thesis concludes with a summary discussion and remarks on possible future directions of research in **Chapter 4**.

1.4 Notations

Notions and their descriptions used in this thesis are summarized in Table 1.1.

Table 1.1: Notations and their descriptions.

Notation	Description
$\mathcal{X}$	image space
$\mathcal{Y}$	label space
$\mathcal{Z}$	feature space
$\mathbb{E}$	Expectation operator
N_s, N_t	number of source/target samples
N	number of all samples, $N_s + N_t$
M	number of domains
d	the input dimension
C	the number of classes
class $C+1$	unknown target class
$P_S(x,y), P_T(x,y)$	source/target distribution
$P_S(x)$,$P_T(x)$	source/target marginal distribution
$\mathcal{D}_S = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$	set of source samples
$\mathcal{D}_t = \{x_i^t\}_{i=1}^{N_t}$	st of target samples
X^s	data matrix $[x_1^s, ..., x_{N_s}^s] \in \mathbb{R}^{d \times N_s}$, source samples
Y^s	label matrix $[y_1^s, ..., y_{N_s}^s] \in \mathbb{R}^{C \times N_s}$, source samples
X^t	data matrix $[x_1^t, ..., x_{N_t}^t] \in \mathbb{R}^{d \times N_t}$, target samples
$\mathbb{D}$	domain label matrix $[d_1, ..., d_N] \in \mathbb{R}^{M \times N}$
Z^s	feature matrix $[z_1^s, ..., z_{N_s}^s] \in \mathbb{R}^{d \times N_s}$, source samples
Z^t	feature matrix $[z_1^t, ..., z_{N_t}^t] \in \mathbb{R}^{d \times N_t}$, target samples
$\phi(\cdot), \mathbb{K}(\cdot,\cdot)$	kernel feature map and kernel function induced by $\phi(\cdot)$
$G(\cdot)$	encoder (feature extractor)
$F(\cdot)$	2-dimensional Binary discriminator
$D(\cdot)$	$(C+1)$ dimensional Multi-class discriminator
$h(\cdot)$	$(C+1)$ dimensional multi-class classifier
α	K-dimensional binary vector for latent features
ϵ^s/ϵ^t	zero-mean Gaussian noises for source/target samples
$\mathbf{I}_d$	$d \times d$-dimensional Identity matrix
γ_s/γ_t	precision values for Gaussian noises ϵ^s/ϵ^t
$\text{Ber}(\pi_k)$	bernoulli distribution with parameter π_k
$H(\cdot)$	entropy operator
$q(\cdot)$	variational posterior distribution
$\mathcal{H}$	hypothesis space of classifiers
$e_T(h)$	error rate of the classifier h on target samples
$e_S(h)$	error rate of the classifier h on source samples
$Tr(\cdot)$	matrix Trace operator
$det(\cdot)$	matrix determinant operator
B_S/B_T	size of mini-batch for source/target samples
$a, b, \lambda, \lambda_r, \lambda_c, \lambda_d$	hyper-parameters
$Q(z\|x)$	conditional distribution of latent features given the samples
$P(z)$	prior distribution over latent features
$Q(z)$	aggregated posterior distribution over latent features
$\mathcal{D}(p\|\|q)$	statistical divergence between two distributions p and q
$I(x;z)$	mutual information between two random variables x and z
z_s/z_p	latent shared/private features
G_s/G_p	shared/private encoder
R	decoder

Chapter 2

Background and Related Work

This chapter covers related work on unsupervised domain adaptation problem (in the context of image classification) where it begins with the problem formulation, continues with a review of related work and finally presents the datasets on which the proposed algorithms will be tested.

2.1 Computer Vision: Domain Adaptation

2.2 Unsupervised Domain Adaptation

Domain adaptation refers to the problem of leveraging labeled task data in a source domain S to learn an accurate model of the same tasks in a target domain T where the labels are unavailable or very scarce [115]. Due to many factors (e.g., illumination, pose, and image quality), there is always a distribution change or domain shift between two domains that can degrade the performance, as shown in Figure 2.1.

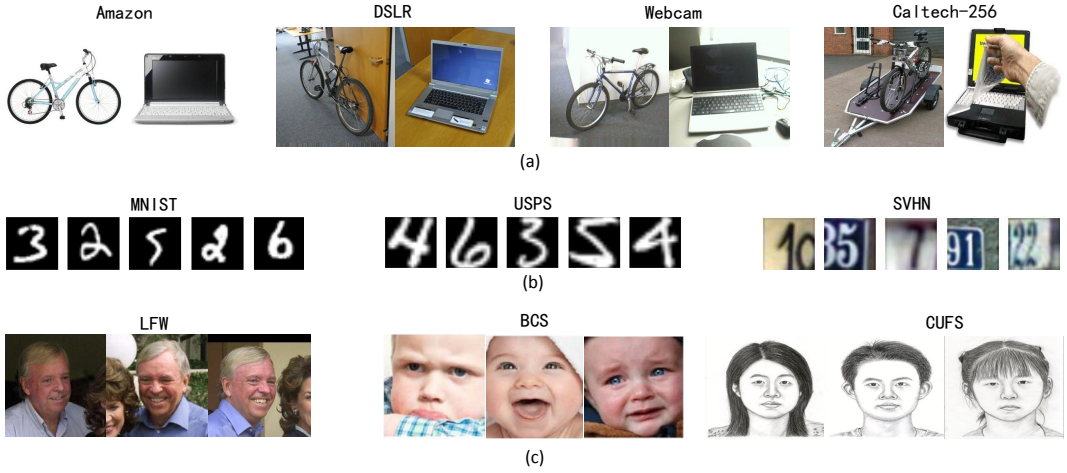


Figure 2.1: (a) Some object images from the "Bike" and "Laptop" categories in Amazon, DSLR, Webcam, and Caltech-256 databases. (b) Some digit images from MNIST, USPS, and SVHN databases. (c) Some face images from LFW, BCS and CUFS databases. Realworld computer vision applications, such as face recognition, must learn to adapt to distributions specific to each domain.

More specifically, while the tasks have identical label sets $Y^s = Y^t$ they possess (slightly) different conditional distributions $P_S(y|x) \sim P_T(y|x)$. The domains are different in terms of marginal data distribution $P_S(x) \neq P_T(x)$ in image spaces. Our goal is to estimate a classifier from source and target that can be used to classify sample points from the target domain. Domain adaptation has been studied in two main settings: one is the semi-supervised case, where the target presents few labeled data, while the other is the unsupervised case that considers only unlabeled examples for the target. In both cases, the source set is generally rich in labeled samples. In this thesis we will focus on the unsupervised case.

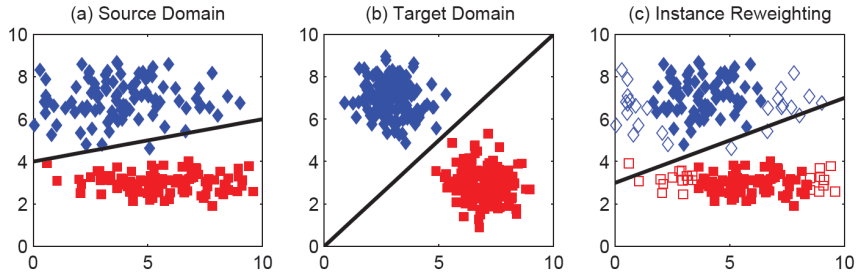


Figure 2.2: Illustration of the effect of instance re-weighting samples on the source classifier.

2.2.1 Literature survey

In recent years, numerous unsupervised domain adaptation methods have been proposed. Traditional approaches addressed the problem of reducing the discrepancy between the source and the target distributions by considering two main strategies. The first is based on instance re-weighting [21, 69, 116, 42], where the source samples are weighted according to their similarity to the target data. The re-weighted samples are then used to learn a classification model for the target domain (Figure 2.2). The main issue with these type of approaches is that they do not offer the modeling flexibility of incorporating the domain knowledge of data representations. Instance weighting is not flexible in modeling the intrinsic structures in data.

In some application domains, the data favor some special structures which could ease the adaptation of classifiers. For example, in computer vision, the data often have low-rank or manifold properties. To exploit such structures, an alternative line of research works on learning feature representations for domain adaptation.

The majority of the approaches try to bridge the gap between the source and target in a joint feature space both considering shallow models [43, 76, 33] and deep architectures [75, 34, 38, 15], so that a task classifier trained on labeled source data can be effectively transferred to the target. A feature representation is domain-invariant if the features follow the same distribution regardless of whether the input data is from the source or target domain. If a classifier can be trained to perform well on the source data using domain-invariant features, then the classifier may generalize well to the target domain

since the features of the target data match those on which the classifier was trained. The general Mechanism of these methods is illustrated in Figure 2.3.

One of the first such approaches is the Transfer Component Analysis (TCA) [85]. The main idea is to find a low-dimensional linear transformation such that the source and target domains are as close as possible in their marginal distributions, while maintaining the intrinsic structure of the original domains. The latter is achieved by incorporating a local geometry (manifold) preserving regularization term into the TCA's objective function. Likewise, [94] proposed a metric learning-based DA method with cross-domain constraints. This method learns a symmetric transformation to map source and target domain samples onto a new domain invariant space. [46] proposed an feature alignment method for DA based on the Sampling Geodesic Flow (SGF) that exploits the geodesic distance between the source and target subspaces.

Gong et al. [43] improved the SGF technique using the whole geodesic curve (considers all of the subspaces along the geodesic path) connecting the source and the target subspace on the Grassmann manifold instead of sampling a few points on the geodesic.

Long et al. [76] proposed Transfer Sparse Coding (TSC) which learns robust sparse representations by penalizing the distances between the sample means in the objective function to bring the domains closer.

Likewise, [109] proposed a simple but effective method for unsupervised DA called Correlation Alignment (CORAL), which minimizes domain shift by aligning the second-order statistics of source and target distributions.

In overall, methods in this category differ in how they align the domains. In this regard, an important challenge is how to measure the discrepancy between the two domains. Many domain discrepancy measures have been proposed in previous **DA** studies, such as the moment matching-based methods [79, 15, 85, 127, 125], and adversarial methods [117, 14, 99, 128, 35]. Moment matching-based methods use Maximum Mean Discrepancy (**MMD**) [106] to align the distributions by matching all their statistics. Inspired by Generative Adversarial Networks (GAN) [44], adversarial divergences train a domain discriminator to discern the source from the target, while an encoder feature extractor is simultaneously learned to create features indistinguishable

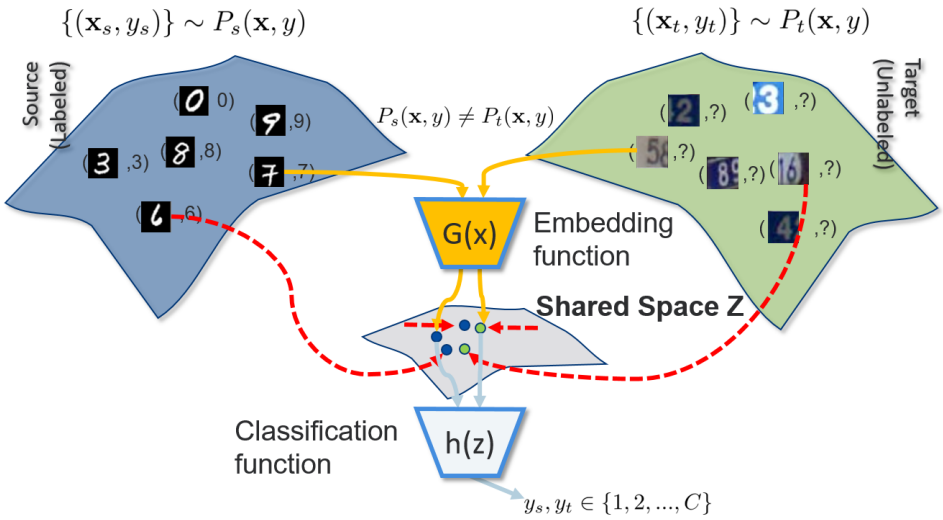


Figure 2.3: Shared feature learning for domain adaptation.

across the source and the target, confusing the discriminator.

Domain Adaptation using MMD

Maximum mean discrepancy (MMD) [49] is a two-sample statistical test of the hypothesis that two distributions are equal based on observed samples from the two distributions. The test is computed from the difference between the mean values of a smooth function on the two domains'samples. If the means are different, then the samples are likely not from the same distribution. The smooth functions chosen for MMD are unit balls in characteristic reproducing kernel Hilbert spaces(RKHS) since it can be proven that the population MMD is zero if and only if the two distributions are equal [49] (Figure 2.4). Specifically, given two sets of source/target samples, the MMD measures the distance between the mean of the two sets after mapping each sample to a RKHS:

$$MMD^2(Z^s, Z^t) = \left\| \sum_{i=1}^{N_s} \frac{\Phi(z_i^s)}{N_s} - \sum_{j=1}^{N_t} \frac{\Phi(z_j^t)}{N_t} \right\|^2, \tag{2.1}$$

where Z^s/Z^t denote the source/target features in shared space,N_s, N_t denote the number of source/target samples, $\Phi(.)$ denotes the mapping from feature space to RKHS. In practice, this mapping is typically unknown. By expanding Eq. 2.1, and using the kernel trick to replace the inner products by their kernel values, we rewrite the squared MMD,

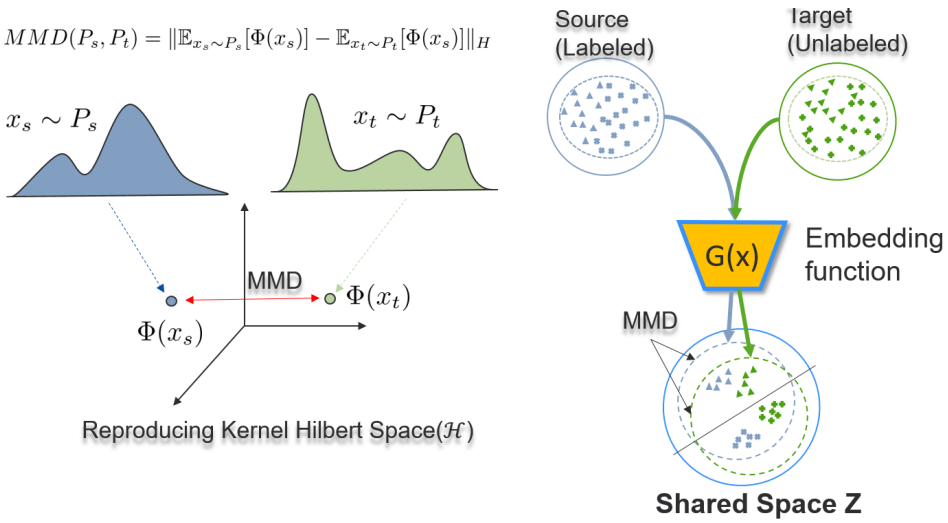


Figure 2.4: Domain Adaptation using MMD distance.

leading to the following regularizer:

$$\mathcal{L}(Z^s, Z^t) = \sum_{i,j} \frac{\mathbb{K}(z_i^s, z_j^s)}{N_s^2} - 2\sum_{i,j} \frac{\mathbb{K}(z_i^s, z_j^t)}{N_s N_t} + \sum_{i,j} \frac{\mathbb{K}(z_i^t, z_j^t)}{N_t^2},$$

where $\mathbb{K}(.,.)$ denotes the kernel function.

Instead of aligning the source and target domains in a (low) dimensional manifold, a few works attempted to reduce the domain mismatch by expanding the source and target features in a non-parametric fashion using the notion of Reproducing Kernel Hilbert Spaces (RKHS). The main assumption here is that in RKHS the domains can be brought together more easily compared to parametric (fix-dimension) transformations. Specifically, [6, 7] proposed the Domain Invariant Projection (DIP) method that compares the domain distributions in RKHS, while constraining the transformation to be orthogonal.

More recently, [56] proposed a DA scheme to construct a RKHS using the Mahalanobis metric in the target space. This is achieved by simultaneously learning the projections from the source and target domains to RKHS, by minimizing a notion of domain distance while maximizing a measure of discriminatory power of RKHS.

By the Taylor expansion of the Gaussian kernel, MMD can be viewed as minimizing the distance between the weighted sums of all raw moments [70]. The interpretation of MMD as moment matching procedures motivated Zellinger et al. [127] to match

the higher-order moments of the domain distributions, which we call central moment discrepancy (CMD). An empirical estimate of the CMD metric for the domain discrepancy in the activation space $[a, b]^N$ is given by

$$CMD_K(X^s, X^t) = \frac{1}{(b-a)}\left\|E(X^s) - E(X^t)\right\|_2 + \sum_{k=2}^{K} \frac{1}{|b-a|^k}\left\|C_k(X^s) - C_k(X^t)\right\|_2 \tag{2.2}$$

where $C_k(X) = E((x - E(X))^k$ is the vector of all k^{th}-order sample central moments and $E(X) = \frac{1}{|X|}\sum_{x \in X} x$ is the empirical expectation.

Based on our Misaligned-Feature-Norm Hypothesis [124], **MMFND** proposed the Maximum Mean Feature Norm Discrepancy to characterize the mean-feature-norm distance between the two distributions and verify whether bridging this statistical domain gap can result in appreciable transfer gains.

Despite of the popularity of MMD, it is notoriously hard to choose the optimal kernel(s) for MMD (e.g., how to set the bandwidth in Gaussian RBF) in domain adaptation, considering that there are not labeled data in the target domain for cross-validation. There exist some attempts to tackling this issue [49]. However, they are often limited to specific application scenarios.

Adversarial Domain Adaptation

Recently, great success has been achieved by the GAN method [44], which its goal is to generate realisic images via an adversarial process. GAN consists of two models: a generative model G that extracts the data distribution and a discriminative model D that distinguishes whether a sample is from the generator G or a given dataset by assigning a binary label to the sample. The models are trained on the label prediction loss in a mini-max fashion: simultaneously optimizing G to minimize the loss while also training D to maximize the probability of assigning the correct label (Figure 2.5):

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.3}$$

In DA, this idea has been employed to ensure that the model cannot distinguish between

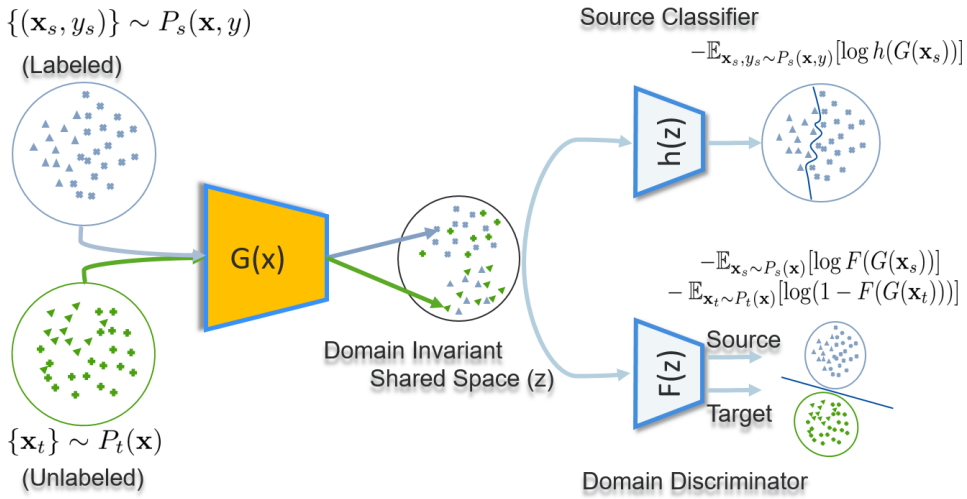


Figure 2.5: Adversarial domain adaptation.

the source and target domains. [128] proposed a unified framework for adversarial-based approaches and summarized the existing approaches according to whether to use a generator, which loss function to employ, or whether to share weights across domains. Overally speaking, adversarial-based approaches can be categorized into two subsettings: generative models and non-generative models.

Generative Models

Synthetic target data with ground-truth labels are alternatives to address the issue of a lack of training data. First, with the help of source data, generators render unlimited quantities of synthetic target data, which are paired with source data to share labels or appear as if they were sampled from the target domain while maintaining labels. Then, the generated samples with labels are used to train the target model as if no DA were required. Adversarial-based approaches with generative models are able to learn such a transformation in an unsupervised manner based on GAN idea.

The core idea of CoGAN [73] is to generate synthetic target data that are paired with synthetic source data. The model contains two GANs: GAN_1 for generating source data and GAN_2 for generating target data. The weights of the first few layers of the generators and the last few layers in the discriminators are tied. This weight-sharing constraint allows CoGAN to achieve a domain-invariant feature space without correspondence

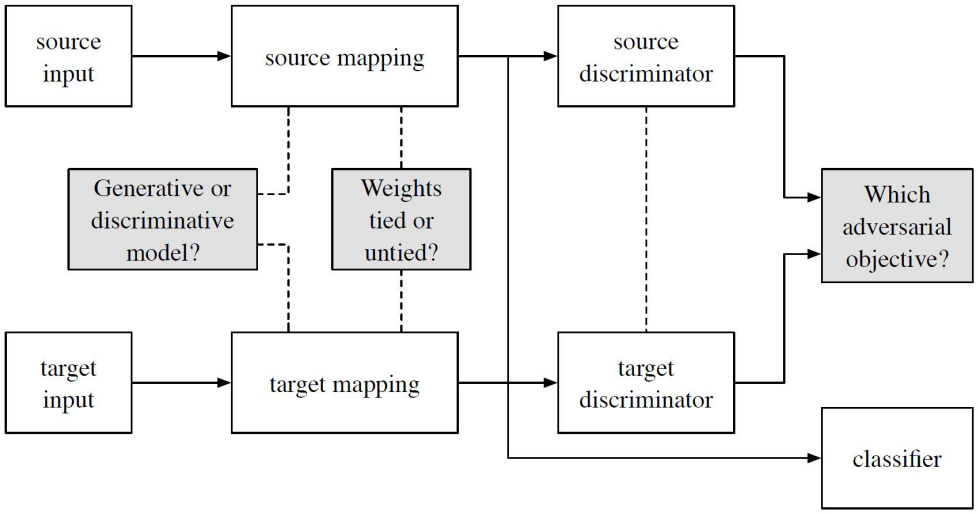


Figure 2.6: Generalized architecture for adversarial domain adaptation. Existing adversarial adaptation methods can be viewed as instantiations of a framework with different choices regarding their properties.

supervision. A trained CoGAN can adapt the input noise vector to paired images that are from the two distributions and share the labels. Therefore, the shared labels of synthetic target samples can be used to train the target model.

Yoo et al. [126] used GAN to transfer knowledge from the source domain to pixel-level target images. A domain discriminator ensures the invariance of content to the source domain, and another real/fake discriminator helps the generator to produce similar images to the target domain.

Shrivastava et al. [103] proposed a combined simulated and unsupervised learning method that uses a combined objective of minimizing an adversarial loss and a self-regularization loss, where the goal is to make synthetic images look more realistic using unlabeled target data.

Unlike other works in which the generator is conditioned only on a noise vector or source images, Bousmalis et al. [15] developed a model that exploits GANs conditioned on both noise and source images. The category classifier T is trained to predict class labels of both source and synthetic images, while the discriminator is trained to predict the domain labels of target and synthetic images. Moreover, to expect synthetic images with similar foregrounds and different backgrounds from the same source images, a content similarity loss is proposed to penalize large differences between source and synthetic

images for foreground pixels only by a masked pairwise mean squared error [30].

Non-Generative Models

The key of (deep) feature based DA is learning domain-invariant representations from source and target samples by which the distribution of both domains can be similar enough such that the classifier can be directly used in the target domain even if it is trained on source samples. Therefore, whether the features are domain-invariant or not is crucial to transferring knowledge. Inspired by GAN, domain confusion loss, which is produced by the discriminator, is introduced to improve the performance of deep DA without generators. The domain adversarial neural network (**DANN**) [35] first introduced a gradient reversal layer (GRL) that reversed the gradients of the domain discriminator in order to encourage domain confusion. The network consists of shared feature extraction layers and two classifiers. DANN minimizes the domain confusion loss (for all samples) and label prediction loss (for source samples) while maximizing domain confusion loss via the use of the GRL.

ADDA [117] recently proposed an adversarial framework for directly minimizing the distance between the source and target encoded representations (shared features). A discriminator and (target) encoder are iteratively optimized in a two-player game, where the goal of the discriminator is to distinguish the target features from the source features, with the goal of the encoder being to confuse the discriminator. ADDA minimizes the source and target representation distances through iteratively minimizing these following functions, which is most similar to the original GAN:

$$\min_{G,h} \mathcal{L}_{cls}(X^s, Y^s) = -\mathbb{E}_{(x^s,y^s)\sim P_S(x,y)} \sum_{c=1}^{C} \mathbf{1}_{[c=y^s]} \log h(G(x^s))$$

$$\min_{D} \mathcal{L}_{advD}(X^s, X^t, G) = -\mathbb{E}_{x^s\sim P_S(x)}[\log D(G(x^s))] - \mathbb{E}_{x^t\sim P_T(x)}[\log(1 - D(G(x^t)))]$$

$$\min_{G} \mathcal{L}_{advM}(G) = -\mathbb{E}_{x^t\sim P_S(x)}[\log D(G(x^t))] \tag{2.4}$$

where the mapping G is learned from the source and target data, X^s and X^t. h represents a classifier working on the source domain. The first classification loss function $\mathcal{L}_{cls}$ is optimized by training the source model using the labeled source data. The second

function $\mathcal{L}_{advD}$ is minimized to train the discriminator, while the third function $\mathcal{L}_{advM}$ is learning a representation that is domain invariant.

The **DupGAN** [57] proposed a **GAN**-like model with duplex discriminators to restrict the latent representation to be domain invariant, with its category information preserved. Saito et al. [96] further introduce two classifiers as a discriminator to avoid ambiguous features near the class boundaries. By deploying two classifiers, the method therein employs the adversarial learning techniques to detect the disagreement across classifiers, such that the encoder is able to minimize this discrepancy on target samples.

The **ARCA** [24] proposed a method that trains a shared embedding to align the joint distributions of inputs (domain) and outputs (classes), making any classifier agnostic to the domain. Joint alignment ensures that not only the marginal distributions of the domain are aligned, but the labels as well. A novel objective function is introduced that encourages the class-conditional distributions to have disjoint support in feature space. Adversarial regularization was also exploited to improve the performance of the classifier on the domain for which no annotated data is available.

Drop to Adapt (**DTA**) [66] leverages adversarial dropout to learn strongly discriminative features by enforcing the cluster assumption on target domain by pushing the decision boundary away from the target domain's features. More precisely, to support various tasks, **DTA** introduces element-wise and channel-wise adversarial dropout operations for fully-connected and convolutional layers, respectively.

CAT [27] proposed Cluster Alignment with a Teacher for unsupervised domain adaptation, which can effectively incorporate the discriminative clustering structures in both domains for better adaptation. Technically, **CAT** leverages an implicit ensembling teacher model to reliably discover the class-conditional structure in the feature space for the unlabeled target domain. Then CAT forces the features of both the source and the target domains to form discriminative class-conditional clusters and aligns the corresponding clusters across domains.

In addition to the adversarial distribution matching oriented algorithms, pseudo-labels or conditional entropy regularization are also adopted in literature [95, 101, 128, 20]. Sener et al. [101] construct a k-NN graph of target points based on a predefined similarity

Dataset	Images	classes	Domains	Description
Digit	140,000	10	4	mnist [65], svhn [84], usps [120], mnist-m [35]
Traffic Signs	150,000	43	2	Synthetic [83] and German Traffic Signs [107]
Office31 [94]	2500	10	3	Amazon, DSLR, Webcam
Office-Caltech [43]	4700	10	4	Amazon, DSLR, Webcam, Caltech
Multi-Pie [52]	120,000	6	5	face images taken from different angles
Visda [87]	280,000	12	2	Synthetic and Real Images
PACS [68]	10000	7	4	Photo, Art, Painting, Cartoon

Table 2.1: Statistics of the datasets.

graph. Pseudo-labels are assigned to target samples via their nearest source neighbors, which allows end-to end joint training of the adaptation loss. Saito et al. [95] employ the asymmetric tri-training, which leverages target samples labeled by the source-trained classifier to learn target discriminative features. Zhang et al. [128] iteratively select pseudo-labeled target samples based on their proposed criterion and retrain the model with a training set including pseudo-labeled samples. However, these methods based on pseudo-labeled target samples have a critical bottleneck where false pseudo-labels can mislead learning of target discriminative features, leading to degraded performance.

2.2.2 Datasets

The following section presents the datasets commonly used by DA methods and the protocols which will be used to evaluate our DA methods proposed in chapter 3. The statistics of the datasets is available in Table. 2.1. In the following we describe each dataset in detail.

Digits like datasets

The following datasets are fairly simple, with moderate intra class variability. They are commonly used to evaluate the modern DA models.

- **MNIST** [65] is the dataset on which the first convolutional neural network was trained. It has a training set of $60k$ examples, and a test set of $10k$ examples. It is a subset of a larger set available from NIST. The digits have been size-normalized

Figure 2.7: Samples from the digits datasets. From left to right: MNIST, SVHN, MNIST-M and USPS.

and centered in a fixed-size image. The images contain a single digit numbers on a black background.

- **MNIST-M** [35] is a variant where the background is substituted by a randomly extracted patch obtained from color photos of BSDS500 [35].

- **USPS** [120] is a digit dataset automatically scanned from envelopes by the U.S. Postal Service containing a total of $9,298$ 16×16 pixel grayscale samples; the images are centered, normalized and show a broad range of font styles.

- **SVHN** [84] is the challenging real-world Street View House Number dataset. It contains over 600k 32×32 pixel color samples, while we focused on the smaller version of almost 100k cropped digits. Besides presenting a great variety of shapes and textures, images from this dataset often contain extraneous numbers in addition to the labeled, centered one.

- **Synth Signs**: the Synthetic Signs collection [83] contains $100k$ samples of common street signs obtained from Wikipedia and artificially transformed to simulate various imaging conditions.

- **GTSRB**: the German Traffic Signs Recognition Benchmark (GTSRB [107]) consists of $51,839$ cropped images of German traffic signs.

Figure 2.8: Samples from the signs datasets. From left to right: Synth Signs and GTSRB

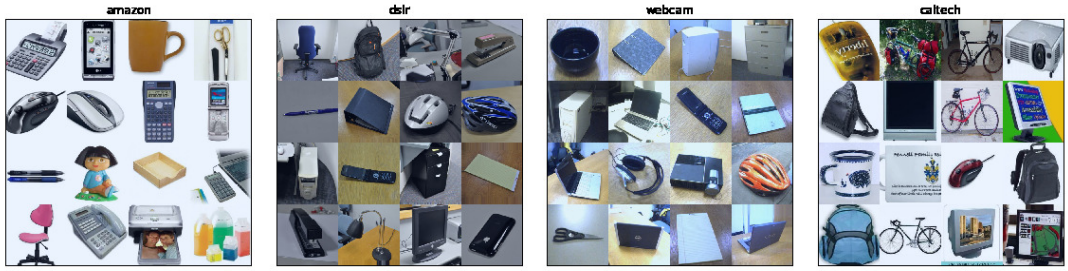


Figure 2.9: Samples from the Office and Caltech settings. From left to right: Amazon, DSLR, Webcam and Caltech

Real life datasets

These datasets are more complex and unconstrained; they better mimic real world applications. Most of them are commonly used for evaluation of recent deep learning based methods.

- The **Office 31**[94] dataset is one of the main benchmark datasets for testing domain-adaptation methods. It contains 4652 images grouped into 31 classes from three different domains: Amazon (A), DSRL (D) and Webcam (W). Amazon images are collected from `amazon.com`, Webcam and DSLR images were manually collected in an office environment. As can be seen in Figure 2.9, the domain gap is pretty large between Amazon and the remaining two, and small between DSLR and Webcam.

- **Office-Caltech** [43] dataset is obtained by selecting the subset of 10 common categories in the Office31 and the Caltech256 datasets. It contains 2533 images of which about half belong to Caltech256. Each of Amazon (A), DSLR (D), Webcam

Figure 2.10: Samples from VisDA dataset. First two rows: Synthetic images. Last two rows: Real Images.

(W) and Caltech256 (C) are regarded as separate domains.

- **VisDA** [87] is the large scale domain adaptation dataset containing two datasets: synthetic dataset and real dataset. The synthetic dataset containing 152,397 images was generated by rendering 3D models of the same object categories as in the real data (55,388 images) from different angles and under different lighting conditions.

- **Multi-PIE** dataset [52] includes face images of 337 individuals captured from different expressions, views, and illumination conditions categorized by the face expressions (**normal, smile, surprise, squint, disgust, scream**) as labels. The images from each view contains 27120 images of size $64 \times 64 \times 3$.

- **PACS** [68] is a recently proposed benchmark which is especially interesting due to the significant domain shift between different domains. It contains 9991 images in total across 7 categories ("dog", "elephant", "giraffe", "guitar", "house", "horse" and "person") and four domains of different stylistic depictions ("**Photo**", "**Art painting**", "**Cartoon**" and "**Sketch**"). The diverse depiction styles provide a significant domain gap. The goal is to train in a set of domains and recognize objects in a very different domain.

Figure 2.11: Samples from Multi-PIE dataset.

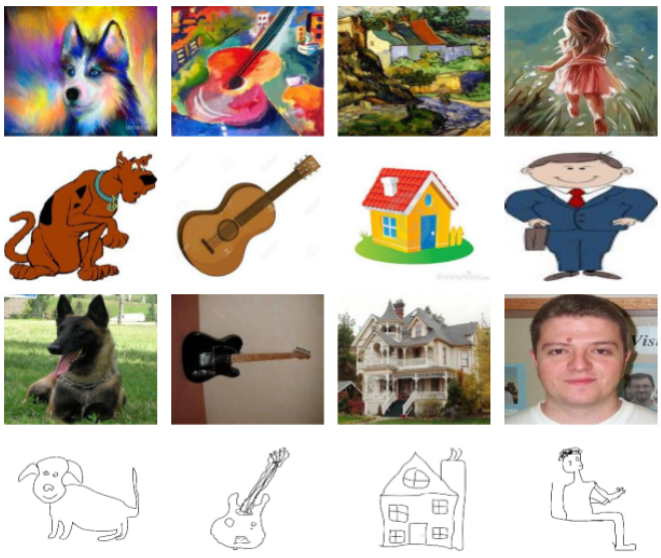


Figure 2.12: Samples from PACS dataset.

Chapter 3

Learning to see across domains

This chapter presents different strategies that allow us to deal with the Unsupervised Domain Adaptation problem. Four methods are presented: ***Punda****,* ***MCDA****,****TDDA****,* ***MTDA-ITA****. The first uses probabilistic latent variable model to learn projections from each source and target domain to a latent (shared) space jointly with the classifier in the latent space, which simultaneously minimizes the domain disparity while maximizing the classifier's discriminative power. The second proposes a more systematic and effective way to match the distribution of the domains using Gaussian processes (****GP****). The* ***GP*** *induces a hypothesis space of classifiers from the posterior distribution of the latent random functions, turning the learning into a large-margin posterior separation problem. The third improve the performance of adversarial* ***DA*** *by introducing a discriminative discrepancy measure which takes advantage of auxiliary information available in the source and the target domains to better align the source and target distributions. The last method proposes an information theoretic approach for domain adaptation in the novel context of multiple target domains with unlabeled instances and one source domain with labeled instances. The model aims to find a shared latent space common to all domains, while simultaneously accounting for the remaining private, domain-specific factors.*

3.1 PUnDA: Probabilistic Unsupervised Domain Adaptation for Knowledge Transfer Across Visual Categories

Traditional machine learning algorithms assume that the training and test data are independent and identically distributed (i.i.d.), coming from the same underlying distribution [109]. However, in real-world data, this assumption rarely holds due to a number of artifacts, such as different types of noise, changes in object view, etc. This inevitably introduces different types of biases in the observed data sampled during the training and test stage, causing the training and test data to change over time and space. Consequently, the assumptions made by traditional learning algorithms are usually violated, resulting in (significant) degradation of their performance during the inference of test data.

In most existing unsupervised DA methods, the first step is to project the source and target data onto a common space such that the source data is as close as possible to the target data in their distribution [46, 42, 6, 34, 78, 60, 82, 38]. Then, a classifier trained on the transformed source domain is applied to the target data, hoping that it will perform equally well across the domains as the domain mismatch is minimized through the learned projections.

However, most of these methods suffer from at least one or more of the following limitations that can adversely affect their performance and/or constrain their applicability to the target tasks. First, majority of existing methods are deterministic [46, 42, 6, 34, 78], relying on costly cross-validation procedures to find the size of the underlying manifold in which the mismatch between the source and target domains can effectively be reduced — increasing the computational complexity of the model and making it more prone to overfitting. Second, the minimization of the domain mismatch and learning of the target classifiers are done independently resulting in the joint feature space that is suboptimal for the main task, i.e., classification.

To overcome the above-mentioned limitations, in this section, we introduce a novel probabilistic framework that we call Probabilistic Unsupervised Domain Adaptation (**PUnDA**). In contrast to existing two-stage approaches where new feature spaces

and classifiers are separately learned, our approach learns both the classifier and low dimensional subspace jointly via a newly introduced Bayesian learning framework. Moreover, the probabilistic nature of our **PUnDA** allows it to automatically infer the dimensionality of the common subspace.

Since these benefits come with computational challenges if not addressed properly, we introduce an efficient learning and inference method based on the variational Bayes (VB) framework. Within this framework, we also propose an extension of the Maximum Mean Discrepancy (MMD) score [13], traditionally used to measure the domain mismatch, with the aim to align the source and target domains via estimated (variational) posteriors - thus, exploiting the model uncertainty — something the deterministic approaches fail to account for. Finally, the proposed VB learning in our **PUnDA** allows us to effectively incorporate unlabeled data of the target domain into the classifier learning via the regularizers specifically designed to minimize the expected classification loss in target domain. Our method is expected to bring most benefits in the DA cases when: (i) the data in both the source and target domains are tightly clustered, and (ii) the clusters from the two domains are geometrically close to each other. We show in our experiments on several benchmark datasets that the proposed approach significantly outperforms the state-of-the-art methods for unsupervised DA as they fail to account for the properties exploited in our **PUnDA** approach.

3.1.1 Unsupervised Domain Adaptation using Probabilistic Latent Variable Models

In this section, we present **PUnDA** for unsupervised DA. We consider a multi-class classification problem as the running example. Specifically, suppose we are given source-domain training examples $X^s = [x_1^s, ..., x_N^s] \in \mathbb{R}^{d \times N_s}$, with labels $Y^s = [y_1^s, ..., y_{N_s}^s] \in \mathbb{R}^{1 \times N_s}$, $y \in \{1, 2, ..., C\}$ (we assume the shared set of class labels between the two domains), and target data $X^t = [x_1^t, ..., x_{N_t}^t] \in \mathbb{R}^{d \times N_t}$. Our goal is to assign the correct class label Y^t to target data points X^t. Figure 3.1 shows the model's representation as a Bayesian network. There are three observed variables represented by the shaded nodes: the source features X^s, the target features X^t, and the source labels Y^s. Note

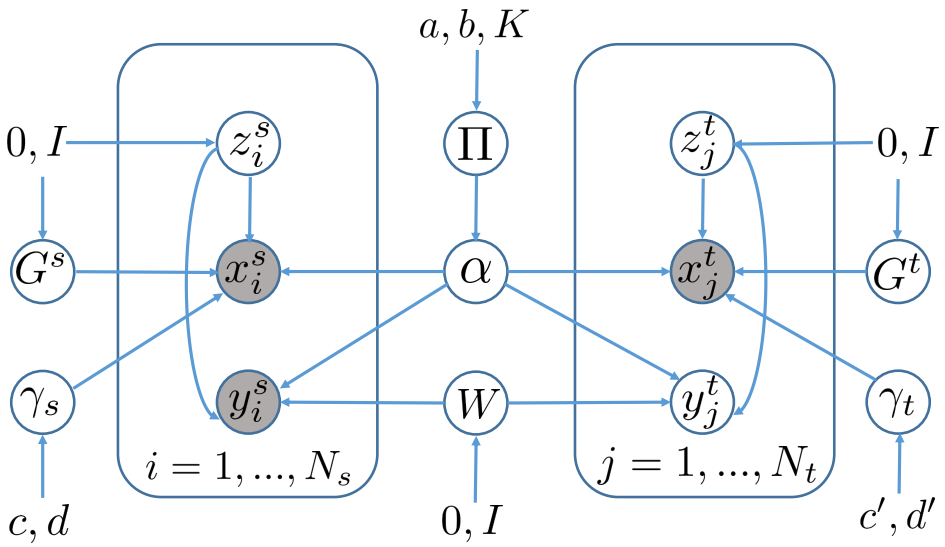


Figure 3.1: The graphical representation of **PUnDA** (the shaded circles denote the observed data). $\{x_i^s, x_j^t\}$ are the source and the target variables in the observation space, $\{y_i^s, y_j^t\}$ are the labels of the source and the target data, and $\{z_i^s, z_j^t\}$ are the representation of two domains in the shared space. G^s/G^t are the source/target projection matrices. The elements of $\boldsymbol{W}$ are the classifier parameters that are shared between both the source and target domains. α defines the underlying dimension of the shared space, and γ_s and γ_t are the noise parameters of the source and target domain, respectively.

that we assume that we do not have access to target labels, hence, Y^t are unobserved. By assuming the existence of a low-dimensional latent space where the source and target distributions are similar, we model each feature x_i^s/x_i^t, as a linear transformation G^s/G^t of their latent representations z_i^s/z_i^t in the source/target domain, corrupted with an additive Gaussian noise ϵ^s/ϵ^t, as

$$x_i^s = G^{s\top} z_i^s + \epsilon^s, \quad x_i^t = G^{t\top} z_i^t + \epsilon^t, \tag{3.1}$$

where $G^s = [g_1^s, ..., g_K^s] \in \mathbb{R}^{d\times K}$, and $G^t = [g_1^t, ..., g_K^t] \in \mathbb{R}^{d\times K}$ are the transformation matrices for source and target domains, respectively. $\epsilon^s \sim \mathcal{N}(0, \gamma_s^{-1}\boldsymbol{I}_d)$ and $\epsilon^t \sim \mathcal{N}(0, \gamma_t^{-1}\boldsymbol{I}_d)$ are the zero-mean Gaussian noise with precision values γ_s and γ_t, respectively ($\boldsymbol{I}_d$ denotes a $d \times d$ identity matrix). To keep the exponential family conjugacy between the prior and likelihood distributions, we place non-informative gamma hyper-priors on γ_s and γ_t, as

$$\gamma_s \sim Ga(c_1, c_2), \quad \gamma_t \sim Ga(c_1', c_2'),$$

where Ga denotes the Gamma distribution.

To automatically infer the dimensionality K of the shared latent space, we introduce an auxiliary binary vector $\alpha \in \{0,1\}^K$ for the latent features $\{z_i^s\}$, $\{z_j^t\}$, where the non-zero entries of α specify which latent features are used to represent the observations. Consequently, the model in Eq. 3.1 is reformulated as

$$x_i^s = G^{s\top}(\alpha \odot z_i^s) + \epsilon^s, \;\; x_i^t = G^{t\top}(\alpha \odot z_i^t) + \epsilon^s, \tag{3.2}$$

where $\odot$ denotes the element-wise multiplication operator. Note that all the source and target data points (x_i^s/x_j^t) share the same set of important latent features defined by α, but each have their unique weights (z_i^s/z_j^t).

Using the notion of the probabilistic hierarchical framework as in [32], we place a non-parametric prior on the binary vector α by introducing auxiliary variables $\mathbf{\Pi} = \{\pi_k\}_{k=1}^K$ drawn from the Beta distribution as

$$\pi_k \sim Beta(a/K, b(K-1)/K),$$

where a, b are the hyper-parameters and the integer K is the largest possible dimension for α (by letting $K \to \infty$, the length of the binary code α can be learned from the observed data [112]). Then, we model the binary vector α as a random sample from the Bernoulli process parameterized by Π as

$$\alpha \sim \prod_{k=1}^{K} Ber(\alpha_k; \pi_k), \quad k = 1, ..., K,$$

where α_k denotes the k-th element of the binary vector α and Ber denotes the Bernoulli distribution (we obtain the Indian Buffet Process (IBP) prior[51] on α by integrating out Π and letting $K \to \infty$). For computational simplicity, we model the latent features $Z^s = [z_1^s, ..., z_{N_s}^s] \in \mathbb{R}^{K \times N_s}$ and $Z^t = [z_1^t, ..., z_{N_t}^t] \in \mathbb{R}^{K \times N_t}$ using a multivariate zero-mean Gaussian distribution:

$$P(z_i^s) \sim \mathcal{N}(0, \boldsymbol{I}_K), \; P(z_j^t) \sim \mathcal{N}(0, \boldsymbol{I}_K).$$

Similarly, we also assume that the elements of the transformation matrices are drawn from a multivariate zero-mean Gaussian distribution:

$$P(g_i^s) \sim \mathcal{N}(0, \boldsymbol{I}_d), \; P(g_j^t) \sim \mathcal{N}(0, \boldsymbol{I}_d).$$

In order to make the latent representations discriminative for the classification task, we employ the softmax regression classifier. More precisely, for the shared space representation $\alpha \odot z$ of a sample x, the probability of the x's label y belonging to class $c = 1, \ldots, C$ is computed as:

$$P(y = c|\boldsymbol{W}, z, \alpha) = \frac{exp\big(w_c^\top (\alpha \odot z)\big)}{\sum_{c'=1}^{C} exp\big(w_{c'}^\top (\alpha \odot z)\big)},$$

where $\boldsymbol{W} = [w_1, ..., w_C] \in \mathbb{R}^{(K+1)\times C}$ contains the class projection vectors. Again, within our probabilistic framework, we assume that elements of $\boldsymbol{W}$ are drawn from a multivariate zero-mean Gaussian distribution ($w_c \sim \mathcal{N}(0, \boldsymbol{I}_{K+1})$). It is worth noting that $\boldsymbol{W}$ includes a bias by having an extra dimension $z_0 = 1$ and $\alpha_0 = 1$ for z and α, respectively.

Posterior Inference

Because computing the exact posterior distribution of the latent variables $\Omega = \{Z^s, Z^t, \boldsymbol{W}, G^s, G^t, \alpha, \boldsymbol{\Pi}, \gamma_s, \gamma_t\}$ is intractable, we derive a Variational Bayes (VB) algorithm [37] to approximate this posterior distribution in the proposed **PUnDA** approach.

The goal of the VB is to approximate the true posterior distribution over the latent variables $P(\Omega|X^s, X^t, Y^s)$ with a variational distribution $q(\Omega)$, which is closest in KL divergence to the true posterior distribution. It is easy to show that this equals to maximizing the lower bound of the marginal likelihood $P(X^s, Y^s, X^t|\boldsymbol{\Theta})$

$$q^*(\boldsymbol{\Omega}) = \arg\max_{q(\boldsymbol{\Omega})} \mathbb{E}_q\big[\log(X^s, Y^s, X^t, \boldsymbol{\Omega}|\boldsymbol{\Theta})\big] + \boldsymbol{H}[q(\boldsymbol{\Omega})],$$

where $\boldsymbol{\Theta} = \{a, b, K, c, d, c', d'\}$ denotes the set of hyper-parameters, $\mathbb{E}_q[.]$ denotes the expectation operator under the distribution q, and $\boldsymbol{H}[.]$ the entropy operator. For our framework to yield a computationally effective inference method, we employ a factorized variational distribution:

$$q(\boldsymbol{\Omega}) = \Pi_{i=1}^{N_s} q(z_i^s) \Pi_{j=1}^{N_t} q(z_j^t) \Pi_{k=1}^{K} q(g_k^s) q(g_k^t) \Pi_{c=1}^{C} q(w_c) q(\gamma_s) q(\gamma_t).$$

For simplicity, we also fix K and set it to a finite but large number. If K is large enough (see Section 3.1.2), the observed data will reveal fewer than K components for shared space features.

Apart from maximizing the marginal likelihood, we also need the shared latent features to be invariant to differences between the source and target domains, i.e., to be robust to the covariate shift that may exist in the target space. To this end, we introduce a regularizer $\mathcal{L}(Z^s, Z^t)$, based on on the Maximum Mean Discrepancy (MMD) [13], designed to minimize the distance between the distributions of the source and target representations. Using the kernel trick, we rewrite the squared MMD, leading to the following regularizer:

$$\mathcal{L}(Z^s, Z^t) = \sum_{i,j} \frac{\mathbb{K}(z_i^s, z_j^s)}{N_s^2} - 2\sum_{i,j} \frac{\mathbb{K}(z_i^s, z_j^t)}{N_s N_t} + \sum_{i,j} \frac{\mathbb{K}(z_i^t, z_j^t)}{N_t^2},$$

where $\mathbb{K}(.,.)$ denotes the kernel function. In contrast to most existing DA methods that measure the domain distance directly in the learned RKHS [85, 6, 77, 119, 76], **PUnDA** encodes this distance using the posterior distributions of the shared features Z^s and Z^t – thus, accounting also for uncertainty of the projections from the two domains. To this end, we use the Bhattacharyya kernel [22] to measure the posterior similarity as

$$\mathbb{K}(q(z^s), q(z^t)) = \log \int_{\mathbb{R}^K} q(z^s)^{1/2} q(z^t)^{1/2}\, dz^s\, dz^t.$$

The intuition behind this kernel is that it measures the amount of overlap (similarity) between two distributions $q(z^s)$ and $q(z^t)$, by integrating the square root of their product over the whole space [22].

To learn a good classifier, we also leverage knowledge of the target domain samples by minimizing the uncertainty of the classifier over the target samples. To this end, we introduce a regularizer $\mathcal{L}'(\boldsymbol{W}, Z^t, \alpha)$ designed to minimize the Shanon Entropy of the probability vectors $P(y_j^t|\boldsymbol{W}, z_j^t, \alpha)$ over the target domain samples:

$$\mathcal{L}'(\boldsymbol{W}, Z^t, \alpha) = \sum_{j=1}^{N_t} \sum_{c=1}^{C} \mathbb{E}_{P(y_j^t|\boldsymbol{W}, z_j^t, \alpha)} \log P(y_j^t = c).$$

Intuitively, if our assumptions about two sets of clusters being geometrically close indeed hold in the used datasets, the probability vector $P(y_j^t|\boldsymbol{W}, \alpha, z_j^t) = [p_j^1, ..., p_j^C]$ should ideally look like a posterior probability vector $[0, 0, ..., 1, ..., 0]$ (using 1-of-many coding). Since we do not know the true label, we cannot measure directly the similarity of $P(y_j^t|\boldsymbol{W}, \alpha, z_j^t)$ and the correct label. However, we can minimize the entropy of

$P(y_j^t|\boldsymbol{W}, \alpha, z_j^t)$ by which we can reduce the amount of information that $P(y_j^t|\boldsymbol{W}, \alpha, z_j^t)$ contains about the confusing labels.

By defining the regularizers $\mathcal{L}(Z^s, Z^t)$ and $\mathcal{L}'(Z^t, \boldsymbol{W}, \alpha)$, the proposed regularized VB algorithm can be written as the following optimization problem:

$$q^*(\boldsymbol{\Omega}) = \arg\max_{q(\boldsymbol{\Omega})} \mathbb{E}_q\big[\log(X^s, Y^s, X^t, \boldsymbol{\Omega}|\boldsymbol{\Theta})\big] + \boldsymbol{H}[q(\boldsymbol{\Omega})] \\ - \lambda\mathcal{L}(Z^s, Z^t) + \lambda'\mathcal{L}'(\boldsymbol{Z}^t, \boldsymbol{W}, \alpha),$$

where $\lambda \geq 0$ and $\lambda' \geq 0$ denote the regularization parameters. The VB algorithm solves the above optimization problem using the Coordinate Descent algorithm. The computational complexity of each iteration of the proposed VB algorithm, for training, in one iteration is $O((N_s+N_t)dK^2)$, i.e., linear in the size of the source+target data N_s+N_t, the data dimensionality d, and quadratic in the dimensionality of the shared space $K(K << d)$. Details of the proposed VB algorithm and its computational complexity analysis, along with other derivations, are available in the Appendix.

Target Class Label Prediction

After computing the posterior distribution $q^*(\boldsymbol{\Omega})$, to determine the target class-label y_j^t of a given target domain instance x_j^t, we first compute the distribution of y_j^t given x_j^t by integrating out the latent variables $\{\boldsymbol{W}, \alpha, z_j^t\}$. Then, we select the most likely label as

$$\hat{y}_j^t = \arg\max_{y_j^t \in \{1,\ldots,C\}} P(y_j^t|x_j^t),$$

where $P(y_j^t|x_j^t)$ can be computed as

$$P(y_j^t = c|x_j^t) = \sum_{\alpha} \int P(y_j^t|\boldsymbol{W}, \alpha, z_j^t) q^*(\alpha) q^*(z_j^t) q^*(\boldsymbol{W})\ dz_j^t\, d\boldsymbol{W}.$$

Since the above expression cannot be computed in a closed form, we approximate $q^*(\alpha), q^*(z_j^t)$, and $q^*(\boldsymbol{W})$ with their mean values: $\mathbb{E}_{q^*(\alpha)}[\alpha], \mathbb{E}_{q^*(\boldsymbol{W})}[\boldsymbol{W}]$ and $\mathbb{E}_{q^*(z_j^t)}[z_j^t]$, respectively. Using this approximation, we compute z_j^t as:

$$\hat{y}_j^t = \arg\max_{c \in \{1,\ldots,C\}} \mathbb{E}_{q^*(w_c)}[w_c]^\top (\mathbb{E}_{q^*(\alpha)}[\alpha] \odot \mathbb{E}_{q^*(z_j^t)}[z_j^t]).$$

3.1.2 Experimental Results

In this section, we show in our experiments on several benchmark datasets that the proposed approach significantly outperforms the state-of-the-art methods for unsupervised DA as they fail to account for the properties exploited in our **PUnDA** approach.

We run extensive experiments on unsupervised DA tasks, where we use the hand-crafted features (SURF) [9] and the current state-of-the-art deep-net features (VGG-Net) [104]. We compare our approach to several state-of-the-art unsupervised DA methods on two DA benchmark datasets: the Office+Caltech10 and Multi-PIE datasets.

Office+Caltech10 dataset contains images collected from four different sources and 10 object classes. The corresponding domains are **A**mazon, **W**ebcam, **D**SLR, and **C**altech. The Multi-PIE dataset used in this experiment, includes face images of 67 individuals captured from different expressions, views, and illumination conditions. We compare the performance of the proposed **PUnDA** approach to the following benchmarks:

- **1-NN** and **SVM**: original features are used without any adaptation, a basic 1-nearest neighbor (1-NNs) and linear SVM is found by comparing the target samples to the training data from the source domain.
- **GFK** [43]: The geodesic flow kernel algorithm. Results are evaluated using the kernel-NNs.
- **SA** [33]: The subspace alignment algorithm. Results are evaluated using 1-NN.
- **CORAL** [109]: The correlation alignment algorithm that uses a linear SVM on the similarity matrix formed by the correlation matching.
- **ILS** [56]: Invariant Latent Space algorithm. Results are evaluated using 1-NN.

For the VB algorithm, we set the truncation level for the dimensionality of the latent space to ($K = 100$) for both datasets. The hyper-parameters a, b of the Beta distributions are set with $a = 1$ and $b = 1$ (other settings of a and b yield similar results). All Gamma priors are set as $Ga(10^{-6}, 10^{-6})$ to make the prior distributions uninformative. In all

Table 3.1: Unsupervised domain adaptation results using VGG-FC6 features on Office+Caltech10 dataset with the evaluation setup of [56].

method	A → W	A → D	A → C	W → A	W → D	W → C	D → A	D → W	D → C	C → A	C → W	C → D	Ave.
1-NN	60.9	52.3	70.1	66.4	91.3	60.2	57.0	86.7	48.0	81.9	65.9	55.6	66.4
SVM	63.1	51.7	74.2	73.3	94.2	68.2	58.7	91.8	55.5	86.7	74.8	61.5	71.1
GFK[43]	74.1	63.5	77.7	81.1	**96.6**	73.5	69.9	92.4	64.0	86.2	76.5	66.5	76.8
SA[33]	76.0	64.9	77.1	80.2	94.2	71.9	69.0	90.5	62.3	83.9	76.0	66.2	76.0
CORAL[109]	74.8	67.1	79.0	82.3	96.0	75.9	75.8	**94.6**	64.7	89.4	77.6	67.6	78.7
ILS[56]	82.4	72.5	78.9	**87.2**	89.3	79.9	79.2	94.2	66.5	87.6	84.4	73.0	81.3
PUnDA	**82.7**	**76.2**	**82.3**	86.9	89.8	**82.6**	**83.1**	93.4	**69.2**	**90.3**	**88.3**	**76.2**	**83.4**

Table 3.2: Unsupervised domain adaptation results using VGG-FC7 features on Office+Caltech10 dataset with the evaluation setup of [56].

method	A → W	A → D	A → C	W → A	W → D	W → C	D → A	D → W	D → C	C → A	C → W	C → D	Ave.
1-NN	64.0	50.8	72.6	67.8	88.8	64.2	61.2	88.2	52.8	82.6	65.3	54.9	67.8
SVM	68.0	51.8	76.2	74.6	93.0	70.6	58.7	91.2	56.0	86.7	74.8	61.3	71.9
GFK[43]	74.0	57.6	76.6	76.0	92.9	69.5	67.5	91.9	62.9	84.1	73.6	63.4	74.2
SA[33]	75.0	60.7	76.2	76.4	94.0	69.0	66.0	89.5	59.4	82.6	73.6	63.2	73.8
CORAL[109]	71.8	61.3	78.6	82.0	**94.6**	73.7	71.2	**93.5**	63.0	88.6	76.0	63.8	76.5
ILS[56]	80.9	71.3	78.4	**86.7**	88.2	76.3	76.5	91.8	66.2	87.1	80.1	67.1	79.2
PUnDA	**81.4**	**75.8**	**81.0**	85.7	90.1	**80.1**	**80.4**	92.0	**69.1**	**91.1**	**83.8**	**70.8**	**81.7**

our experiments, we set $\lambda = 0.1$ and $\lambda' = 1$. We use the classification accuracy for the target data as the evaluation metric:

$$Accuracy = \frac{\left|\{x^t : x^t \in X^t, f(x^t) = y^t\}\right|}{\left|\{x^t : x^t \in X^t\}\right|},$$

where $f(x^t)$ shows the predicted label by the methods for the target data point x^t, and y^t is the actual label of x^t.

Results for OFFICE+CALTECH10

For this dataset, we used 4096 dimensional VGG-fc6 and VGG-fc7 features extracted with the network model of [104] for the deep-net feature experiments. Following the experimental protocol in [56], we also use SURF features [9] (each image is encoded with an 800-bin histogram and the histograms are then normalized to have zero mean and unit standard deviation in each dimension) as hand-crafted features. We set the latent space dimensionality to 20 for VGG features and to 100 for SURF features in all compared methods, as these were empirically found to be the best for the competing methods [56]. For each pair of the source and target domains, we conduct experiments using 20 random train/test splits.

In Tables 3.1 and 3.3, we report the performance using VGG-FC6, VGG-FC7

Table 3.3: Unsupervised domain adaptation results using SURF features on the Office+Caltech10 dataset with the evaluation setup from [56].

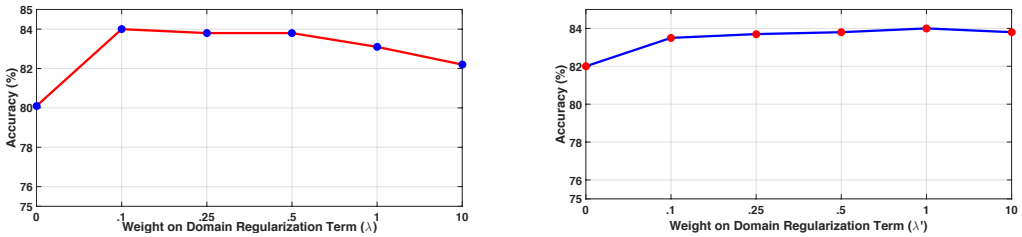
method	A → W	A → D	A → C	W →A	W → D	W → C	D → A	D → W	D → C	C → A	C → W	C → D	Ave.
1-NN	23.1	22.3	20.0	13.8	40.6	12.2	23.0	51.7	19.9	21.0	19.0	23.6	24.2
SVM	25.6	33.4	35.9	32.1	78.9	25.2	34.6	70.2	31.2	43.8	30.5	40.3	40.1
GFK[43]	35.7	35.1	37.9	35.5	71.2	29.3	36.2	79.1	32.7	40.4	35.8	41.1	42.5
SA[33]	38.6	37.6	35.3	37.4	80.3	32.3	38.0	83.6	32.4	39.0	36.8	39.6	44.2
CORAL[109]	38.7	38.3	**40.3**	37.8	84.9	34.6	38.1	**85.9**	34.2	47.2	39.2	40.7	46.7
ILS[56]	40.6	**41.0**	37.1	39.0	78.7	34.2	38.9	79.1	36.9	48.6	**42.0**	44.1	46.7
PUnDA	**42.5**	40.3	39.5	**42.4**	**85.2**	**36.5**	**40.3**	83.2	**38.9**	**50.1**	41.7	**45.8**	**48.8**

and SURF features, respectively. As can be seen, for all the feature types, **PUnDA** outperforms the state-of-the-art methods in most of domain transformations, and, generally, provides the highest overall classification accuracies for all the feature types. We also note that the VGG-fc7 is less favorable than VGG-fc6 for majority of the DA algorithms compared.

The higher performance of **PUnDA** compared to other methods is mainly attributed to the joint learning of the discriminative classifier and low-dimensional feature spaces. The key observation is that good representations are beneficial to data classification, with classification results providing supervisory signals to representation learning. Furthermore, from the results obtained, it is obvious that it is more beneficial to make the use of information coming from unlabeled target data during classifier learning process compared to when no data from target domain is used. Indeed, using the proposed learning scheme, we find a representation space in which we embed the knowledge from the target domain into the learned classifier.

Sensitivity Analysis

In the experiments above, we keep $\lambda = 0.1, \lambda' = 1$. To analyze the sensitivity of our method to changes in parameters λ and λ', we conducted additional experiments to analyze the parameter sensitivity of **PUnDA** w.r.t. the various values of λ and λ'. To this end, we consider random splits from each of the Office+Caltech10 dataset along VGG-FC6 features here. Figure 3.2 shows the sensitivity analysis for the parameters of **PUnDA** on these random splits. Sensitivity analysis is performed by varying one parameter at the time over a given range, while for the other parameters we set them to their final values ($\lambda = 0.1, \lambda' = 1$). From Figure 3.2 (a), we see that when $\lambda = 0$ (no



(a) Sensitivity to λ (b) Sensitivity to λ'

Figure 3.2: Sensitivity analysis of **PUnDA**.

domain mismatch regularization term is considered), the performance drops considerably. For other values of λ, the performance is superior and there is little variation in the model performance, evidencing the robustness of **PUnDA** w.r.t. λ. Similarly, from Figure 3.2 (b), **PUnDA** is largely insensitive to the parameter λ' over the specified range of its values. Moreover, it is clear that using the unlabeled target data improves the discriminative power of the classifier.

Results on Multi-PIE Faces

In this experiment, we follow the setting in [56] and use the views: $C27$ (looking forward) and $C09$ (looking down), as the source domain, and the views: $C05, C37, C02, C25$ (looking towards left in an increasing angle, see Figure 3.12), as target domains. We expect the face inclination angle to reflect the complexity of transfer learning. We normalize the images to 32×32 pixels and use the vectorized gray-scale images as features. The dimensionality of the common feature space for all the feature learning-based methods is set to 100.

Table 3.4 shows the classification accuracy w.r.t. the increasing angle of inclination. As can be seen, **PUnDA** achieves the best performance (on average) as well as the best scores for the 3 views and the second best for the $C02$. Clearly, with the increasing camera angle, the feature structure changes up to a certain extent (the features become heterogeneous). However, our method produces good accuracies even under such challenging conditions.

Table 3.4: Multi-PIE results. The changes in performance w.r.t. the changing face orientations when frontal face images ($C27$) are considered as the source domain.

method	C09	C05	C37	C25	C02	Ave.
1-NN	92.5	55.7	28.5	14.8	11.0	40.5
SVM	87.8	65.0	35.8	15.7	16.7	44.2
GFK[43]	92.5	74.0	32.1	14.1	12.3	45.0
SA[33]	**97.9**	85.9	47.9	16.6	13.9	52.4
CORAL[109]	91.4	74.8	35.3	13.4	13.2	45.6
ILS[56]	96.6	88.3	72.9	28.4	**34.8**	64.2
PUnDA	94.3	**92.2**	**78.8**	**28.9**	34.7	**65.7**

Model Selection

To demonstrate the ability of the proposed method to learn the dimensionality of the latent space automatically, we conduct experiments on both Office+Caltech10 and Multi-PIE datasets. We consider a random split from A $\rightarrow$ W of the Office+Caltech10 dataset along VGG-FC6 features, and $C27 \rightarrow C25$ from the Multi-PIE dataset.

We plot the sorted values of $\mathbb{E}[q^*(\alpha)]$ for the selected source/target datasets, inferred by the algorithm in Figure 3.4. As can be seen, the **PUnDA** inferred approximately $25 - 30$ dimensions for the learned latent space for the selected domain transformations of Office+Caltech10, and $80 - 85$ dimensions for the learned latent space for domain transformations in the Multi-PIE dataset, fewer than 100, as initially provided. It is worth noting that since the number of data points in the $C27, C25$ datasets is much larger than the number of samples in the A, W dataset, we need more latent dimensions for $C27, C25$ than for A, W to capture the variations in these datasets.

Remark

In the experiments conducted, we showed that our approach is able to achieve better performance than the competing methods. Namely, as stated before, our method is expected to bring most benefits in the DA cases when data in both domains are tightly clustered, with the clusters being geometrically proximal. Indeed, Figure 3.3 depicts the embedding of the learned features z^s/z^t, and those of **ILS** and the original features x. Colors indicate source (red) and target (blue) domains. Notice that **PUnDA** significantly reduces the domain mismatch, resulting in the expected tight clustering.

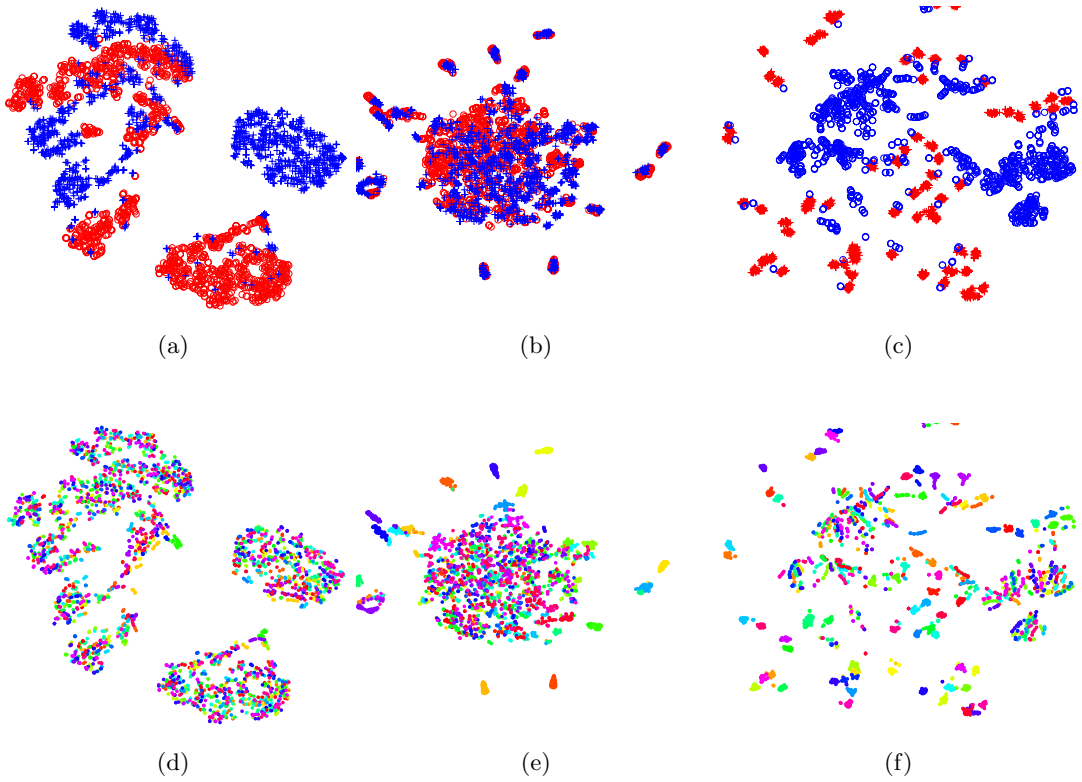
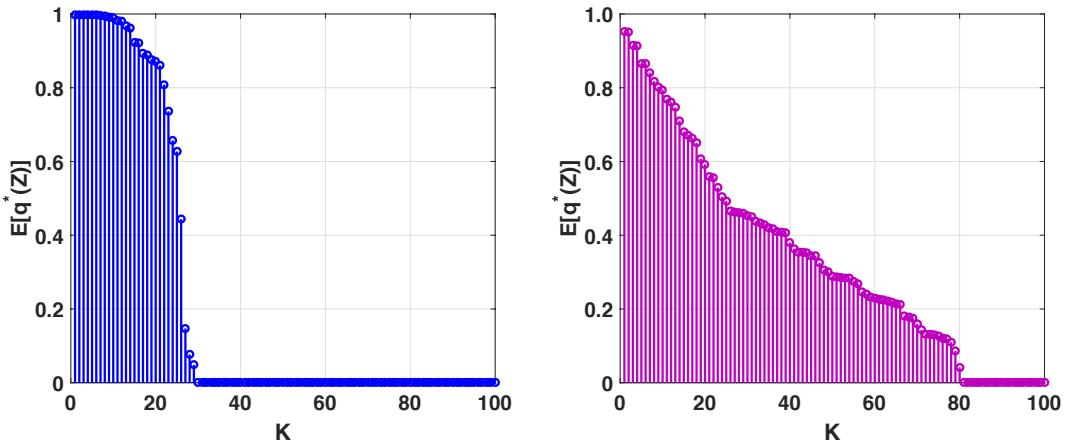


Figure 3.3: Feature visualization. The embedding of Multi-PIE $C05$ data using t-sne algorithm [80]. (a) Original features. (b) **PUnDA** features. (c) **ILS** features.

This is partially due to the use of the proposed probabilistic MMD with Bhattacharyya kernel, which penalizes the domain mismatch while exploiting the uncertainty in the shared feature space - something the **ILS** fails to account for.

3.1.3 Summary

In this section, we proposed a novel *probabilistic* approach for unsupervised DA that learns an efficient domain-adaptive classifier that can generalize well on target domains. The key to the proposed approach is that it jointly learns a latent space along with its size, and a softmax classifier, by exploiting both labeled source and unlabeled target data in Bayesian fashion. To tackle the intractability of computing the exact posteriors in our model, we proposed a novel Bayesian approximation to efficiently approximate the target distributions. We showed on two benchmark datasets for image classification,



(a) $A \rightarrow W$ (b) $C27 \rightarrow C25$

Figure 3.4: Inferred $\mathbb{E}[q^*(\alpha)]$ for the Office+Caltech10 and Multi-PIE datasets.

using both hand-crafted and deep-net features, the superiority of the proposed method compared to the state-of-the-art methods for unsupervised domain adaptation of visual domain categories.

3.2 GPDA: A Deep Max-Margin Gaussian Process Approach for Unsupervised Domain Adaptation

In previous section, we proposed an unsupervised DA method in which, the target domain error can be provably reduced by having a shared input representation that makes the source and target domains indistinguishable from each other.

The idea is to find a latent space shared by both domains such that the classifier learned on it using the fully labeled data from the source will also perform well on the target domain. This is accomplished, and supported in theory [11], by enforcing a requirement that the distributions of latent points in the two domains be indistinguishable from each other. A large family of **UDA** approaches including [46, 42, 1, 6, 34, 78, 2, 60, 82, 38] leverage this idea. However, their performance remains unsatisfactory, in part because the methods inherently rely on matching of marginal, class-free, distributions while using the underlying assumption that the shift in the two distributions, termed covariate shift [108], can be reduced without using the target domain labels. The main reason is that it merely matches the marginal input distributions of source and target, ignoring the alignment of class distributions.

To address this issue, as effective solution was proposed in [97], which aims to take into account the class-specific decision boundary. Its motivation follows the theorem in [10] relating the target domain error to the maximal disagreement between any two classifiers, tighter than the former bound in [11]. It implies that a provably small target error is achievable by minimizing the maximum classifier discrepancy (**MCD**). The approach in [97], the **MCD** Algorithm (**MCDA** for short), attempted to minimize **MCD** directly using adversarial learning similar to GAN training [45], i.e., through solving a minimax problem that finds the pair of most discrepant classifiers and reduces their disagreement.

In this section, we further extend the **MCD** principle by proposing a more systematic and effective way to achieve consistency in the hypothesis space of classifiers $\mathcal{H}$ through Gaussian process (**GP**) [92] endowed priors, with deep neural networks (DNNs) used to induce their mean and covariance functions. The crux of our approach is to regard

the classifiers as random functions and use their posterior distribution conditioned on the source samples, as the prior on $\mathcal{H}$. The key consequence and advantages of this Bayesian treatment are: (1) One can effectively minimize the inconsistency in $\mathcal{H}$ over the target domain by regularizing the source-induced prior using a max-margin learning principle [122], a significantly easier-to-solve[1] task than the minimax optimization of [97] which may suffer from the difficulty of attaining an equilibrium point coupled with the need for proper initialization. (2) We can quantify the measure of prediction uncertainty and use it to credibly gauge the quality of prediction at test time.

Although **GP** models were previously known to suffer from the scalability issues [92], we utilize recent deep kernel techniques [58, 123] to turn the non-parametric Bayesian inference into a more tractable parametric one, leading to a learning algorithm computationally as scalable and efficient as conventional (non-Bayesian) deep models.

3.2.1 Unsupervised Max-Margin Domain Adaptation using Gaussian Process

Given source-domain training examples with labels $\mathcal{D}_S = \{(\mathbf{x}_i^S, y_i^S)\}_{i=1}^{N_S}$ and target data $\mathcal{D}_T = \{\mathbf{x}_i^T\}_{i=1}^{N_T}$ with no labels, we seek to learn the embedding function $G : \mathcal{X} \to \mathcal{Z}$ and a classifier $h : \mathcal{Z} \to \mathcal{Y}$ in the *shared latent space* $\mathcal{Z}$. The embedding function $G(\cdot)$ and the classifier $h(\cdot)$ are shared across both domains and will be applied to classify samples in the target domain using the composition $y = h(\mathbf{z}) = h(G(\mathbf{x}))$.

Our goal is to find the pair (h, G) resulting in the lowest generalization error on the target domain,

$$(h^*, G^*) = \arg\min_{h,G} e_T(h, G) = \arg\min_{h,G} \mathbb{E}_{(\mathbf{x},y)\sim p_T(\mathbf{x},y)}[I(h(G(\mathbf{x})) \neq y)], \tag{3.3}$$

with $I(\cdot)$ the 1/0 indicator function. Optimizing e_T directly is typically infeasible. Instead, one can exploit the upper bounds proposed in [10] and [11], which we restate, without loss of generality, for the case of fixed G.

[1] In the sense of optimization stability: it is well known that a good equilibrium point of the minimax optimization (adversarial learning), adopted in MCDA, is difficult to attain computationally, being highly sensitive to the choice of optimization hyperparameters.

Theorem 1. *[10, 11] Suppose that $\mathcal{H}$ is symmetric (i.e., $h \in \mathcal{H}$ implies $-h \in \mathcal{H}$). For any $h \in \mathcal{H}$, the following holds*[2]*:*

$$e_T(h) \leq e_S(h) + \sup_{h,h' \in \mathcal{H}} \left| d_S(h,h') - d_T(h,h') \right| + e^* \tag{3.4}$$

$$\leq e_S(h) + \sup_{h \in \mathcal{H}} \left| d_S(h,+1) - d_T(h,+1) \right| + e^* \tag{3.5}$$

Here $e_S(h)$ is the error rate of $h(\cdot)$ on the source domain, $e^* := \min_{h \in \mathcal{H}} e_S(h) + e_T(h)$, and $d_S(h,h') := \mathbb{E}_{z \sim S}[\mathbb{I}(h(z) \neq h'(z))]$ denotes the discrepancy between two classifiers h and h' on the source domain S, and similarly for $d_T(h,h')$. We use $z \sim S$ to denote the distribution of z in the latent space induced by G and $p_S(x,y)$.

The presence of $e_S(h)$ indicates that we need to choose a classifier $h \in \mathcal{H}$ that performs well on the source domain. Unfortunately, we may not be able to control e^* directly since it contains $e_T(h)$ that cannot be estimated from the training data. We can just hope that our hypothesis space $\mathcal{H}$ contains an optimal h^* that performs well on both domains so that e^* becomes small.

Looser bound. With e^* the uncontrollable quantity, due to the lack of labels for T in the training data, the optimal h can be sought through minimization of the source error $e_S(h)$ and the worst-case discrepancy terms. In the looser bound in Eq. 3.5, the supremum term is, up to a constant, equivalent to $\sup_{h \in \mathcal{H}} \mathbb{E}_{\mathbf{z} \sim S}[I(h(z) = +1)] + \mathbb{E}_{z \sim T}[I(h(z) = -1)]$, the maximal accuracy of a domain discriminator (labeling S as $+1$ and T as -1). Hence, to reduce the upper bound one needs to choose the embedding G where the source and the target inputs are indistinguishable from each other in $\mathcal{Z}$. This input density matching was exploited in many previous approaches [118, 36, 15, 117], and typically accomplished through adversarial learning [45] or the maximum mean discrepancy [50].

Tighter bound. Recently, [97] exploited the tighter bound in Eq. 3.4 under the assumption that $\mathcal{H}$ is restricted to classifiers with small errors on S. Consequently, $d_S(h,h')$ becomes negligible as any two $h, h' \in \mathcal{H}$ agree on the source domain. The supremum in Eq.3.4, interpreted as the *Maximum Classifier Discrepancy (**MCD**)*,

[2] *Note that the theorems assume binary classification ($y \in \{+1,-1\}$), however, they can be straightforwardly extended to multi-class setups.*

reduces to:

$$\sup_{h,h' \in \mathcal{H}} \mathbb{E}_{(x,y)\sim p_T(x,y)}[\mathbb{I}(h(z) \neq h'(z))]. \tag{3.6}$$

Named **MCDA**, [97] aims to minimize Eq. 3.6 directly via adversarial-cooperative learning of two deep classifier networks $h(z)$ and $h'(z)$. For the source domain data, these two classifiers and G aim to minimize the classification errors cooperatively. An adversarial game is played in the target domain: h and h' aim to be maximally discrepant, whereas G seeks to minimize the discrepancy.

In this section, we propose to adopt the **MCD** principle, in a systematic and effective way to achieve hypothesis consistency, instead of the difficult minimax optimization. Our idea is to adopt a Bayesian framework to induce the hypothesis space. Specifically, we build a Gaussian process classifier model [92] on top of the share space. The **GP** posterior inferred from the source data naturally defines our hypothesis space $\mathcal{H}$. We then optimize the embedding G and the kernel of the **GP** so the posterior hypothesis distribution leads to consistent, least discrepant, class predictions most of the time, resulting in reduction of (3.6). Our approach is denoted by **GPDA**, and its details are described below.

Gaussian Process

A Gaussian Process (GP) is an infinite collection of random variables $\{f(\mathbf{x})|\mathbf{x} \in X\}$, such that any finite number of samples have a joint Gaussian distribution. A GP is fully specified by the mean function $\mu(\mathbf{x})$ and the covariance function $k(\mathbf{x}, \mathbf{x}')$, typically user-defined. GPs can also be interpreted as a distribution over functions $f(\mathbf{x}) \sim \mathcal{GP}(\mu(x), k(x, x))$ such that any finite collection of function values $[f(\mathbf{x_1}), \ldots, f(\mathbf{x_N})]$ have a joint Gaussian distribution:

$$[f(\mathbf{x_1}), \ldots, f(\mathbf{x_N})] \sim \mathcal{N}(\boldsymbol{\mu}, \mathbb{K}), \tag{3.7}$$

where $\boldsymbol{\mu}$ is the $N \times 1$ vector $\mu_i = \mu(\mathbf{x_i})$ and $\mathbb{K}$ is the $N \times N$ covariance (Kernel) matrix with $\mathbb{K}_{ij} = \mathbb{K}(\mathbf{x_i}, \mathbf{x_j})$. A training dataset consists of N pairs of data $(\mathbf{x_i}, y_i)_{i=1}^N$, where y_i are noisy observations of some latent function f with Gaussian noise $y_i = f(\mathbf{x}_i) + \epsilon_i$, $\epsilon_i \in \mathcal{N}(0, \sigma^2)$. The likelihood of the data $\mathbf{y}|\mathbf{f} \sim \mathcal{N}(f, \sigma^2 I)$ and the prior $\mathbf{f} \sim \mathcal{N}(0, \mathbb{K})$

give the joint probability model $p(\mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f})$, where $\mathbf{y}$ denotes the noisy targets and $\mathbf{f}$ denotes the vector of underlying latent function values. The predictive distribution at a set of test points X_* is given in closed form using the properties of conditional Gaussians,

$$\mathbf{f}_*|\mathbf{y}, X, X_*, \boldsymbol{\theta}, \sigma^2 \sim \mathcal{N}(\mathbf{f}_*, cov(\mathbf{f}_*)) \tag{3.8}$$

$$\mathbf{f}_* = \mathbb{K}_*(\mathbb{K} + \sigma^2 I)^{-1}\mathbf{y} \tag{3.9}$$

$$cov(\mathbf{f}_*) = \mathbb{K}_{**} - \mathbb{K}_*(\mathbb{K} + \sigma^2 I)^{-1}\mathbb{K}_*^T, \tag{3.10}$$

where $\mathbb{K}_{**}$ denotes the covariance matrix evaluated among the test inputs X_* and $\mathbb{K}_*$ denotes the covariance matrix evaluated between the test points X_* and the training set X. If there are N_* test points, the covariance matrix $\mathbb{K}_{**}$ is of size $N_* \times N_*$ and $\mathbb{K}_*$ is of size $N_* \times N$.

Gaussian Process Classification

In Gaussian Process Classification (GPC), the target values are discrete class labels, hence it is not appropriate to model them via a multivariate Gaussian density. Instead, we use the Gaussian process as a latent function whose sign determines the class label for binary classification; for multi-class classification one can use multiple GPs or a multivariate GP.

The key difference between the GP regression and GPC is how the output data, $\mathbf{y}$, are connected to the underlying function values, $\mathbf{f}$. Precisely, they are no longer connected via a simple noise process as in the previous section, instead now discrete: for example, for binary classification framework, say $y = 1$ for one class and $y = -1$ for the other. In this case, one could try fitting a GP that produces an output of 1 for some values of x and -1 for others, simulating the discrete nature of the problem. Then, the classification of a new data point x_* involves two steps:

1. Evaluate a 'latent function' f which models qualitatively how the likelihood of one class versus the other changes over the x axis. This is the usual GP.

2. Squeeze the output of this latent function onto $[0, 1]$ using logistic function, $\pi(f) = \sigma(y = 1|f)$.

Writing these two steps schematically,

$$\boxed{\text{data, } x_*} \xrightarrow{\text{GP}} \boxed{\text{latent function, } f_*|x_*} \xrightarrow{\text{sigmoid}} \boxed{\text{class probability, } \pi(f_*)}.$$

GP-endowed Maximum Separation Model

We consider a multi-class Gaussian process classifier defined on $\mathcal{Z}$: there are K underlying latent functions $\mathbf{f}(\cdot) := \{f_j(\cdot)\}_{j=1}^{C}$, a priori independently **GP** distributed, namely

$$P(\mathbf{f}) = \prod_{j=1}^{C} P(f_j), \quad f_j \sim \mathcal{GP}\big(0, k_j(\cdot, \cdot)\big), \tag{3.11}$$

where each k_j is a covariance function of f_j, defined on $\mathcal{Z} \times \mathcal{Z}$. For an input point $z \in \mathcal{Z}$, we regard $f_j(z)$ as the model's confidence toward class j, leading to the class prediction rule:

$$\text{class}(z) = \arg \max_{1 \le j \le K} f_j(z). \tag{3.12}$$

We use the softmax likelihood model,

$$P(y = j|\mathbf{f}(z)) = \frac{e^{f_j(z)}}{\sum_{r=1}^{C} e^{f_r(z)}}, \quad \text{for } j = 1, \ldots, C. \tag{3.13}$$

Source-driven $\mathcal{H}$ Prior. The labeled source data, $\mathcal{D}_S$, induces a posterior distribution on the latent functions $\mathbf{f}$,

$$p(\mathbf{f}|\mathcal{D}_S) \propto p(\mathbf{f}) \cdot \prod_{i=1}^{N_s} P(y_i^s|\mathbf{f}(z_i^s)), \tag{3.14}$$

where $z_i^s = G(x_i^s)$. The key idea is to use Eq. 3.14 to define our hypothesis space $\mathcal{H}$. The posterior places most of its probability mass on those $\mathbf{f}$ that attain high likelihood scores on source while being smooth due to the **GP** prior.

It should be noted that we used the term *prior* of the hypothesis space $\mathcal{H}$ that is induced from the *posterior* of the latent functions $\mathbf{f}$. We use the $\mathcal{H}$ prior and the posterior of $\mathbf{f}$ interchangeably.

Note that due to the non-linear/non-Gaussian likelihood in Eq. 3.13, exact posterior inference is intractable, and one has to resort to approximate inference. We will discuss

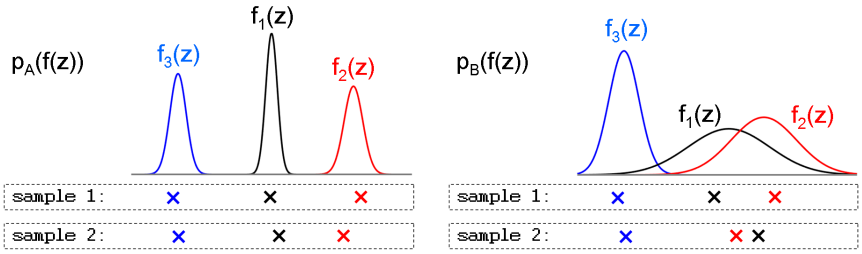


Figure 3.5: Illustration of ideal (p_A) and problematic (p_B) posteriors at some fixed point z in the target domain. For each posterior, we also depict two plausible samples (marked as crosses). In p_A, most samples $\mathbf{f}(z)$, including the two shown, are consistent in deciding the class label (class 2, red, predicted in this case). On the other hand, in p_B where $f_1(z)$ and $f_2(z)$ have considerable overlap, there is significant chance of different predictions: class 2 for the first sample and class 1 for the second.

an approach for efficient variational approximate inference in Section 3.2.1. For the exposition here, let us assume that the posterior distribution is accessible.

Target-driven Maximally Consistent Posterior. While $\mathcal{D}_S$ serves to induce the prior of $\mathcal{H}$, $\mathcal{D}_T$ will be used to reshape this prior. According to **MCD**, we want this hypothesis space to be shaped in the following way: for each target domain point $z^t = G(x^t)$, $x^t \sim P_T(x)$, the latent function values $\mathbf{f}(z^t)$ sampled from the posterior in Eq. 3.14 should lead to the class prediction (made by (3.12)) that is as consistent as possible across the samples.

This is illustrated in Figure 3.5. Consider two different $\mathcal{H}$ priors p_A and p_B at a point z, $p_A(\mathbf{f}(z))$ and $p_B(\mathbf{f}(z))$, where for brevity we drop the conditioning on $\mathcal{D}_S$ in notation. The class cardinality is $C = 3$. For simplicity, we assume that the latent functions f_j's are independent from each other. Figure 3.5 shows that the distributions of f_j's are well-separated from each other in p_A, yet overlap significantly in p_B. Hence, there is a strong chance for the class predictions to be inconsistent in p_B (identical ordering of colored samples below figure), but consistent in p_A. This means that the hypothesis space induced from p_B contains highly discrepant classifiers, whereas most classifiers in the hypothesis space of p_A agree with each other (least discrepant). In other words, the maximum discrepancy principle translates into the *maximum posterior separation* in our Bayesian **GP** framework.

We describe how this goal can be properly formulated. First we consider the posterior of $\mathbf{f}$ to be approximated as an independent Gaussian[3]. For any target domain point $z^t \sim P_T(x)$ and each $j = 1, \ldots, C$ let the mean and the variance of the $\mathcal{H}$ prior in Eq. 3.14 be:

$$\mu_j(z^t) := \int f_j(z^t)\ p\big(f_j(z^t)|\mathcal{D}_S, z^t\big)\ df_j(z^t), \tag{3.15}$$

$$\sigma_j^2(z^t) := \int (f_j(z^t) - \mu_j(z^t))^2\ p\big(f_j(z^t)|\mathcal{D}_S, z^t\big)\ df_j(z^t). \tag{3.16}$$

The maximum-a-posterior (MAP) class prediction by the model is denoted by $j^* = \arg\max_{1\le j\le C} \mu_j(z^t)$. As we seek to avoid fluctuations in class prediction j^* across samples, we consider the worst scenario where even an unlikely (e.g., at 5% chance level) sample from $f_j(z^t)$, j other than j^*, cannot overtake $\mu_{j^*}(z^t)$. That is, we seek

$$\mu_{j^*}(z^t) - \alpha\sigma_{j^*}(z^t) \ge \max_{j\neq j^*} \big(\mu_j(z^t) + \alpha\sigma_j(z^t)\big), \tag{3.17}$$

where α is the normal cutting point for the least chance (e.g., $\alpha = 1.96$ if 2.5% one-side is considered).

While this should hold for most samples, it will not hold for all. We therefore introduce an additional slack $\xi \ge 0$ to relax the desideratum. Furthermore, for ease of optimization[4], we impose slightly stricter constraint than (3.17), leading to the final constraint:

$$\max_{1\le j\le C} \mu_j(z^t) \ge 1 + \max_{j\neq j^*} \mu_j(z^t) + \alpha \max_{1\le j\le C} \sigma_j(z^t) - \xi(z^t). \tag{3.18}$$

A constant, 1 here, was added to normalize the scale of f_j's.

Our objective now is to find such embedding G, **GP** kernel parameters $\mathbb{K}$, $\mathbb{K}(z, z') = \boldsymbol{\phi}(z)^\top \boldsymbol{\phi}(z')$, and minimal slack ξ, to impose (3.18). Equivalently, we pose it as the following optimization problem, for each $z \sim P_T(z)$:

$$\min_{\Phi,\mathbb{K}} \left(\max_{j\neq j^*} \mu_j(z^t) - \max_{1\le j\le C} \mu_j(z^t) + 1 + \alpha \max_{1\le j\le C} \sigma_j(z^t) \right)_+ \tag{3.19}$$

with $(a)_+ = \max(0, a)$.

---

[3]This choice conforms to the variational density family we choose in Section 3.2.1.

[4]We used the `topk()` function in `PyTorch` to compute the largest and the second largest elements. The function allows automatic gradients.

Note that Eqs. 3.18) and 3.19 are reminiscent of the large-margin classifier learning in traditional supervised learning [121]. In contrast, we replace the ground-truth labels with the *the most confidently* predicted labels by our model since the target domain is unlabeled. This aims to place class boundaries in low-density regions, in line with *entropy minimization* or *max-margin confident prediction* principle of classical semi-supervised learning [47, 131, 111, 19].

In what follows, we describe an approximate, scalable **GP** posterior inference, where we combine the variational inference optimization with the aforementioned posterior maximum separation criterion (Eq. 3.19).

Variational Inference with Deep Kernels

We describe our scalable variational inference approach to approximate the posterior in Eq. 3.14. Although there are scalable **GP** approximation schemes based on the random feature expansion [91] and the pseudo/induced inputs [90, 105, 114, 29], here we adopt the *deep kernel* trick [58, 123] to exploit the deeply structured features. The main idea is to model an explicit finite-dimensional feature space mapping to define a covariance function. Specifically, we consider a nonlinear feature mapping $\boldsymbol{\phi} : \mathcal{Z} \to \mathbb{R}^d$ such that the covariance function is defined as an inner product in a feature space, namely $\mathbb{K}(z, z') := \boldsymbol{\phi}(z)^\top \boldsymbol{\phi}(z')$, where we model $\boldsymbol{\phi}(\cdot)$ as a deep neural network. A critical advantage of explicit feature representation is that we turn the non-parametric **GP** into a parametric Bayesian model. As a consequence, all inference operations in the non-parametric **GP** reduce to computationally more efficient parametric ones, avoiding the need to store the Gram matrix of the entire training data set, as well as its inversion.

Formally, we consider K latent functions modeled as $f_j(z) = \mathbf{w}_j^\top \boldsymbol{\phi}(z)$ with $\mathbf{w}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ independently for $j = 1, \ldots, C$. We let $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_C]^\top$. Note that the feature function $\boldsymbol{\phi}(\cdot)$ is shared across classes to reduce the number of parameters and avoid overfitting. The parameters of the deep model that represents $\boldsymbol{\phi}(\cdot)$ serve as **GP** kernel parameters, since $\mathrm{Cov}(f(z), f(z')) = \mathrm{Cov}(\mathbf{w}^\top \boldsymbol{\phi}(z), \mathbf{w}^\top \boldsymbol{\phi}(z')) = \boldsymbol{\phi}(z)^\top \boldsymbol{\phi}(z') =$

$k(z, z')$. Consequently, the source-driven $\mathcal{H}$ prior in Eq. 3.14 becomes

$$p(\mathbf{W}|\mathcal{D}_S) \propto \prod_{j=1}^{C} \mathcal{N}(\mathbf{w}_j; \mathbf{0}, \mathbf{I}) \cdot \prod_{i=1}^{N_s} P(y_i^s | \mathbf{W}\boldsymbol{\phi}(z_i^s)). \tag{3.20}$$

Since computing (3.20) is intractable, we introduce a variational density $q(\mathbf{W})$ to approximate it. We assume a fully factorized Gaussian,

$$q(\mathbf{W}) = \prod_{j=1}^{C} \mathcal{N}(\mathbf{w}_j; \mathbf{m}_j, \mathbf{S}_j), \tag{3.21}$$

where $\mathbf{m}_j \in \mathbb{R}^d$ and $\mathbf{S}_j \in \mathbb{R}^{d\times d}$ constitute the variational parameters. We further let $\mathbf{S}_j$'s be diagonal matrices. To have $q(\mathbf{W}) \approx p(\mathbf{W}|\mathcal{D}_S)$, we use the following fact that the marginal log-likelihood can be lower bounded:

$$\log P\Big(\{y_i^s\}_{i=1}^{N_s} \;\Big|\; \{z_i^s\}_{i=1}^{N_s}, \boldsymbol{\phi}(\cdot)\Big) \geq \text{ELBO}, \tag{3.22}$$

where the evidence lower-bound (ELBO) is defined as:

$$\text{ELBO} := \sum_{i=1}^{N_s} \mathbb{E}_{q(\mathbf{W})}\big[\log P(y_i^s|\mathbf{W}\boldsymbol{\phi}(z_i^s))\big] \; - \sum_{j=1}^{C} \text{KL}\big(q(\mathbf{w}_j) \;||\; \mathcal{N}(\mathbf{w}_j; \mathbf{0}, \mathbf{I})\big), \tag{3.23}$$

with the likelihood stemming from Eq. 3.13. As the gap in Eq. 3.22 is the KL divergence between $q(\mathbf{W})$ and the true posterior $p(\mathbf{W}|\mathcal{D}_S)$, increasing the ELBO wrt the variational parameters $\{(\mathbf{m}_j, \mathbf{S}_j)\}$ brings $q(\mathbf{W})$ closer to the true posterior. Raising the ELBO wrt the **GP** kernel parameters (i.e., the parameters of $\boldsymbol{\phi}$) and the embedding[5] Φ can potentially improve the marginal likelihood, i.e., the left hand side in Eq. 3.22.

In optimizing the ELBO in Eq. 3.23, the KL term (denoted by KL) can be analytically derived as

$$\text{KL} = \frac{1}{2} \sum_{j=1}^{C} \big(\; \text{Tr}(\mathbf{S}_j) + ||\mathbf{m}_j||_2^2 - \log\det(\mathbf{S}_j) - d \; \big). \tag{3.24}$$

However, there are two key challenges: the log-likelihood expectation over $q(\mathbf{W})$ does not admit a closed form, and one has to deal with large N_S. To address the former, we adopt Monte-Carlo estimation using M iid samples $\{\mathbf{W}^{(m)}\}_{m=1}^{M}$ from $q(\mathbf{W})$, where the samples are expressed in terms of the variational parameters (i.e., the reparametrization trick [64]) to facilitate optimization. That is, for each j and m,

$$\mathbf{w}_j^{(m)} = \mathbf{m}_j + \mathbf{S}_j^{1/2}\boldsymbol{\epsilon}_j^{(m)}, \qquad \boldsymbol{\epsilon}_j^{(m)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{3.25}$$

[5]Note that the inputs z also depend on Φ.

For the latter issue, we use stochastic optimization with a random mini-batch $B_S \subset \mathcal{D}_S$. That is, we optimize the sample estimate of the log-likelihood defined as:

$$\mathrm{LL} = \frac{1}{M}\sum_{m=1}^{M}\frac{N_S}{|B_S|}\sum_{i\in B_S}\log P(y_i^s|\mathbf{W}^{(m)}\boldsymbol{\phi}(z_i^s)). \tag{3.26}$$

Optimization Strategy

We combine the maximum posterior separation criterion in Eq. 3.19 with the variational inference of the previous section to arrive at the comprehensive optimization task.

Our approximate posterior in Eq. 3.21 leads to closed-form expressions for $\mu_j(z)$ and $\sigma_j(z)$ in Eqs. 3.15–3.16 as follows:

$$\mu_j(z) \approx \mathbf{m}_j^\top \boldsymbol{\phi}(z), \quad \sigma_j(z) \approx \left(\boldsymbol{\phi}(z)^\top \mathbf{S}_j \boldsymbol{\phi}(z)\right)^{1/2}. \tag{3.27}$$

With $q(\mathbf{W})$ fixed, we rewrite our posterior maximum separation loss in Eq. 3.19 as follows. We consider stochastic optimization with a random mini-batch $B_T \subset \mathcal{D}_T = \{z_i^t\}_{i=1}^{N_t}$ sampled from the target domain data.

$$\begin{aligned}\mathrm{MS} := \frac{1}{|B_T|}\sum_{i\in B_T}\Bigg(&\max_{j\neq j^*}\mathbf{m}_j^\top\boldsymbol{\phi}(z_i^t) - \max_{1\leq j\leq K}\mathbf{m}_j^\top\boldsymbol{\phi}(z_i^t) \\ &+ 1 \;+\; \alpha\max_{1\leq j\leq C}\left(\boldsymbol{\phi}(z_i^t)^\top\mathbf{S}_j\boldsymbol{\phi}(z_i^t)\right)^{1/2}\Bigg)_+\end{aligned} \tag{3.28}$$

Combining all objectives thus far, our **GPDA** algorithm[6] can be summarized as the following two optimizations alternating with each other:

- $\min_{\{\mathbf{m}_j,\mathbf{S}_j\}}\ -\mathrm{LL} + \mathrm{KL}$ (variational inference)
- $\min_{\Phi,k}\ -\mathrm{LL} + \mathrm{KL} + \lambda\cdot\mathrm{MS}$ (model selection)

with λ the impact of the max separation, e.g., $\lambda = 10.0$.

[6]In the algorithmic point of view, our algorithm can be seen as a *max-margin Gaussian process classifier* on the original input space $\mathcal{X}$ without explicitly considering the shared space $\mathcal{Z}$.

Connection to MCD Approach

Perhaps the most related to our proposed method is the Maximum Classifier Discrepancy Algorithm (**MCDA**) [97], that proposed directly optimizing the classifier discrepancy measure in the context of deep neural networks. Our approach also has a discrepancy minimization term over the predictions for target samples but the core idea in our approach is fundamentally different where we model the classifier as a random function and treat the posterior distribution of the classifier given training data, as the hypothesis space of the classifiers. This results in a simple *non-adversarial* large margin posterior separation problem, leading to highly stable convergence to a desirable solution in contrast to **MCDA** using adversarial approaches which are difficult to optimize with gradient descent and they often do not converge well without careful hyperparameter tuning and proper initialization.

3.2.2 Experimental Results

We compare the proposed method with state-of-the-art on standard benchmark datasets: Digit classification task consists of three datasets, containing ten digit classes: **MNIST** [65], **SVHN** [84], **USPS** [117]. We also evaluated our method on the traffic sign datasets, Synthetic Traffic Signs (SYN SIGNS) [83] and the German Traffic Signs Recognition Benchmark [107] (GTSRB), which contain 43 types of signs. Finally, we report performance on **VisDA** object classification dataset [87] with more than 280K images across twelve categories.

We evaluate the performance of all methods with the classification accuracy score. We used ADAM [62] for training; the learning rate was set to 0.0002 and momentum to 0.5 and 0.999. We used batches of size 32 from each domain, and the input images were mean-centered. The hyper-parameters are empirically set as $\lambda = 50.0, \alpha = 2.0$. The sensitivity w.r.t. hyperparameters λ and α will be discussed in Section3.2.2. We also used the same network structure as [97]. Specifically, we employed the CNN architecture used in [34] and [14] for digit and traffic sign datasets and used ResNet101 [55] model pre-trained on Imagenet [26]. We added batch normalization to each layer in these

models. Quantitative evaluation involves a comparison of the performance of our model to previous works and to "**Source Only**" that do not use any domain adaptation. For "**Source Only**" baseline, we train models on the unaltered source training data and evaluate on the target test data.

Results on Digit and Traffic Signs datasets

We show the accuracy of different methods in Table 3.5. It can be seen the proposed method outperformed competitors in all settings confirming consistently better generalization of our model over target data. This is partially due to combining DNNs and GPs/Bayesian approach. GPs exploit local generalization by locally interpolating between neighbors [12], adjusting the target functions rapidly in the presence of training data. DNNs have good generalization capability for unseen input configurations by learning multiple levels of distributed representations. The results demonstrate **GPDA** can improve generalization performance by adopting both of these advantages.

Results on VisDA dataset

Results for this experiment are summarized in Tabel 3.6. We observe that our **GPDA** achieved, on average, the best performance compared to other competing methods. Due to vastly varying difficulty of classifying different categories of objects, in addition to reporting the average classification accuracy we also report the average rank of each method over all objects (the lower rank, the better). The higher performance of **GPDA** compared to other methods is mainly attributed to modeling the classifier as a random function and consequently incorporating the classifier uncertainty (variance of the prediction) into the proposed loss function in Eq. 3.28. The image structure for this dataset is more complex than that of digits, yet our method exhibits very strong performance even under such challenging conditions.

Another key observation is that some competing methods, e.g., **MMD**, **DANN**, perform worse than the source-only model in classes such as car and plant, while **GPDA** and **MCDA** performed better across all classes, clearly demonstrating the effectiveness of the **MCD** principle.

Table 3.5: Classification results on the digits and traffic signs datasets. Results are cited from each study. The score of MMD is cited from DSN [15]. † indicates the method used a few labeled target samples as validation, different from our GPDA setting. We repeated each experiment five times and report the average and the standard deviation of the accuracy. The accuracy for MCDA was obtained from classifier F_1. n is MCDA's hyper-parameter, which denotes the number of times the feature generator is updated to mimic classifiers. MNIST* and USPS* denote all the training samples were used to train the models.

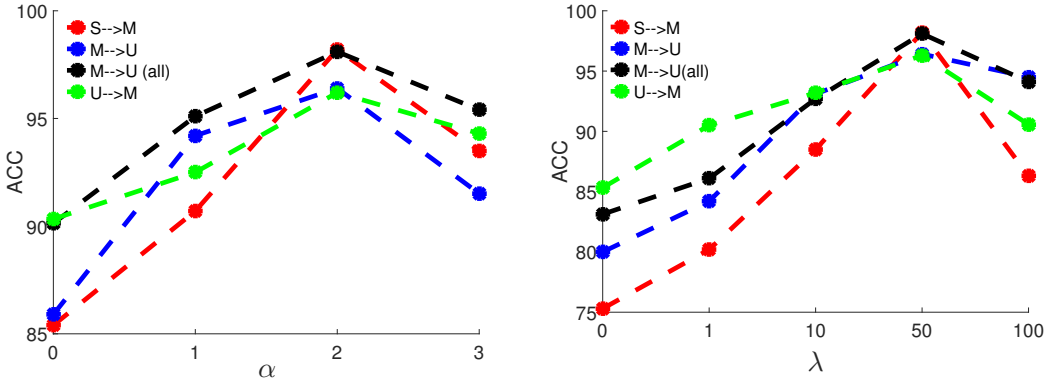
METHOD	**SVHN** to **MNIST**	**SYNSIG** to **GTSRB**	**MNIST** to **USPS**	**MNIST*** to **USPS***	**USPS** to **MNIST**
Source Only	67.1	85.1	76.7	79.4	63.4
MMD † [75]	71.1	91.1	-	81.1	-
DANN † [34]	71.1	88.7	77.1	85.1	73.0
DSN † [15]	82.7	93.1	91.3	-	-
ADDA [117]	76.0	-	89.4	-	90.1
CoGAN [74]	-	-	91.2	-	89.1
PixelDA [14]	-	-	-	95.9	-
ATDA † [95]	86.2	96.1	-	-	-
ASSC [53]	95.7	82.8	-	-	-
DRCN [38]	82.0	-	91.8	-	73.7
G2A [100]	92.4	-	92.8	95.3	90.8
SimNet [88]	-	-	-	96.4	95.6
MCDA ($n = 2$)	94.2	93.5	92.1	93.1	90.0
MCDA ($n = 3$)	95.9	94.0	93.8	95.6	91.8
MCDA ($n = 4$)	96.2	94.4	94.2	96.5	94.1
GPDA	**98.2**	**96.19**	**96.45**	**98.11**	**96.37**

Ablation Studies

Two complementary studies are conducted to investigate the impact of two hyper-parameters α and λ, controlling the trade off of the variance of the classifier's posterior distribution and the **MCD** loss term, respectively. To this end, we conducted additional experiments for the digit datasets to analyze the parameter sensitivity of **GPDA** w.r.t. α and λ, with results depicted in Figure 3.6(a) and Figure 3.6(b), respectively. Sensitivity analysis is performed by varying one parameter at the time over a given range, while for the other parameters we set them to their final values ($\alpha = 2, \lambda = 50$). From Figure 3.6(b), we see that when $\lambda = 0$ (no **MCD** regularization term), the performance drops considerably. As λ increases from 0 to 50, the performance also

Table 3.6: Accuracy of the ResNet model fine-tuned on the VisDA dataset. All models adopt ResNet101 except for [88] which used ResNet152. Last column shows the average rank of each method over all classes.

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	mean	Ave. ranking
Source Only	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4	7.25
MMD [75]	87.1	63.0	76.5	42.0	90.3	42.9	**85.9**	53.1	49.7	36.3	**85.8**	20.7	61.1	4.41
DANN [34]	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4	5.58
SimNet [88]	**94.3**	**82.3**	73.5	47.2	87.9	49.2	75.1	**79.7**	85.3	**68.5**	81.1	**50.3**	72.9	3.83
MCDA ($n = 2$)	81.1	55.3	83.6	65.7	87.6	72.7	83.1	73.9	85.3	47.7	73.2	27.1	69.7	4.25
MCDA ($n = 3$)	90.3	49.3	82.1	62.9	**91.8**	69.4	83.8	72.8	79.8	53.3	81.5	29.7	70.6	4.08
MCDA ($n = 4$)	87.0	60.9	**83.7**	64.0	88.9	**79.6**	84.7	76.9	88.6	40.3	83.0	25.8	71.9	3.00
GPDA (ours)	83.0	74.3	80.4	**66.0**	87.6	75.3	83.8	73.1	**90.1**	57.3	80.2	37.9	**73.31**	**2.75**



(a) Sensitivity to α (b) Sensitivity to λ

Figure 3.6: Sensitivity analysis of our **GPDA** on the Digit datasets. $S \rightarrow M$ denotes adaptation from **SVHN** to **MNIST** (similarly for others), and $M \rightarrow U$ (*all*) indicates using all training samples.

increases demonstrating the benefit of hypothesis consistency (**MS** term) over the target samples. Indeed, using the proposed learning scheme, we find a representation space in which we embed the knowledge from the target domain into the learned classifier. Similarly, from Figure 3.6(a), we see that when $\alpha = 0$ (no prediction uncertainty) the classification accuracy is lower than the case where we utilize the prediction uncertainty, $\alpha > 0$. The key observation is that it is more beneficial to make use of the information from the full posterior distribution of the classifier during the learning process in contrast to when the classifier is considered as a deterministic function.

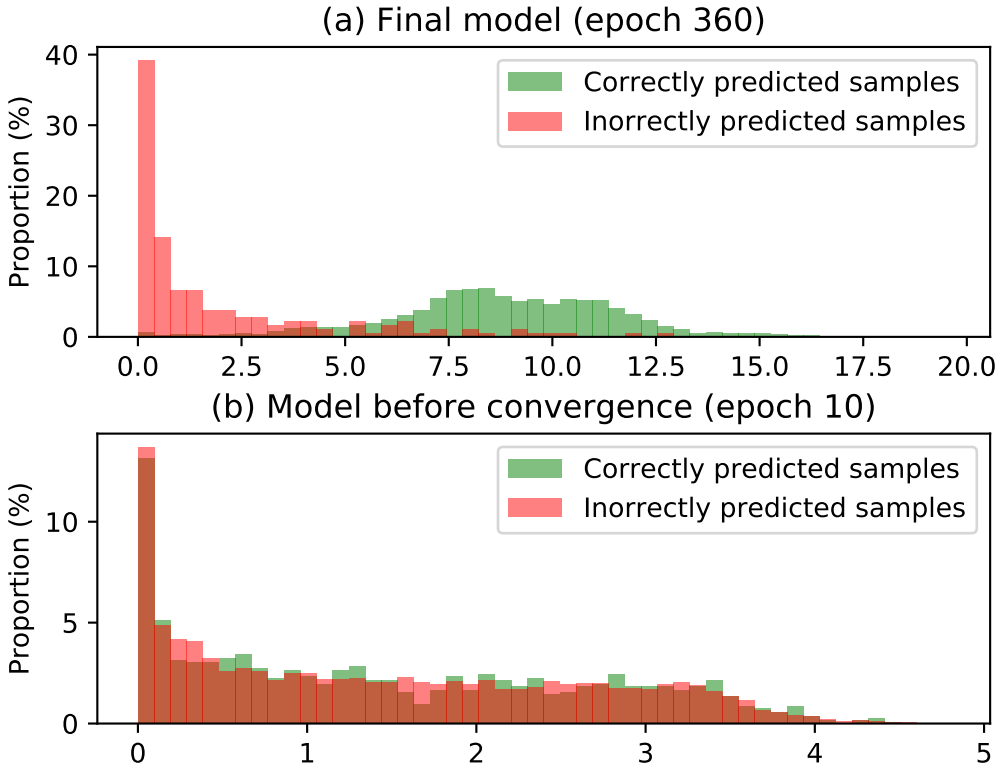


Figure 3.7: Histograms of prediction (un)certainty for our models: (a) after convergence, (b) at an early stage of training. Abscissa is the Bhattacharyya distance b/w two largest mean posteriors, an indication of *prediction certainty*; the higher the distance, the more certain the prediction is. For each model, we compute the histograms of correctly and incorrectly predicted samples (green vs. red). In our final model (a), there is a strong correlation between prediction (un)certainty (abscissa) and prediction correctness (color).

Prediction Uncertainty vs. Prediction Quality

Another advantage of our **GPDA** model, inherited from Bayesian modeling, is that it provides a quantified measure of prediction uncertainty. In the multi-class setup considered here, this uncertainty amounts to the degree of overlap between two largest mean posteriors, $p(f_{j^*}(z^t)|\mathcal{D}_S)$ and $p(f_{j^\dagger}(z^t)|\mathcal{D}_S)$, where j^* and $j^\dagger$ are the indices of the largest and the second largest among the posterior means $\{\mu_j(z^t)\}_{j=1}^C$, respectively (c.f., (3.15)). Intuitively, if the two are overlapped significantly, our model's decision is less certain, meaning that we anticipate the class prediction may not be trustworthy. On the other hand, if the two are well separated, we expect high prediction quality. To

verify this hypothesis more rigorously, we evaluate the distances between two posteriors, the measure of certainty in prediction, for two different cohorts: correctly classified test target samples by our model and the incorrectly predicted ones. Specifically, for the **SVHN** to **MNIST** adaptation task, we evaluate the Bhattacharyya distances [28] between the two cohorts. In our variational Gaussian approximation in Eq. 3.27, the Bhattacharyya distance can be computed in a closed form; Supplement for details.

The histograms of the distances are depicted in Figure 3.7, where we contrast the two models, one at an early stage of training and the other after convergence. Our final model in Figure 3.7(a) exhibits large distances for most samples in the correctly predicted cohort (green), implying well separated posteriors or high certainty. For the incorrectly predicted samples (red), the distances are small suggesting significant overlap between the two posteriors, i.e., high uncertainty. In contrast, for the model prior to convergence, Figure 3.7(b), the two posteriors overlap strongly (small distances along horizontal axis) for most samples regardless of the correctness of prediction. This confirms our algorithm enforces posterior separation by large margin during the training process. This analysis also suggests that the measure of prediction uncertainty provided by our **GPDA** model, can be used as an *indicator of prediction quality*, namely whether the prediction made by our model is trustworthy or not. To verify this, we depict some sample test images in Figure 3.8. We differentiate samples according to their Bhattacharyya distances. When the prediction is uncertain (left panel), we see that the images are indeed difficult examples even for human. An interesting case is when the prediction certainty is high but incorrectly classified (lower right panel), where the images look peculiar in the sense that humans are also prone to misclassify those with considerably high certainty.

Analysis of Shared Space Embedding

We use t-SNE [80] on **VisDA** dataset to visualize the feature representations from different classes. Figure 3.9 depicts the embedding of the learned features $\Phi(x)$, and the original features x. Colors indicate source (red) and target (blue) domains. Notice that **GPDA** significantly reduces the domain mismatch, resulting in the expected tight

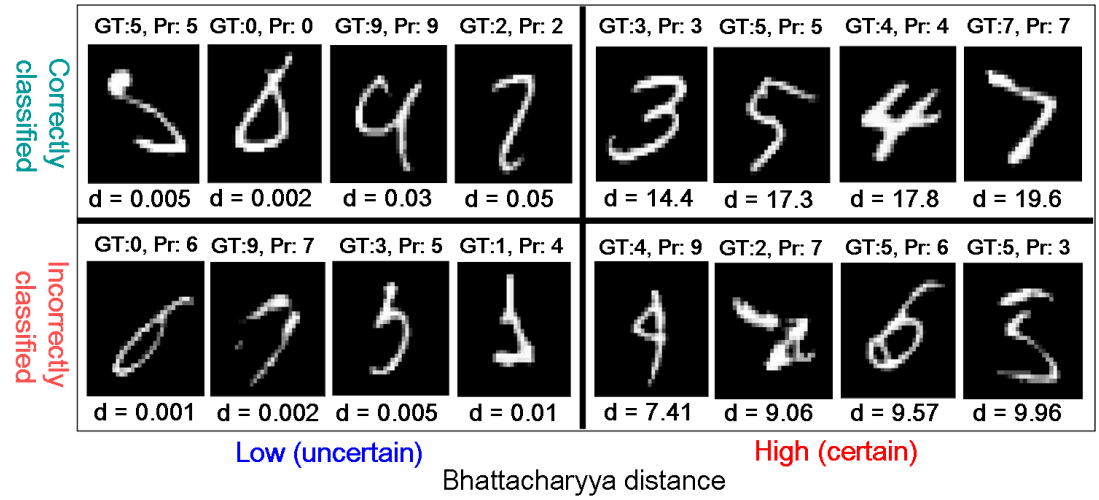
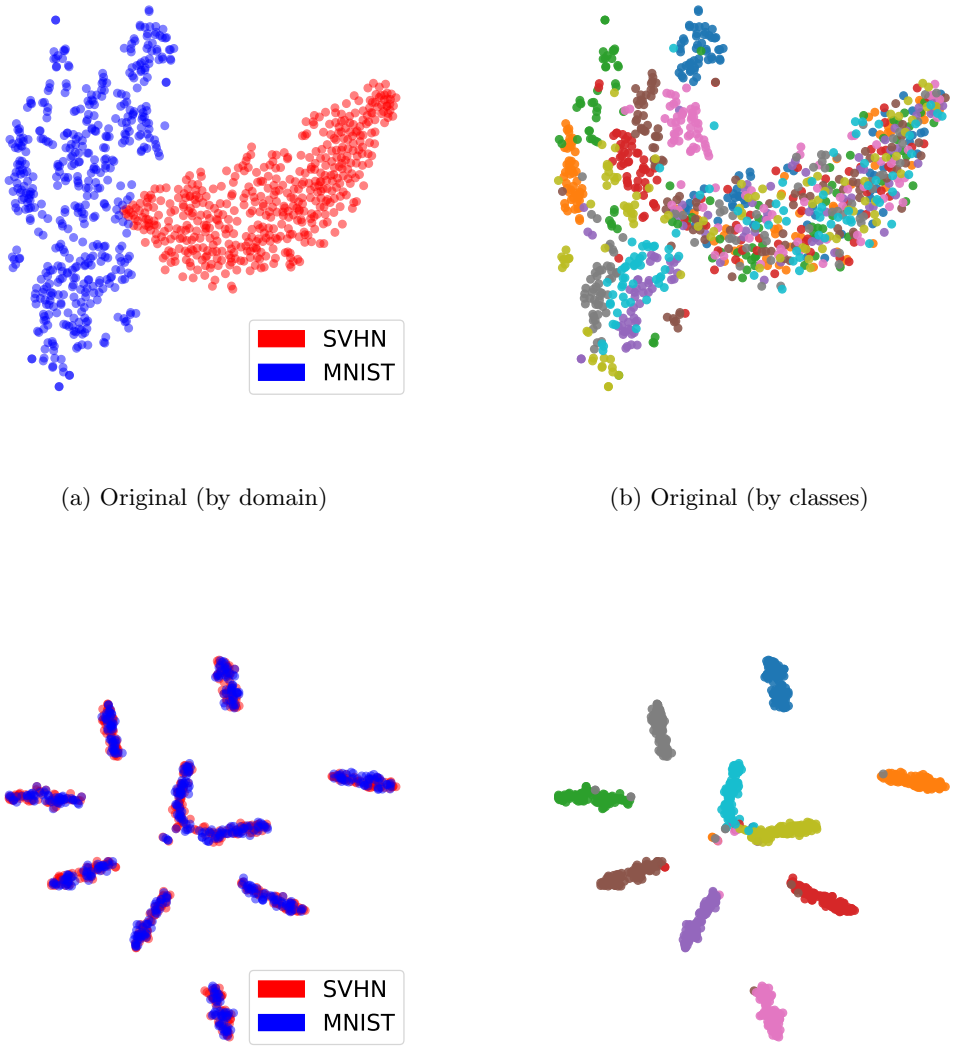


Figure 3.8: Selected test (**MNIST**) images according to the Bhattacharyya distances. Right: samples with low distances (uncertain prediction). Left: high distances (certain prediction). Top: correctly classified by our model. Bottom: incorrectly classified. For each image, GT, Pr, and d means ground-truth label, predicted label, and the distance, respectively.

clustering. This is partially due to the use of the proposed probabilistic **MCD** approach, which shrinks the classifier hypothesis class to contain only consistent classifiers on target samples while exploiting the uncertainty in the prediction.

3.2.3 Summary

In this section, we proposed a novel *probabilistic* approach for UDA that learns an efficient domain-adaptive classifier with strong generalization to target domains. The key is to model the classifier's hypothesis space in Bayesian fashion and impose consistency over the target samples in their space by constraining the classifier's posterior distribution. To tackle the intractability of computing the exact posteriors, we combined the variational Bayesian method with a deep kernel technique to efficiently approximate the classifier's posterior distribution. We showed, on three challenging benchmark datasets for image classification, that the proposed method outperforms current state-of-the-art in unsupervised domain adaptation of visual categories.



(a) Original (by domain)

(b) Original (by classes)

(c) GPDA (by domain)

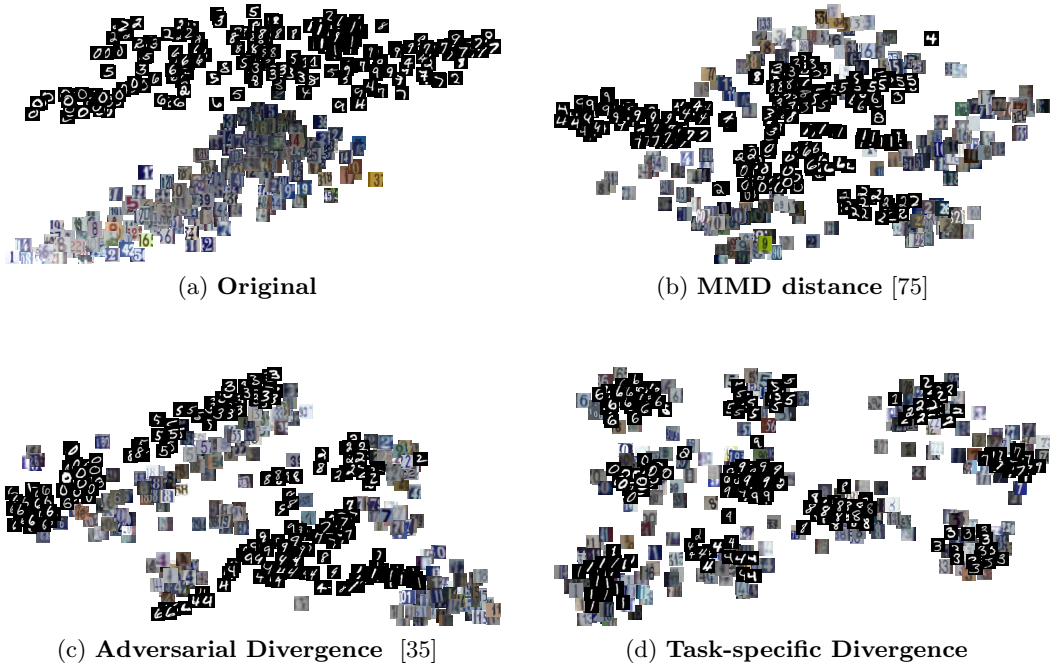
(d) GPDA (by classes)

Figure 3.9: Feature visualization for embedding of digit datasets for adapting **SVHN** to **MNIST** using t-SNE algorithm. The first and the second columns show the domains and classes, respectively, with color indicating domain and class membership. (a),(b): Original features. (c),(d): learned features for **GPDA**.

3.3 Task-Discriminative Domain Alignment for Unsupervised Domain Adaptation

In the previous two sections, we presented two methods using the process of domain alignment in some shared space. However, this process often leads to unsatisfactory adaptation performance, in part because it ignores the task-specific structure of the data. In this section, we improve the performance of **DA** by introducing a discriminative discrepancy measure which takes advantage of auxiliary information available in the source and the target domains to better align the source and target distributions. Specifically, we leverage the cohesive clustering structure within individual data manifolds, associated with different tasks, to improve the alignment. This structure is explicit in the source, where the task labels are available, but is implicit in the target, making the problem challenging. We address the challenge by devising a deep **DA** framework, which combines a new task-driven domain alignment discriminator with domain regularizers that encourage the shared features as task-specific and domain invariant, and prompt the task model to be data structure preserving, guiding its decision boundaries through the low density data regions.

Existing discrepancy approaches, reviewed in Section 2.1, mainly focus on aligning domain-level feature distributions without considering category-level alignment. Thus, the alignment enforced by such discrepancy measures does not guarantee a good target performance as it ignores the cluster structure of the samples, aligned with their task labels. The assumption that the source features exhibit a well-defined cluster structure naturally transfers to the target: target features indicative of the same tasks as the source should manifest a similar cluster structure. In other words, when optimally aligned, the target features should amass around the source clusters such that the decision boundaries of the learned task classifiers do not induce partitioning of smooth clusters of target features. However, the aforementioned domain discrepancy measures only focus on global feature overlap, ignoring the finer task-aligned structure in the data. Consequently, they may inaccurately match the clusters and also cause the source features to form weakly separable clusters, as illustrated in Figure 3.10 ((b), (c)).



(a) **Original** (b) **MMD distance** [75]

(c) **Adversarial Divergence** [35] (d) **Task-specific Divergence**

Figure 3.10: Feature visualization for embedding of digit datasets for adapting **SVHN** to **MNIST** using t-SNE algorithm. The first and the second columns show the domains and classes, respectively, with color indicating domain and class membership. (a),(b): Original features. (c),(d): learned features for **GPDA**.

To alleviate the limitations of existing discrepancy measures for domain adaptation, we introduce a task (e.g., classification)-specific adversarial discrepancy measure that extends the discriminator output over the source classes, in order to additionally incorporate task knowledge into the adversarial domain alignment. The new discrepancy measure helps the feature extractor (encoder) make discriminative source/target features by considering the decision boundary information. Consequently, source-target alignment not only takes into account the domain-level feature densities but also the category-conditioned clusters-of-features information to produce an improved overlap, evident in Figure 3.10 (d).

Motivated by the information-bottleneck principle [113], whose goal is to improve generalization by ignoring irrelevant (domain-variant) distractors present in the original data features, we also introduce a **source regularization** loss by minimizing the information between the source samples and their features by encouraging the marginal

distribution of the source features to be similar to a prior distribution (the standard normal distribution) to enforce the model to focus only on the most discriminative (label-variant) features, less prone to overfitting. Moreover, an additional **target regularization** term is imposed on the classifier, trained on the shared features of the source samples, to encourage it not to pass through high-density regions of the target data. Previous **DA** methods did not explicitly consider these desiderata. Our ablation study in Section 3.3.4 empirically demonstrates the importance of the introduced objectives.

3.3.1 Problem Formulation

In this section, we introduce a shared (stochastic) encoder Q (rather than a deterministic one introduced in previous sections) between the source and the target domains that maps a sample x into a stochastic embedding[7] $z \sim Q(z|x)$, and then apply a classifier h to map z into the label space $y \sim h(y|z)$ (h is trained to classify samples drawn from the encoder distribution). Although one can consider domain-wise different encoders, more recent **DA** approaches tend to adopt a shared encoder, which can prevent domain-specific nuisance features from being learned, reducing potential overfitting issues.

We define the stochastic encoder Q as a **conditional Gaussian** distribution with **diagonal** covariance that has the form $Q(z|x) = \mathcal{N}(\mathbf{z}|\mathbf{f}_\mu(\mathbf{x}), \mathbf{f}_{\mathbf{\Sigma}}(\mathbf{x}))$ where $\mathbf{f}$ is a deep network mapping the data point x to the $2p$-dimensional latent code, with the first p outputs from $\mathbf{f}$ encoding $\mathbf{f}_\mu$, and the remaining p outputs encoding $\mathbf{f}_\Sigma$. The classifier h outputs a C-dim probability vector of class memberships, modeled as a softmax form $h(z) = \text{softmax}(\mathbf{f}_c(z))$,where $\mathbf{f}_c(z)$ is a deep network mapping the latents $\mathbf{z}$ to the logits of C classes.

Remark 1. *The reason to choose a stochastic encoder over a deterministic one is two fold. First, it allows one to impose smoothness (local-Lipschtizness) constraint on the classifier h over target samples; see Section 3.3.1 for more details. Second, adding*

[7]Please see Remark 1 for the benefits of choosing a stochastic encoder over a deterministic one.

continuous noise to the inputs of the discriminators has been shown to improve instability and vanishing gradients in adversarial optimization problems through smoothening the distribution of features [5]. Our stochastic encoder equipped with the reparametarization approach inherently provides such mechanism to feature distribution smoothness; see Sections 3.3.1 and 3.3.1 for more details.

The proposed domain adaptation method can be summarized by the objective function consisting of six terms:

$$\mathcal{L}_{Class} + \mathcal{L}_{Disc} + \mathcal{L}_{Teach} + \mathcal{L}_{Smooth} + \mathcal{L}_{Entropic} + \mathcal{L}_{Adv}, \tag{3.29}$$

where $\mathcal{L}_{Class}$ is the classification loss applied to $\mathcal{D}_S$, $\mathcal{L}_{Disc}$ is the domain discrepancy loss measuring the discrepancy between the source and target distribution, $\mathcal{L}_{Teach}$ is the source-to-target teaching loss, which couples the source classifier with the target discriminator. The remaining losses, $\mathcal{L}_{Smooth}, \mathcal{L}_{Entropic}, \mathcal{L}_{Adv}$ will impose different regularization constraints on the model: $\mathcal{L}_{Smooth}$ will impose Lipschitz classifiers in the target space, $\mathcal{L}_{Entropic}$ will strive to drive the classifier towards regions of low density in the same target space, while $\mathcal{L}_{Adv}$ will impose regularization towards a reference density in the shared space $\mathcal{Z}$. We next discuss each of the above losses in more detail and then propose an algorithm to efficiently optimize the desired objective.

Source Classification Loss $\mathcal{L}_{Class}$

Having access to source labels, the stochastic mappings Q and h are trained on source samples to correctly predict the class label by minimizing the standard cross entropy loss,

$$\mathcal{L}_{Class}(Q, h) := -\mathbb{E}_{x,y\sim P_s(x,y)}\Big[\mathbb{E}_{z\sim Q(z|x)}\big[y^\top \log h(z)\big]\Big], \tag{3.30}$$

where y is the C-D one-hot vector representing the label y.

Domain Discrepancy Loss $\mathcal{L}_{Disc}$

Since the stochastic encoder Q is shared between the source and target samples, to make sure the source and the target features are well aligned in the shared space

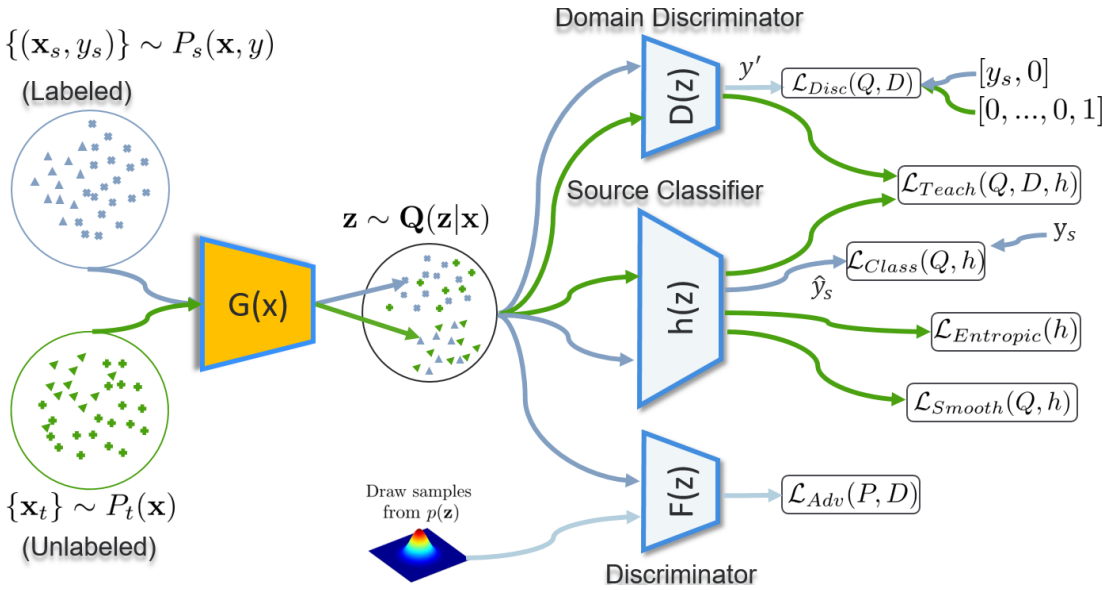


Figure 3.11: **Proposed architecture** includes a deep feature extractor $G(x)$ and a deep label predictor $h(z)$, which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a task-specific discriminator $D(z)$ connected to the feature extractor distinguishing the source from target features. The training proceeds standardly and minimizes the label prediction loss (for source examples) $\mathcal{L}_{Class}$, the domain discrepancy losses (for all samples) $\mathcal{L}_{Disc}$ and $\mathcal{L}_{Teach}$, the source domain regularization loss $\mathcal{L}_{Adv}$, and the target domain regularization losses $\mathcal{L}_{Smooth}$ and $\mathcal{L}_{Entropic}$.

and respect the cluster structure of the original samples, we propose a novel domain alignment loss, which will be optimized in adversarial manner.

Rather than using the standard adversarial approach to minimizing the alignment loss between the source and the target densities in the shared space $\mathcal{Z}$, i.e., finding the encoder Q which "fools" the best binary discriminator D trying to discern source from target samples, our approach is inspired by semi-supervised GANs [25] where it has been found that incorporating task knowledge into the discriminator can jointly improve classification performance and quality of images produced by the generator. We incorporate task knowledge by replacing what would be a binary discriminator with a $(C + 1)$-way multi-class discriminator $y' = D(z) = \text{softmax}(\mathbf{f}_d(z))$. The first C classes indicate that a sample z belongs to the source domain *and* belongs to a specific classes in $\mathcal{Y}$, while the last $(K + 1)$-th class "t" indicates z belongs to the target domain.

Since we have the class label for the source samples, the discriminator is trained to classify source features correctly, hence creating crisp source clusters in the feature space. On the other hand, the new discriminator seeks to distinguish the samples from

the target domain from those of the source by assigning them to the $(C+1)$-th, "target" class.

$$\mathcal{L}_{Disc}(Q,D) := -\mathbb{E}_{x^t\sim P_T(x)}\left[\mathbb{E}_{z^t\sim Q(z|x^t)}\left[[\mathbf{0},1]^\top \log D(z^t)\right]\right] - \mathbb{E}_{x^s,y^s\sim P_S(x,y)}\left[\mathbb{E}_{z^s\sim Q(z|x^s)}\left[[y^s,0]^\top \log D(z^s)\right]\right], \quad (3.31)$$

where $[\mathbf{0},1]$ is a one-hot vector indicating a point from the target domain and $[y,0]$ stands for a point from the source domain, labeled according to class label y.

Teacher Target-Source Loss $\mathcal{L}_{Teach}$

Here, we seek the encoder Q to generate a feature representative of one of the first C task-specific classes for target samples preserving their cluster structure and aligning them to the source clusters in the feature space. However, the target data points are unlabeled, and the encoder will not have the chance to enforce the desired clustering structure of the target points, where points within a cluster would have the same predicted label. To "teach" the encoder, we ask the classifier $h(\cdot)$ to provide pseudo soft labels for the target points to our new discriminator using the following loss:

$$\mathcal{L}_{Teach}(Q,D,h) := -\mathbb{E}_{x^t\sim P_T(x)}\left[\mathbb{E}_{z^t\sim Q(z|x^t)}[[h(z^t),0]^\top \log D(z^t)]\right]. \quad (3.32)$$

Intuitively, the encoder tries to fool the discriminator by assigning one of the first C classes to target features, leveraging on the output of the classifier h (augmented with 0 for the $C+1$-th dimension) as pseudo-labels for target features.

Remark 2. *The proposed task-specific domain discriminator can be used to improve any domain adaptation method that has an adversarial domain alignment component. Indeed, we observe (see Section 3.3.4) that the proposed discriminator significantly improves upon the standard binary discriminator.*

Source Domain Regularization Loss

One of the standard goals in representation learning is to find an encoding of the data point x that is maximally expressive about its label y while being maximally compressive

about x—finding a representation z which ignores as many details of x as possible. This is specifically useful for domain adaptation where we require a representation to be domain invariant. Essentially, we want z to act like a minimal sufficient statistic of x for predicting y in order to generalize better for samples form unseen domains. To do so, we introduce a regularizer that acts on the aggregated posterior of the shared features of the source samples $Q_z(z^s) = \mathbb{E}_{x^s \sim P_S(x)}[Q(z|x^s)]$. The regularizer encourages z to be less informative about x in the form of mutual information by matching the aggregated posterior of the shared features with a factorized prior distribution $P_z(z^s)$[8], which in turn constrains the implicit capacity of z^s and encourages it be factorized:

$$\mathcal{D}\big[P_z(z^s)||Q_z(z^s)\big], \tag{3.33}$$

where $\mathcal{D}(\cdot||\cdot)$ is an arbitrary distribution divergence measure.

As the proxy for this divergence, we define an auxiliary loss which will be **adversarially** optimized. We introduce an a binary discriminator F in the latent space trying to separate ***true*** points sampled from P_z and ***fake*** ones sampled from Q_z. The encoder Q ensures the aggregated posterior distribution Q_z can fool the binary discriminator into thinking that the source features comes from the distribution P_z:

$$\mathcal{L}_{Adv}(Q, F) \;=\; -\mathbb{E}_{x^s \sim P_S(x^s)}\big[\mathbb{E}_{z^s \sim Q(z|x^s)}\big[\log F(z^s)\big]\big] \;-\; \mathbb{E}_{z^s \sim P(z^s)}\big[\log(1 \,-\, F(z^s))\big]. \tag{3.34}$$

Remark 3. *We empirically observed that imposing such regularization on target samples could be harmful to performance. We conjecture this is due to the lack of true class labels for the target samples, without which the encoder would not preserve the label information of the features, leading to unstructured target points in feature space.*

Target Domain Regularization Losses

In order to incorporate the target domain information into the model, we apply the cluster assumption, which states that the target data points $\mathcal{D}_T$ contains clusters and that points in the same cluster have homogeneous class labels. If the cluster assumption

[8] In this work, we consider $P_z(z^s) = \mathcal{N}(0, \mathbf{I})$

holds, the optimal decision boundaries should occur far away from data-dense regions in the feature space z. We achieve this by defining an entropic loss,

$$\mathcal{L}_{Entropic}(h, Q) := -\mathbb{E}_{x^t \sim P_T(x)} \Big[\mathbb{E}_{z^t \sim Q(z|x^t)} \big[h(z^t)^\top \log h(z^t) \big] \Big]. \tag{3.35}$$

Intuitively, minimizing the conditional entropy forces the classifier to be confident on the unlabeled target data, thus driving the classifier's decision boundaries away from the target data. In practice, the conditional entropy must be empirically estimated using the available data.

However, Grandvale [48] suggested this approximation can be very poor if h is not locally-Lipschitz smooth. Without the smoothness constraint, the classifier could abruptly change its prediction in the neighborhood of training samples, allowing decision boundaries close to the training samples even when the empirical conditional entropy is minimized. To prevent this, we take advantage of our stochastic encoder and propose to explicitly incorporate the locally-Lipschitz constraint in the objective function,

$$\mathcal{L}_{Smooth}(h, Q) := \mathbb{E}_{x^t \sim P_T(x)} \Big[\mathbb{E}_{z_1^t, z_2^t \sim Q(z|x^t)} ||h(z_1^t) - h(z_2^t)||_1 \Big], \tag{3.36}$$

with $\| \cdot \|_1$ the L_1 norm. Intuitively, we enforce classifier consistency over proximal features of any target point x^t.

Remark 4. *We empirically observed that having such constraints for source features would not improve performance. This is because access to the source labels and forcing the classifier to assign each source feature to its own class would already fulfill the smoothness and entropy constraints on the classifier for the source samples.*

3.3.2 Model Learning and Loss Optimization

Our goal is to train the task-specific discriminator D, binary discriminator F, classifier h, and encoder Q to facilitate learning of the cross-domain classifier h. By approximating the expectations with the sample averages, using the stochastic gradient Descent (SGD), and the reparametarization approach [63], we solve the optimization task in the following four subtasks.

Optimizing the encoder Q

$$Q^* = \arg\min_{Q} \mathcal{L}_{Class}(Q, h^*) + \mathcal{L}_{Disc}(Q, D^*, h^*) + \lambda_Q\big[\mathcal{L}_{Adv}(Q, F^*)\big], \quad (3.37)$$

where λ_Q is a weighting factor. Intuitively, The first term in Eq. 3.37 encourages Q to produce discriminative features for the labeled source samples to be correctly classified by the classifier h. The second term simulates the adversarial training by encouraging Q to fool the task-specific discriminator D by pushing the target features toward the source features, leveraging the soft pseudo-labels provided by the classifier. Through the last term, the encoder seeks to fool the binary discriminator F into treating the source features as if they come from the fully-factorized $P(z)$ to produce domain-invariant source features.

Optimizing the classifier h

$$h^* = \arg\min_{h} \lambda_h[\mathcal{L}_{Class}(Q^*, h)] + \lambda'_h\big[\mathcal{L}_{Entropic}(Q^*, h) + \mathcal{L}_{Smooth}(Q^*, h)\big], \quad (3.38)$$

where λ_h and λ'_h are the trade-off factors. Intuitively, we enforce the classifier h to correctly predict the class labels of the source samples by the first term in Eq. 3.38. We use the second term to minimize the entropy of h for the target samples, reducing the effects of "confusing" labels of target samples. The last term guides the classifier to be locally consistent, shifting the decision boundaries away from target data-dense regions in the feature space.

Optimizing the task-specific discriminator D

$$D^* = \arg\min_{D} \mathcal{L}_{Disc}(Q^*, D). \quad (3.39)$$

The loss in Eq. 3.39 prompts D to shape its decision boundary to separate the source features (according to their class label) and target features from each other.

Optimizing the binary discriminator F

$$F^* = \arg\min_F \ \mathcal{L}_{Adv}(Q^*, F). \tag{3.40}$$

Intuitively, the loss in Eq. 3.40 encourages F to separate the source features from the features generated from the fully-factorized distribution $P_z(z)$ by assigning label 1 and 0 to the source feature samples and $P_z(z)$ samples, respectively.

3.3.3 Target Class Label Prediction

After model training, to determine the target class-label y^t of a given target domain instance x^t, we first compute the distribution of y^t given x^t by integrating out the shared feature z^t. Then, we select the most likely label as

$$\hat{y}^t = \arg\max_{y^t \in \{1,\dots,C\}} P(y^t|x^t), \tag{3.41}$$

where $P(y^t|x^t)$ can be computed as

$$P(y_t = c|z^t) = \mathbb{E}_{z^t \sim Q(z|x^t)=\mathcal{N}(\mathbf{f}_\mu(x^t),\mathbf{f}_\Sigma(x^t))}[h_c(z^t)], \tag{3.42}$$

where $h_c(.)$ is the c-th entry of the classifier output. Since the above expression cannot be computed in a closed form, we approximate it with its mean value. Using this approximation, we compute y^t as:

$$\hat{y}^t = \arg\max_{c \in \{1,\dots,C\}} h_k(z^t), \quad z^t = \mathbf{f}_\mu(x^t). \tag{3.43}$$

Remark 5. *We empirically observed that estimating the expectation in Eq. 3.42 with Gibbs sampling from the posterior $Q(z|x^t)$ instead of its mean would not boost the performance. We conjecture this is due to the smoothness constraint we impose on the classifier through Eq. 3.36, enforcing consistency over proximal target samples drawn from $Q(z|x^t)$.*

3.3.4 Experimental Results

We compare our proposed method with state-of-the-art on three benchmark tasks. The Digit datasets embody the digit cross-domain classification task across four datasets:

MNIST, **MNIST-M**, **SVHN**, **USPS**, which consist of $K = 10$ digit classes (0-9). We also evaluated our method on **VisDA** object classification dataset [87] with more than 280K images across twelve categories. Finally, we report performance on **PACS** [68] containing 9991 images of seven categories extracted from four different domains:*Photo* (P), *Art paintings* (A), *Cartoon* (C), and *Sketch* (S). We evaluate the performance of all methods with the classification accuracy metric. We used ADAM [62] for training; the learning rate was set to 0.0002 and momentums to 0.5 and 0.999. Batch size was set to 16 for each domain, and the input images were mean-centered/rescaled to $[-1, 1]$. All the used architectures replicate those of state-of-the-art methods. We also set the hyper-parameters $\lambda_Q = 0.4, \lambda_h = 0.05, \lambda'_h = 0.01$. We compare the proposed method with several related methods, including **CORAL** [110], **DANN** [34], **ADDA** [117], **DTN** [129], **UNIT** [72], **PixelDA** [14], **DIAL** [18], **DLD** [81], **DSN** [15], and **MCDA** [96] on digit classification task (Digit datasets), and the object recognition task (VisDA and PACS datasets).

Results On Digits Recognition

In this evaluation, we follow the same protocol across all methods. Specifically, we use the network structure similar to **UNIT** [72].

We show the accuracy of different methods (averaged over five different runs) in Table 3.7. The proposed method outperformed the competing methods in five out of six settings, confirming consistently and significantly better generalization of our model over target data.

The higher performance of the proposed model compared to other methods is mainly attributed to the proposed task-specific alignment method, which not only encourages the source features to be well-separated, according to their class label, but also aligns the target to source features in a cluster-wise manner, "matching" the source and target clusters. This is in contrast to the standard domain-wise alignment, which ignores the source/target inherent cluster structure. This superiority also benefits from the proposed source and target domain regularizers, which improve the source feature domain-invariance and the classifier's robustness respectively. See Section 3.3.4 for more

Table 3.7: Mean classification accuracy on digit classification. M: MNIST; MM: MNIST-M, S: SVHN, U: USPS. The best is shown in red. The superscript shows the standard deviation. *__UNIT__ trains with the extended __SVHN__ (> 500K images vs ours 72K). *__PixelDA__ uses ($\approx 1,000$) of labeled target domain data as a validation set for tuning the hyper-parameters.

method	S → M	M → MM	M → U	MM → M	MM → U	U → M
Source Only	62.10	55.98	78.30	84.46	80.43	50.64
1-NN	35.86	12.58	41.22	82.13	36.90	38.45
CORAL [110]	63.10	57.70	$81.05^{0.6}$	84.90	87.54	85.01
DANN[35]	73.80	77.40	81.60	61.05	85.34	77.40
ADDA[117]	77.68	91.47	90.51	92.82	80.70	90.10
DTN[129]	81.40	85.70	85.80	88.80	90.68	89.04
PixelDA[14]	–	**98.10***	94.10*	–	–	–
UNIT[72]	90.6*	–	92.90	–		90.60
DSN[15]	82.70	83.20	91.65	90.20	89.95	91.40
MCDA[96]	**96.20**	–	96.50	–	–	94.10
Ours	94.67	**98.01**	**99.05**	**99.11**	**99.16**	**97.85**

details.

Results on Object Recognition

We also evaluate our method on two object recognition benchmark datasets **VisDA** [87] and **PACS** [68]. We follow **MCDA** [96], and use ResNet101 [55] as the backbone network which was pretrained on ImageNet dataset, and then finetune the parameters of ResNet101 with the source only **VisDA** dataset according to the procedure described in [96]. For the **PACS** dataset, we also follow the experimental protocol in [81], using ResNet18 [55] pretrained on ImageNet dataset, and training our model considering 3 domains as sources and the remaining as target, using all the images of each domain. For these experiments, we set the learning rate of resnets to 10^{-9}. We choose this small learning rate for ResNet parameters since the domain shift for both **VisDA** and **PACS** are significant, the training procedure benefits from a mild parameter updates back-propagated from the loss. Results for this experiment are summarized in Tables 3.8 & 3.9. We observe that our model achieved, on average, the best performance compared to other competing methods for both datasets. The higher performance of our method is mainly attributed to incorporating the category-level information into the domain

Table 3.8: Accuracy of ResNet101 model fine-tuned on the VisDA dataset. Last column shows the average rank of each method over all classes. The best in bold red, second best in red.

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	mean	Ave. ranking
Source Only	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4	4.91
MMD [75]	87.1	63.0	76.5	42.0	**90.3**	42.9	**85.9**	53.1	49.7	36.3	**85.8**	20.7	61.13	3.08
DANN [34]	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	**54.6**	82.8	7.8	57.42	3.00
MCDA [96]	87.0	60.9	**83.7**	64.0	88.9	79.6	84.7	**76.9**	88.6	40.3	83.0	**25.8**	71.90	2.41
Ours	**88.2**	**78.5**	79.7	**71.1**	90.0	**81.6**	84.9	72.3	**92.0**	52.6	82.9	18.4	**74.03**	**1.83**

Table 3.9: Mean classification accuracy on **PACS** dataset. The first row indicates the target domain, while all the others are considered as sources.

method	Sketch	Photo	Art	Cartoon	Mean
Resnet18 (Source Only)	60.10	92.90	74.70	72.40	75.00
DIAL [18]	66.80	**97.00**	87.30	85.50	84.20
DLD [81]	69.60	**97.00**	87.70	86.90	85.30
Ours	**71.69**	96.81	**89.48**	**88.91**	**86.72**

alignment through the proposed task-specific discriminator, which is beneficial to boost the discriminability of the source/target features.

Analysis of the task-specific discriminator

To measure how effective the new task-specific discriminator is, we conducted an experiment to compare the task-specific discriminator with the standard adversarial discriminator (training a logistic function on the discriminator by assigning labels 0 and 1 to the source and target domains respectively and training the encoder with inverted labels). The results are shown in Figure 3.12. As is evident from the figure, there is a substantial increase in accuracy over all adaptation scenarios on switching from the standard adversarial discriminator to our task-specific discriminator. The superiority of the performance is mainly due to explicitly accounting for task knowledge in the proposed discriminator during adversarial training that encourages the discriminativity of the source/target samples in the feature space.

We further visualize the distribution of the learnt shared features to investigate the effect of task-specific discriminator (**Task-d**) and its comparison to adversarial

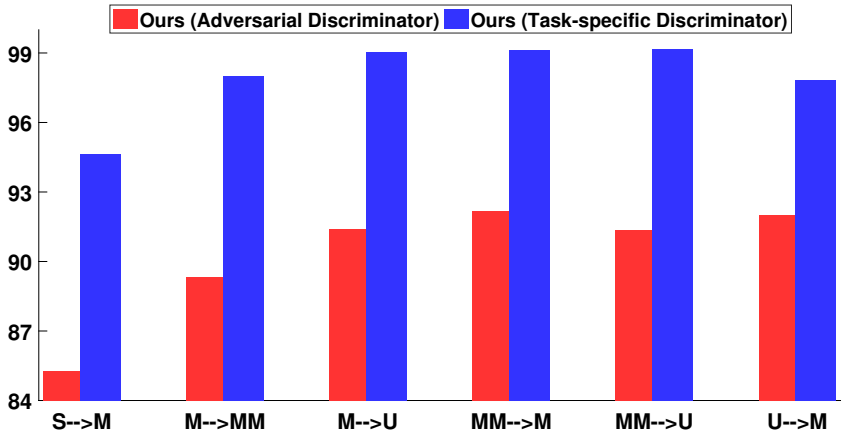
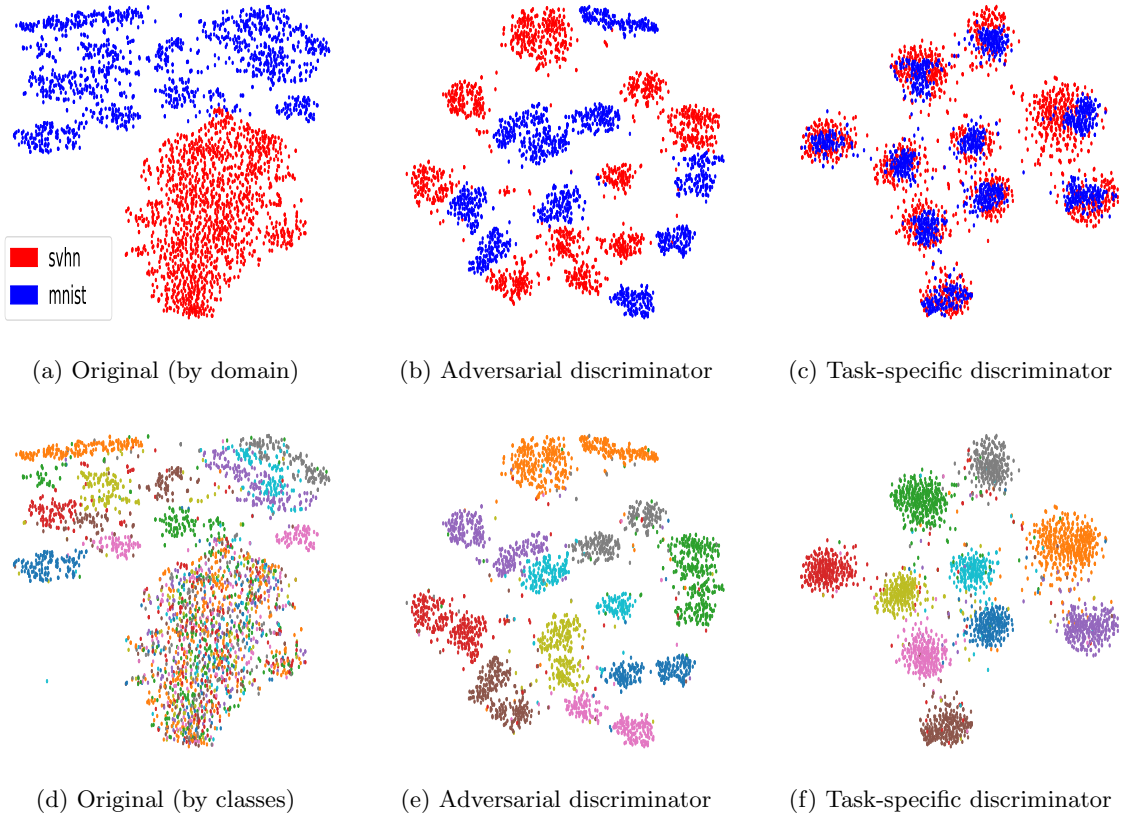


Figure 3.12: Comparison of proposed task-specific discriminator with the standard adversarial discriminator on Digit dataset.

discriminator (**Adv-d**). We use t-SNE [80] on **SVHN** to **MNIST** adaptation to visualize shared feature representations from two domains. Figure 3.13 shows shared features from source (**SVHN**) and target (**MNIST**) before adaptation (a),(d), after adaptation with **Adv-d** (b),(e), and after adaptation with **Task-d** (c),(f). While a significant distribution gap is present between non-adapted features across domains (a), the domain discrepancy is significantly reduced in the feature space for both **Adv-d** (b) and **Task-d** (c). On the other hand, adaptation with **Task-d** led to pure and well-separated clusters in feature space compared to the adaptation with **Adv-d**, and leads to superior class separability. As supported by the quantitative results in Figure 3.12, this implies that enforcing clustering in addition to domain-invariant embedding was essential for reducing the classification error. This is depicted in (f), where the points in the shared space are grouped into class-specific subgroups; color indicates the class label. This is in contrast to (e), where the features show less class-specificity.

Ablation Studies

We performed an ablation study for our unsupervised domain adaptation approach on Digit dataset. Specifically, we considered training without source regularization, denoted as **Ours (w/o-s)**, training without target regularization, **Ours (w/0-t)**, and



(a) Original (by domain) (b) Adversarial discriminator (c) Task-specific discriminator

(d) Original (by classes) (e) Adversarial discriminator (f) Task-specific discriminator

Figure 3.13: Feature visualization for embedding of digit datasets for adapting **SVHN** to **MNIST** using t-SNE algorithm. The first and the second rows show the domains and classes, respectively, with color indicating domain and class membership. (a),(d) Original features. (b),(e) learned features for Ours with (binary) adversarial discriminator. (c),(f) learned features for Ours with task-specific discriminator.

training by excluding both the source and the target regularization, **Ours (w/o-st)**.

The results are shown in Figure 3.14. As can be seen, removing one or more of the objectives results in noticeable performance degradation. The more parts are removed, the worse the performance is. More precisely, disabling the source regularizer results in an average $\approx 3.5\%$ drop in performance. That demonstrates that the source regularizer can improve the generalization over target samples by encouraging the source features to be domain-invariant, less informative about the identity of either of the domains. Immobilizing the target regularizer leads to $\approx 2.0\%$ average drop in performance. These results strongly indicate that it is beneficial to make use of the information from unlabeled target data the during classifier learning process, which further strengthens

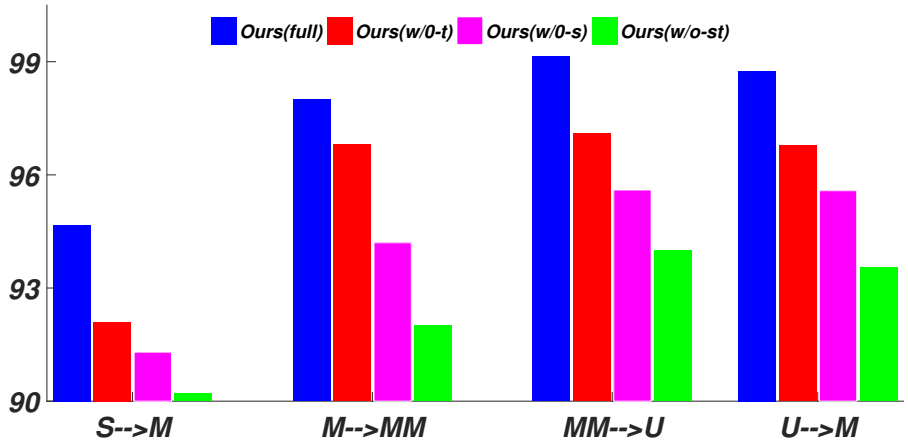


Figure 3.14: Ablation of the proposed method on Digit dataset. The regularization terms contribute to the overall performance.

the feature discriminability in the target domain. Finally, the average performance drop that stems from disabling both the source and the target regularizer is $\approx 5.5\%$. This suggests that the two components operate in harmony with each other, forming an effective solution for domain adaptation.

3.3.5 Summary

In this section, we proposed a method to boosts the unsupervised domain adaptation by explicitly accounting for task knowledge in the cross-domain alignment discriminator, while simultaneously exploiting the agglomerate structure of the unlabeled target data using important regularization constraints. Our experiments demonstrate the proposed model achieves state-of-the-art performance across several domain adaptation benchmarks.

3.4 MTDA-ITA: An Information Theoretic Approach for Unsupervised Multi-Target Domain Adaptation

The methods presented in previous sections, focus on pairwise adaptation settings where there is a single, labeled, source and a single target domain. However, in many real-world settings one seeks to adapt to multiple, but somewhat similar, target domains. Applying pairwise adaptation approaches to this setting may be suboptimal, as they fail to leverage shared information among multiple domains. In this section, we propose an information theoretic approach for domain adaptation in the novel context of multiple target domains with unlabeled instances and one source domain with labeled instances. Our model aims to find a shared latent space common to all domains, while simultaneously accounting for the remaining private, domain-specific factors. Disentanglement of shared and private information is accomplished using a unified information-theoretic approach, which also serves to establish a stronger link between the latent representations and the observed data. The resulting model, accompanied by an efficient optimization algorithm, allows simultaneous adaptation from a single source to multiple target domains.

Most works on **uDA** today focus on a single-source-single-target-domain scenario. However, in many real-world applications, unlabeled data may come from different domains, thus, with different statistical properties but with common task-related content. For instance, we may have access to images of the same class of objects (e.g., cars) recorded by various types of cameras, and/or under different camera views and at different times, rendering multiple different domains (e.g., datasets). Likewise, facial expressions of emotions, such as joy and surprise, shown by different people and recorded under different views, result in multiple domains with varying data distributions. In most cases, these domains have similar *underlying* data distributions. This, in turn, can be leveraged to build more effective and robust classifiers for tasks such as the object or emotion recognition across multiple datasets/domains.

To this end, most of the **uDA** methods focus on the single-source-single-target **DA** scenario. However, in the presence of multiple domains, as typically encountered in real-world settings, this pair-wise adaptation approach may be suboptimal as it fails to

leverage simultaneously the knowledge shared across multiple domains.

For instance, Zhao et al. [130] showed that by having access to multiple source domains can facilitate better adaptation to a single target domain, when compared to the pair-wise **DA** approach. It is intuitive that the access to multiple *labelled* source domains offers more adaptation flexibility for the target domain (i.e., by efficiently exploring the data labels across multiple source domains that are related to the target domain). Yet, it requires labels for the data from multiple source domains, which can be costly and time/labour-intensive to obtain. On the other hand, a simultaneous adaptation to *multiple* and *unlabelled* target domains may circumvent the need for manual labeling of multiple domains/datasets. This **DA** scenario is important as we usually have access to multiple unlabeled domains; yet, it is also more challenging due to the lack of supervision in the target domains. Still, multi-target **DA** offers advantages over a single-target **DA** when: (i) there is direct knowledge sharing between the source and multiple target domains, and (ii) the source and a target domain are related through another target domain. While this seems intuitive, it is critical how the data from multiple *unlabelled* target domains are leveraged within the multi-target **DA** approach, in order to improve its performance over the pair-wise **DA** and naive-fusion of multiple target domains.

To this end, we propose a Multi-Target DA-Information-Theoretic-Approach (**MTDA-ITA**) for a single-source-multi-target **DA**. We exploit a single source domain and focus on multiple target domains to investigate the effects of multi-target **DA**; however, the proposed approach can easily be extended to multiple source domains. This approach leverages the data from multiple target domains to improve performance compared to individually learning from pair-wise source-target domains. Specifically, we simultaneously factorize the information from each available target domain and learn separate subspaces for modeling the shared (i.e., correlated across the domains) and private (i.e., independent between the domains) subspaces of the data [98]. To this end, we employ deep learning to derive an information theoretic approach where we jointly maximize the mutual information between the domain labels and private (domain-specific) features, while minimizing the mutual information between the domain labels and the shared

(domain-invariant) features. Consequently, more robust feature representations are learned for each target domain by exploiting dependencies between multiple target domains.

3.4.1 Information Theory: Background

Let $x = (x_1, x_2, ..., x_d)$ denote a d-dimensional random variable with probability density function (**pdf**) given by $p(x)$. Shannon differential entropy [71] is defined in the usual way as $H(x) = -\mathbb{E}_x [\ln p(x)]$ where $\mathbb{E}$ denotes the expectation operator. Let $z = (z_1, z_2, ..., z_m)$ denote a m-dimensional random variable with pdf $p(z)$. Then mutual information between two random variables, x and z, is defined as $I(x; z) = H(x) + H(z) - H(x, z)$. Mutual information can also be viewed as the reduction in uncertainty about one variable given another variable—i.e., $I(x; z) = H(x) - H(x|z) = H(z) - H(z|x)$.

3.4.2 Multi-target Domain Adaptation

In this section, we describe our proposed information theoretic approach that supports domain adaptation for multiple target domains simultaneously, finding factorized latent spaces that are non-redundant, and that can capture explicitly the shared (domain invariant) and the private (domain dependent) features of the data well suited for better generalization for domain adaptation.

Let $(X, Y, D) = \{(x_i, y_i, d_i)\}_{i=0}^{N}$ be a collection of M domains (a labeled source domain, and $M - 1$ unlabeled target domains), where x_i denotes the i-th sample, and $y_i = [y_i^1, y_i^2, ..., y_i^C]$ and $d_i = [d_i^1, d_i^2, ..., d_i^M]$ are the C-D and M-D encoding of the class and domain labels for x_i, respectively. Note that the class labels are only available for the source samples. The latent space representation of the data point x is denoted as $z = [z_s, z_p]$, where z_s and z_p are the (latent) shared and private features of the data point x, respectively. By factorizing the joint distribution $p(x, y, d, z_s, z_p)$ as

$$p(x, y, d, z_s, z_p) = p(x)p(d)p(z_s|x)p(z_p|x)p(y|z_s), \tag{3.44}$$

we propose to maximize the following objective function:

$$\mathcal{L}(x, y, d) = \lambda_r I(x; z) + \lambda_c I(y; z_s) + \lambda_d \big(I(d; z_p) - I(d; z_s)\big), \tag{3.45}$$

where $p(x)$ and $p(d)$ denote the underlying (true) data distribution and domain label distribution, respectively, $I(x;y)$ denotes the Mutual Information between the random variables x and y. λ_r, λ_c and λ_d denote the hyper-parameters controlling the weights of the objective terms. The proposed objective function in Eq. 3.45 maximizes the three terms described below:

- $I(x;z)$: encourages the latent features (both shared and private) to preserve information about the data samples (that can be used to reconstruct x from z).
- $I(y;z_s)$: enables to correctly predict the true class label of the samples out of their common shared features.
- $I(d;z_p) - I(d;z_s)$: encourages the latent private features to preserve the information about the domain label and penalizes the latent shared features to be domain informative. This not only reduces the redundancy in the shared and private features, but also, penalizes the redundancy of different private spaces, while preserving the shared information.

An additional term could be used to minimize the mutual information between the shared (z_s) and private (z_p) features. However, computing the mutual information (even approximating it) is intractable due to the highly complex joint distribution $p(z_s, z_p)$. Since we want z_s and z_p features to encode different aspects of x, we enforce such constraint by jointly maximizing the term: $I(d;z_p) - I(d;z_s)$.

3.4.3 Optimization

The following lower bound for mutual information is derived using the non-negativity of KL-divergence [8]; i.e., $\Sigma_x p(x|z) \ln \frac{p(x|z)}{q(x|z)} \geq 0$ gives:

$$I(x;z) \geq H(x) + \mathbb{E}_{p(x,z)}[\ln q(x|z;\theta_r)] \tag{3.46}$$

$q(x|z;\theta_r)$ is any arbitrary distribution parameterized by θ_r. We need a variational distribution $q(x|z;\theta_r)$ because the posterior distribution $p(x|z) = p(z|x)p(x)/p(z)$ is intractable since the true data distribution $p(x)$ is assumed to be unknown. Similarly, we

can derive lower bounds for $I(d; z_p) \geq H(d) + \mathbb{E}_{p(d,z_p)}[\ln q(d|z_p; \theta_p)]$ and $I(d; z_s) \geq H(d) + \mathbb{E}_{p(d,z_s)}[\ln q(d|z_s; \theta_s)]$, where $q(d|z_p; \theta_p)$ is any arbitrary distribution parameterized by θ_p.[9] We further drive lower bound for $I(y; z_s)$ as $I(y; z_s) \geq H(y) + \mathbb{E}_{p(y,z_s)}[\ln q(y|z_s; \theta_c)]$, where $q(y|z_s; \theta_c)$ is a variational distribution parameterized by θ_c approximating $p(y|z_s)$.

Let next $G_s(x; \theta_s)$ be a function (shared encoder) parameterized by θ_s that maps a sample x to its corresponding *shared* feature z_s, and $G_p(x; \theta_p)$ be an analogous function (private encoder) which maps x to z_p, the feature that is *private* to each domain (Figure 3.15). We also define $R(z_s, z_p; \theta_r)$ (decoder) as a decoding function mapping the concatenation of the latent features z_s and z_p to a sample reconstruction $\hat{x}$, and $D(z; \theta_d)$ (domain classifier) as a decoding function mapping z_s and z_p to a M-dimensional vector: the predictions of the domain label $\hat{d}$. Finally, $h(z_s; \theta_c)$ is a task-specific function (label classifier) mapping z_s to a K-dimensional probability vector of the class label $\hat{y}$.

We represent $p(d), p(x), p(y)$ as the empirical distribution of a finite training set (e.g. $p(d) = \frac{1}{N}\sum_{i=1}^{N} \delta(d - d_i)$) as in the case of variational autoencoders (VAE) [3, 89], $p(z_s|x), p(z_p|x)$ as deterministic functions of x as $p(z_s|x) = \delta\big(z_s - G_s(x; \theta_s)\big)$ and $p(z_p|x) = \delta\big(z_p - G_p(x; \theta_p)\big)$, and the variational distributions $q(y|z_s)$, $q(x|z)$ and , $q(d|z)$ as

$$
\begin{aligned}
q(y|z_s) &= \text{SoftMax}(h(z_s; \theta_c)), \\
q(d|z) &= \text{SoftMax}(D(z; \theta_d)), \\
q(x|z; \rho) &\propto \exp(\|x - R(z; \theta_r)\|_1)
\end{aligned} \tag{3.47}
$$

where SoftMax$(\cdot)$ denotes the softmax or normalized exponential function [17], and $\|.\|_1$ denotes the L_1 norm. Then, the optimization task can be posed as a minimax saddle point problem, where we use adversarial training to maximize (3.45) w.r.t. the parameters $(\theta_s, \theta_p, \theta_c)$, and to minimize (3.45) w.r.t. the parameters (θ_r, θ_d), using Stochastic Gradient Descent (SGD).

---

[9] Note that, for simplicity, we shared the parameters θ_d between the approximate posterior distributions $q(d|z_s, \theta_p)$ and $q(d|z_p; \theta_s)$.

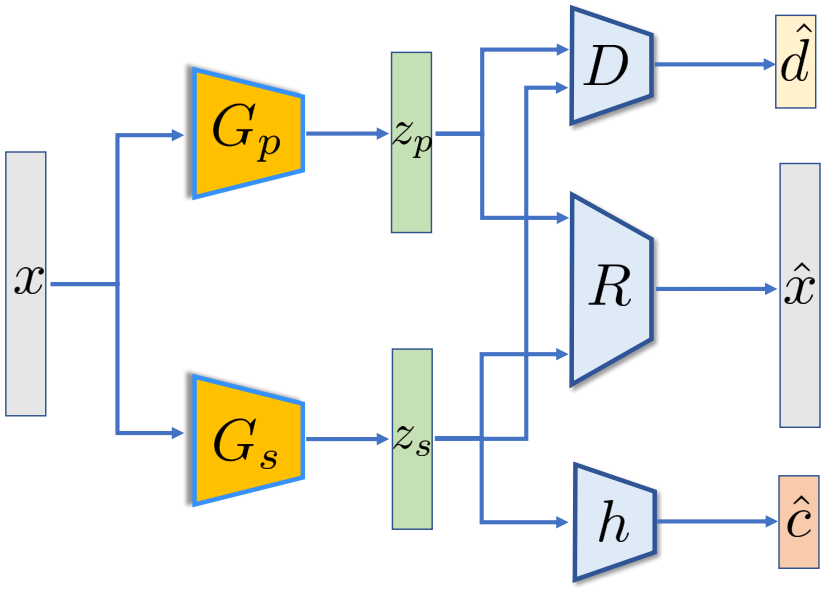


Figure 3.15: **MDTA-ITA**: The encoder $G_s(x)$ captures the feature representations (z_s) for a given input sample x that are shared among domains. $G_p(x)$ captures domain–specific private features (z_p) using the *shared* private encoder. The shared decoder $R(z_p, z_s)$ learns to reconstruct the input sample by using both the private and shared features. The domain classifier $D(z_s/z_p)$ learns to correctly predict the domain labels of the actual samples from both their shared and private features while the classifier $h(z_s)$ learns to correctly predict the class labels from the shared features.

Optimizing the parameters θ_r of the decoder F

$$\hat{\theta_r} = \arg\min_{\theta_r} \mathcal{L}_R = \frac{\lambda_r}{N} \sum_{i=1}^{N} \|x_i - R\big(G_s(x_i), G_p(x_i)\big)\|_1. \tag{3.48}$$

The decoder $G(z_s, z_p; \theta_r)$ is trained in such a way so as to minimize the difference between original input x and its decoding from corresponding shared and private features via the decoder F.

Optimizing the parameters θ_d of the domain classifier D

$$\hat{\theta_d} = \arg\min_{\theta_d} \mathcal{L}_D = -\frac{\lambda_d}{N} \sum_{i=1}^{N} d_i^\top \ln D\big(G_s(x_i)\big) - \frac{\lambda_d}{N} \sum_{i=1}^{N} d_i^\top \ln D\big(G_p(x_i)\big). \tag{3.49}$$

$D(z; \theta_d)$ can be considered as a classifier whose task is to distinguish between the shared/private features of the different domains. More precisely, the two terms in

Eq. 3.49 encourage D to correctly predict the domain labels from the shared and private features, respectively.

Optimizing the parameters θ_c of the label classifier h

$$\hat{\theta}_c = \arg\min_{\theta_c} \big\{ -H(y) - \mathbb{E}_{p(y,z_s)} \big[\ln q(y|z_s) \big] \big\}. \tag{3.50}$$

Since we have access to the source labels, $H(y)$ is a constant for source samples. we can approximate $H[y]$ for the target samples using the output of the classifier h, leading to the following optimization problem:

$$\begin{aligned}\hat{\theta}_c =& \arg\min_{\theta_c} \mathcal{L}_h = -\frac{1}{N} \sum_{i=1}^{N_s} y_i^T \ln h\big(G_s(x_i)\big) - \frac{\lambda_c}{N - N_s} \sum_{i=N_s+1}^{N} h\big(G_s(x_i)\big)^\top \ln h\big(G_s(x_i)\big) + \\ & \frac{\lambda_c}{N - N_s} \sum_{i=N_s+1}^{N} h\big(G_s(x_i)\big)^\top \ln \left(\frac{1}{N - N_s} \sum_{i=N_s+1}^{N} h\big(G_s(x_i)\big) \right),\end{aligned} \tag{3.51}$$

where N_s denotes the number of source samples. Intuitively, we enforce the classifier h to correctly predict the class labels of the source samples by the first term in Eq. 3.51. We use the second term to minimize the entropy of $q(y|z_s)$ for the target samples; effectively, reducing the effects of "confusing" labels of target samples, as given by $p(y|z_s)$ that leads to decision boundaries occur far away from target data-dense regions in the feature space. The intuition behind the last term is that by minimizing only the entropy (second term), we may arrive at a degenerate solution where every target point x_t is assigned to the same class. Hence, the last term encourages the classifier h to have balanced labeling for the target samples where it reaches its minimum, $\ln C$, when each class is selected with uniform probability.

Optimizing the parameter θ_s of the shared encoder G_s

$$\hat{\theta}_s = \arg\min_{\theta_s} \mathcal{L}_S = \frac{\lambda_r}{N}\sum_{i=1}^{N} \|x - R\big(G_s(x_i), G_p(x_i)\big)\|_1 - \frac{\lambda_c}{N}\sum_{i=1}^{N_s} y_i^T \ln C\big(G_s(x_i)\big) + \frac{\lambda_d}{N}\sum_{i=1}^{N} d_i^\top \ln D\big(G_s(x_i)\big) - \frac{\lambda_c}{N-N_s}\sum_{i=N_s+1}^{N} h\big(G_s(x_i)\big)^\top \ln h\big(G_s(x_i)\big) + \frac{\lambda_c}{N-N_s}\sum_{i=N_s+1}^{N} h\big(G_s(x_i)\big)^\top \ln\left(\frac{1}{N-N_s}\sum_{i=N_s+1}^{N} h\big(G_s(x_i)\big)\right). \quad (3.52)$$

The first term in Eq. 3.52 encourages the shared encoder G_s to preserve the recovery ability of the shared features. The second term is the source domain classification loss penalty that encourages G_s to produce discriminative features for the labeled source samples. The third term simulates the adversarial training by trying to fool the domain classifier D when predicting the domain labels d, given the shared features z_s. The effect of this is two-fold: (i) the rendered shared features are more distinct from the corresponding private features, (ii) the shared features of different domains are encouraged to be similar to each other. The last two terms encourage G_s to produce the shared features for target samples so that the classifier is confident on the unlabeled target data, driving the shared features away from the decision boundaries.

Optimizing the parameter θ_p of the private encoder G_p

$$\hat{\theta}_p = \arg\min_{\theta_p} \mathcal{L}_P = \frac{\lambda_r}{N}\sum_{i=1}^{N} \|x_i - R\big(G_s(x_i), G_p(x_i)\big)\|_1 - \frac{\lambda_d}{N}\sum_{i=1}^{N} d_i^\top D\big(G_p(x_i)\big). \quad (3.53)$$

The first term in Eq. 3.53 encourages the private encoder G_p to preserve the recovery ability of the private features. The second term enforces distinct private features be produced for each domain by penalizing the representation redundancy in different private spaces. This, in turn, encourages moving this common information from multiple domains to their shared space.

To train our model, we alternate between updating the shared encoder G_s, the private encoder G_p, the decoder F, the classifier h, and the domain classifier D using the SGD algorithm.

Connection to Multiple Domain Transfer Networks

Recent studies have shown remarkable success in multiple domain transfer (MDT) [23, 4, 59, 54] though not in the context of the image classification, rather in the context of image generation. choi et al. [23] proposed **StarGAN**, a generative adversarial network capable of learning mappings among multiple domains in the contest of image to image translation framework. The goal of **StarGAN** is to train a single generator G though this requires passing in a vector along with each input to the generator specifying the output domain desired, that learns mappings among multiple domains. To achieve this, G is trained to translate an input image $\mathbf{x}$ into an output image $\mathbf{x}'$ conditioned on the target domain label $\mathbf{d}$, $G(x, d) \rightarrow x'$. Similar to our domain classifier module D, they introduce an auxiliary classifier that allows a single discriminator to control multiple domains.

Anoosheh et al. [4] introduced **ComboGAN**, which decouples the domains and networks from each other. Similar to our encoder/decoder modules, **ComboGAN**'s generator networks contain encoder/decoders assigning each encoder and decoder to a domain. They combine the encoders and decoders of the trained model like building blocks, taking as input any domain and outputting any other. For example during inference, to transform an image $\mathbf{x}$ from an arbitrary domain $\mathbf{X}$ to $\mathbf{x}'$ from domain $\mathbf{X}'$, they simply perform $x' = G_{\mathbf{X}'\mathbf{X}}(\mathbf{x}) = Decoder_{X'}(Encoder_X(\mathbf{x}))$. The result of $Encoder_X(\mathbf{x})$ can even be cached when translating to other domains as not to repeat computation.

The main differences between the MDT methods and ours is that, unlike our method which does domain alignment in feature space, MDT methods adapt representations not in feature space but rather in raw pixel space; translating samples from one domain to the "style" of a other domains. This works well for limited domain shifts where the domains are similar in pixel-space, but can be too limiting for settings with larger

domain shifts that results in poor performance in significant structural change of the samples in different domains.

Connection to Domain Separation Networks

The method closest to our work is Domain Separation Networks (**DSN**) [15], which use the notion of auto-encoders to explicitly separate the feature representations private to each source/target domain from those that are shared between the domains. Although extending **DSN** to multiple domains might seem trivial, **DSN** requires an autoencoder per domain, making the model impractical in the case of more than a couple of domains.

The overall loss of **DSN** consists of a reconstruction loss for each domain modeled by a shared decoder, a similarity loss such as **MMD**, which encourages domain invariance modeled by a shared encoder, and a dissimilarity loss modeled by two private encoders: one for the source domain and one for the target domain. While one could attempt to generalize **DSN** to multiple target domains by having individual per-target domain private encoders, doing so would prove problematic when the number of target domains is large — each private encoder would require a large "private" dataset to learn the private parameters. Precisely, for multiple (M) target domains, we could train a **DSN** model with one shared encoder, $M + 1$ private encoder (one for each domain), and one shared decoder. This leads to $M + 3$ models to train that implies the number of models increases linearly with the number of domains, as does the required training time. Second, **DSN** uses an orthogonality constraint among the shared and the private representations which may not be strong enough to remove redundancy and enforce disentangling among different private spaces. Precisely, **DSN** defines the loss via a soft subspace orthogonality constraint between the private and shared representation of each domain. However, it does not enforce the private representation of different domains to be different that may result in redundancy of different private spaces.

In addition, **DSN** enforces separation of spaces using the notion of Euclidean orthogonality, e.g., $\|z_s - z_p\|^2$. In case of multiple target domains, this would result in learning of all pairs of private spaces independently. To address those deficiencies, we first explicitly couple different private encoders into a single private encoder model, E_{θ_p}

of Figure 3.15 , which allows us to generalize to an arbitrary number of target domains. To assure that the information among the private and shared spaces is not shared (i.e., "orthogonal"), we define an information-theoretic criteria enforced by a domain classifier,D_{θ_d} of Figure 3.15, which aims to segment the private space into clusters that correspond to individual target domains. By using D_{θ_d} within the adversarial framework, **MTDA-ITA** learns simultaneously the shared and private features from different domains (see Figure 3.9). Furthermore, our loss functions, Eqs. 3.51 & 3.52, contain terms that encourage 1) classifier determination (low entropy, second terms) to suppress prediction of uncertain labels and 2) balanced labeling (last term) to avoid degenerate solutions where all instances in target are assigned to a single class. We also show in Section 3.4.4 that our model performs better than the trivial extension of DSNs to the multi-domain case.

3.4.4 Experimental Results

We compare the proposed method with state-of-the-art methods on standard benchmark datasets: a digit classification task that includes 4 datasets: **MNIST** [65], **MNIST-M** [35], **SVHN** [84], **USPS** [117], and **Multi-PIE** dataset [52]. We evaluate the performance of all methods with classification accuracy metric. We repeated each experiment 5 times and report the average of the accuracy.

We used ADAM [62] for training; the learning rate was set to 0.0002 and momentum parameters to 0.5 and 0.999. We used batches of size 16 from each domain, and the input images were mean-centered/rescaled to $[-1, 1]$. The hyper-parameters are empirically set as $\lambda_r = 1.0, \lambda_c = 0.01, \lambda_d = 0.20$. All the used architectures replicate those of state-of-the-art methods. Specifically, we use the network structure similar to **UNIT** [72]. Precisely, our private/shared encoders consisted of three convolutional layers as the front-end and four basic residual blocks as the back-end. The decoder consisted of four basic residual blocks as the front-end and four transposed convolutional layers as the back-end. The discriminator and the classifier consisted of stacks of convolutional layers. We used ReLU for nonlinearity. Tanh function is used as the activation function of the last layer in the decoder F for scaling the output pixels to

$[-1, 1]$. The quantitative evaluation involves a comparison of the performance of our model to previous work and to **Source Only** and **1-NN** baselines that do not use any domain adaptation. For **Source Only** baseline, we train our model only on the unaltered source training data and evaluate on the target test data. We compare the proposed method **MTDA-ITA** with several related methods designed for pair-wise source-target adaptation: **CORAL** ([110]), **DANN** [34], **ADDA** [117], **DTN** [129], **UNIT** [72], **PixelDA** [14], and **DSN** [15]. We reported the results of two following baselines: (i) one is to combine all the target domains into a single one and train it using **MTDA-ITA**, which we denote as (**c-MTDA-ITA**). (ii) the other one is to train multiple **MTDA-ITA** separately, where each one corresponds to a source-target pair which we denote as (**s-MTDA-ITA**). For completeness, we reported the results of the competing methods by combining all the target domains into a single one (denoted by **c-DTN**, **c-ADDA**, and **c-DSN**) as well. We also extend **DSN** to multiple domains by (i) having one private encoder for all domains denoted by (**1p-DSN**), (ii) adding multiple private encoders to it denoted by (**mp-DSN**) and contrast them with our model.

Digits Datasets

We combine four popular digits datasets (**MNIST**, **MNIST-M**, **SVHN**, and **USPS**) to build the multi-target domain dataset. All images were uniformly rescaled to 32×32. We take each of **MNIST-M**, **SVHN**, **USPS**, and **MNIST** as source domain in turn, and the rest as targets. We use all labeled source images and all unlabeled target images, following the standard evaluation protocol for unsupervised domain adaptation [35, 77]. We show the accuracy of different methods in Tables 3.10 and 3.11. The results show that first of all **c-MTDA-ITA** has worse performance than **s-MTDA-ITA** and **MTDA-ITA**. We have similar observations for **ADDA**, **DTN**, and **DSN** that demonstrates a naive combination of different target datasets can sometimes even decrease the performance of the competing methods. Furthermore, **MTDA-ITA** outperforms the state-of-the-art methods in most of domain transformations. The higher performance of **MTDA-ITA** compared to other methods is mainly attributed to the

Table 3.10: Classification results on digit datasets. M: **MNIST**; MM: **MNIST-M**, S: **SVHN**, U: **USPS**. c-X: combining all target domains into a single one and train it using X. **s-MTDA-ITA**: training multiple **MTDA-ITA** where each one correspond to a source-target pair. **1p-DSN**: extended **DSN** with single private encoder.**mp-DSN**: extended **DSN** with multiple private encoder. Last column shows the average rank of each method over all adaptation pairs.*__UNIT__ trains with the extended **SVHN** ($>$ 500K images vs ours 72K). *__PixelDA__ uses ($\approx 1,000$) of labeled target domain data as a validation set for tuning the hyper-parameters.

method	S → M	S → MM	S → U	M → S	M → MM	M → U	Ave. ranking
Source Only	62.10 ± 0.60	40.43 ± 0.70	39.90 ± 0.60	30.29 ± 0.59	55.98 ± 0.48	78.30 ± 0.38	14.00
1-NN	35.86	18.21	29.31	28.01	12.58	41.22	15.00
CORAL [110]	63.10 ± 0.61	54.37 ± 0.53	50.15 ± 0.63	33.40 ± 0.74	57.70 ± 0.69	81.05 ± 0.80	11.33
DANN [35]	73.80 ± 0.49	61.05 ± 0.80	62.54 ± 0.91	35.50 ± 0.65	77.40 ± 0.73	81.60 ± 0.60	8.75
ADDA [117]	77.68 ± 0.92	64.23 ± 0.70	64.10 ± 0.79	30.04 ± 0.98	91.47 ± 1.0	90.51 ± 0.80	6.43
c-ADDA	80.10 ± 0.69	56.80 ± 0.79	64.80 ± 0.88	27.50 ± 0.86	83.30 ± 0.90	84.10 ± 0.98	8.95
DTN [129]	81.40 ± 0.42	63.70 ± 0.39	60.12 ± 0.52	40.40 ± 0.50	85.70 ± 0.39	85.80 ± 0.46	6.04
c-DTN	82.10 ± 0.62	59.30 ± 0.59	56.87 ± 0.65	38.32 ± 0.50	80.90 ± 0.80	79.31 ± 0.78	7.96
PixelDA [14]	–	–	–	–	**98.10***	**94.10***	–
UNIT ([72])	**90.6***	–	–	–	–	92.90	–
DSN [15]	82.70 ± 0.37	64.80 ± 0.40	65.30 ± 0.28	49.30 ± 0.30	83.20 ± 0.30	91.65 ± 0.40	2.85
c-DSN	83.10 ± 0.20	60.56 ± 0.36	60.35 ± 0.59	46.80 ± 0.45	80.49 ± 0.40	88.21 ± 0.38	4.84
1p-DSN	81.00 ± 0.47	58.22 ± 0.68	58.06 ± 0.48	45.11 ± 0.33	77.33 ± 0.52	85.16 ± 0.63	4.90
mp-DSN	83.40 ± 0.30	61.00 ± 0.50	58.10 ± 0.64	47.35 ± 0.40	79.30 ± 0.59	86.45 ± 0.71	5.33
s-MTDA-ITA	82.90 ± 0.13	63.10 ± 0.28	63.54 ± 0.30	49.60 ± 0.25	87.42 ± 0.19	89.21 ± 0.28	2.88
c-MTDA-ITA	79.20 ± 0.28	59.90 ± 0.30	63.70 ± 0.26	45.30 ± 0.30	82.12 ± 0.22	87.47 ± 0.25	4.25
MTDA-ITA	**87.70** ± 0.24	**68.30** ± 0.15	**70.03** ± 0.20	**56.01** ± 0.21	93.50 ± 0.18	**94.20** ± 0.20	**1.16**

joint adaptation of related domains where each domain could benefit of other related domains. Furthermore, from the results obtained, we see that it is beneficial to use information coming from unlabeled target data (see Eq. 3.51 for updating the classifier C) during the learning process, compared to when no data from target domain is used (See the ablation study section for more information). Indeed, using our scheme, we find a representation space in which embeds the knowledge from the target domain into the learned classifier. By contrast, the competing methods do not provide a principled way of sharing information across all domains, leading to overall lower performance. The results also verify the superiority of **MTDA-ITA** over both **mp-DSN**, and **1p-DSN**. This can be due to (i) having multiple private encoders increase the number of parameters that may lead to **mp-DSN** overfitting, (ii) superiority of the **MTDA-ITA**'s domain adversarial loss over the **DSN**'s **MMD** loss to separate the shared and private features, (iii) utilization of the unlabeled target data to regularize the classifier in **MTDA-ITA**.

space.5em

Table 3.11: Classification results on digit datasets. M: **MNIST**; MM: **MNIST-M**, S: **SVHN**, U: **USPS**. c-X: combining all target domains into a single one and train it using X. **s-MTDA-ITA**: training multiple **MTDA-ITA** where each one correspond to a source-target pair. **mp-DSN**: extended **DSN** with multiple private encoder. *__UNIT__ trains with the extended **SVHN** (> 500K images vs ours 72K). *__PixelDA__ uses ($\approx 1,000$) of labeled target domain data as a validation set for tuning the hyper-parameters.

method	MM → S	MM → M	MM → U	U → S	U → M	U → MM
Source Only	40.00 ± 0.61	84.46 ± 0.29	80.43 ± 0.50	23.41 ± 0.52	50.64 ± 0.37	41.45 ± 0.38
1-NN	21.45	82.13	36.90	15.34	38.45	18.54
CORAL [110]	40.20 ± 0.60	84.90 ± 0.70	87.54 ± 0.44	38.90 ± 0.96	85.01 ± 0.61	60.45 ± 0.70
DANN [35]	51.80 ± 0.91	61.05 ± 0.71	85.34 ± 0.64	35.50 ± 0.84	77.40 ± 0.64	61.60 ± 0.64
ADDA [117]	40.64 ± 0.86	92.82 ± 0.48	80.70 ± 0.48	41.23 ± 0.78	90.10 ± 0.58	56.21 ± 0.79
c-ADDA	35.43 ± 0.94	88.47 ± 0.61	74.19 ± 0.58	39.36 ± 0.99	84.67 ± 0.94	52.54 ± 0.88
DTN [129]	48.80 ± 0.66	88.80 ± 0.38	90.68 ± 0.35	42.43 ± 0.61	89.04 ± 0.36	55.78 ± 0.40
c-DTN	44.21 ± 0.61	83.60 ± 0.54	84.98 ± 0.41	39.75 ± 0.64	85.04 ± 0.45	48.86 ± 0.54
UNIT [72]	–	–	–		90.60	–
DSN [15]	51.50 ± 0.64	90.20 ± 0.31	89.95 ± 0.29	**48.20** ± 0.59	**91.40** ± 0.30	60.45 ± 0.35
c-DSN	47.10 ± 0.50	84.60 ± 0.40	84.80 ± 0.39	40.50 ± 0.61	86.05 ± 0.46	56.25 ± 0.50
1p-DSN	45.00 ± 0.60	81.96 ± 0.60	83.03 ± 0.49	39.30 ± 0.51	84.55 ± 0.56	55.03 ± 0.60
mp-DSN	47.15 ± 0.64	85.51 ± 0.54	83.24 ± 0.24	38.30 ± 0.74	87.40 ± 0.35	55.47 ± 0.44
s-MTDA-ITA	50.55 ± 0.18	94.82 ± 0.21	89.05 ± 0.28	40.13 ± 0.30	87.10 ± 0.25	61.01 ± 0.24
c-MTDA-ITA	47.32 ± 0.19	90.20 ± 0.30	90.01 ± 0.24	41.10 ± 0.35	85.35 ± 0.28	60.31 ± 0.34
MTDA-ITA	**55.50** ± 0.22	**98.20** ± 0.10	**94.10** ± 0.11	46.00 ± 0.48	**91.50** ± 0.23	**67.30** ± 0.15

Multi-PIE dataset

For this experiment, we use 5 different camera views (positions) $C05$, $C08$, $C09$, $C13$, and $C14$ as different domains and the face expressions (**normal, smile, surprise, squint, disgust, scream**) as labels. Each domain contains 27120 images of size $64 \times 64 \times 3$. We used each view as the source domain, in turn, and the rest as targets. We expect the face inclination angle to reflect the complexity of transfer learning. Tables 3.12, 3.13 and 3.14 show the classification accuracy of different methods. As can be seen, **MTDA-ITA** achieves the best performances as well as the best scores in most settings that verifies the effectiveness of **MTDA-ITA** for multi-target domain adaptation. Clearly, with the increasing camera angle, the image structure changes up to a certain extent (the views become heterogeneous). However, our method produces better results even under such very challenging conditions.

Table 3.12: Classification results on Multi-PIE dataset. Last column shows the average rank of each method over all adaptation pairs.

method	C13 → C05	C13 → C08	C13 → C09	C13 → C14	C14 → C05	C14 → C08	C14 → C09	C14 → C13	Ave. ranking
Source Only	50.79 ± 0.33	45.90 ± 0.50	40.04 ± 0.40	59.68 ± 0.29	60.03 ± 0.55	36.80 ± 0.61	40.11 ± 0.50	60.57 ± 0.36	16.08
1-NN	33.21	37.01	34.45	48.79	47.44	28.24	30.86	44.86	17.00
CORAL	54.89 ± 0.52	48.90 ± 0.48	40.30 ± 0.53	68.90 ± 0.35	59.98 ± 0.45	40.63 ± 0.55	40.80 ± 0.53	65.11 ± 0.45	11.95
DANN	57.86 ± 0.41	50.30 ± 0.43	45.30 ± 0.50	70.68 ± 0.35	57.20 ± 0.45	40.22 ± 0.55	40.77 ± 0.45	70.50 ± 0.55	9.92
ADDA	64.83 ± 0.69	63.20 ± 0.45	55.48 ± 0.65	74.25 ± 0.55	73.62 ± 0.75	43.56 ± 0.95	38.68 ± 0.95	72.84 ± 0.75	9.33
c-ADDA	59.20 ± 0.25	30.70 ± 0.63	53.20 ± 0.40	68.33 ± 0.35	65.88 ± 0.38	30.60 ± 0.61	45.34 ± 0.48	64.30 ± 0.40	11.50
DTN	63.78 ± 0.29	60.45 ± 0.35	60.55 ± 0.35	72.60 ± 0.25	70.67 ± 0.30	41.55 ± 0.65	41.45 ± 0.45	70.67 ± 0.45	8.75
c-DTN	57.53 ± 0.42	55.24 ± 0.45	57.14 ± 0.39	65.16 ± 0.35	63.80 ± 0.42	38.97 ± 0.71	39.80 ± 0.65	62.10 ± 0.45	10.92
PixelDA	45.68 ± 0.52	44.95 ± 0.42	44.45 ± 0.55	**90.50** ± 0.25	46.28 ± 0.60	45.89 ± 0.61	44.45 ± 0.51	69.15 ± 0.45	9.95
UNIT	44.14 ± 0.10	44.47 ± 0.11	44.21 ± 0.12	44.47 ± 0.11	43.03 ± 0.1	44.44 ± 0.15	44.47 ± 0.15	44.47 ± 0.05	11.07
DSN	64.15 ± 0.30	57.70 ± 0.38	49.15 ± 0.45	80.75 ± 0.27	82.20 ± 0.28	38.75 ± 0.53	45.00 ± 0.25	80.50 ± 0.35	5.15
c-DSN	57.34 ± 0.45	31.63 ± 0.60	51.17 ± 0.40	74.52 ± 0.37	82.01 ± 0.35	34.25 ± 0.58	42.63 ± 0.55	79.42 ± 0.35	8.20
1p-DSN	55.84 ± 0.50	30.03 ± 0.50	49.06 ± 0.38	72.11 ± 0.50	81.22 ± 0.45	33.33 ± 0.58	42.03 ± 0.24	78.78± 0.57	8.63
mp-DSN	55.20 ± 0.46	30.40 ± 0.50	47.80 ± 0.35	75.30 ± 0.25	80.75 ± 0.20	30.20 ± 0.55	43.00 ± 0.35	79.02 ± 0.40	8.88
s-MTDA-ITA	70.10 ± 0.27	58.90 ± 0.25	58.10 ± 0.27	80.12 ± 0.15	82.05 ± 0.18	45.90 ± 0.30	52.67 ± 0.30	81.60 ± 0.24	3.65
c-MTDA-ITA	60.34 ± 0.17	55.67 ± 0.21	57.10 ± 0.23	73.50 ± 0.20	76.80 ± 0.10	43.10 ± 0.12	48.10 ± 0.14	80.90 ± 0.11	5.01
MTDA-ITA	**78.40** ± 0.2	**66.70** ± 0.17	**70.30** ± 0.14	85.49 ± 0.11	**87.20** ± 0.10	**61.40** ± 0.14	**60.05** ± 0.13	**86.70** ± 0.10	**1.20**

Table 3.13: Classification results on Multi-PIE dataset.

method	C08 → C05	C08 → C09	C08 → C13	C08 → C14	C09 → C05	C09 → C08	C09 → C13	C09 → C14
Source Only	33.70 ± 0.33	50.10 ± 0.28	50.80 ± 0.32	40.13 ± 0.26	33.32 ± 0.44	48.24 ± 0.32	49.24 ± 0.30	36.19 ± 0.27
1-NN	28.75	35.39	39.79	32.13	26.82	35.30	34.26	28.41
CORAL [110]	35.89 ± 0.44	55.79 ± 0.50	60.00 ± 0.29	40.67 ± 0.48	35.89 ± 0.40	51.56 ± 0.43	50.45 ± 0.41	40.67 ± 0.35
DANN [35]	40.20 ± 0.50	56.89 ± 0.39	55.83 ± 0.40	43.25 ± 0.41	50.63 ± 0.38	**58.40** ± 0.51	55.81 ± 0.53	48.90 ± 0.43
ADDA [117]	37.40 ± 0.68	58.40 ± 0.73	60.40 ± 0.83	42.10 ± 0.48	29.40 ± 0.70	53.30 ± 0.49	45.30 ± 0.53	38.30 ± 0.63
c-ADDA	41.60 ± 0.64	39.65 ± 0.70	50.00 ± 0.52	46.25 ± 0.52	45.01 ± 0.63	52.14 ± 0.53	37.43 ± 0.60	43.26 ± 0.58
DTN [129]	44.13 ± 0.41	57.42 ± 0.42	55.89 ± 0.48	45.76 ± 0.39	44.53 ± 0.49	57.34 ± 0.35	52.43 ± 0.38	51.55 ± 0.40
c-DTN	45.10 ± 0.44	49.78 ± 0.50	47.43 ± 0.46	45.79 ± 0.48	49.80 ± 0.40	55.69 ± 0.35	50.10 ± 0.38	52.31 ± 0.29
PixelDA [14]	**46.45** ± 0.45	44.33 ± 0.38	44.87 ± 0.41	46.83 ± 0.29	45.63 ± 0.34	16.37 ± 0.27	45.43 ± 0.35	47.00 ± 0.49
UNIT [72]	43.88 ± 0.18	43.99 ± 0.23	44.47 ± 0.19	44.47 ± 0.24	44.47 ± 0.17	43.95 ± 0.21	44.64 ± 0.22	44.47 ± 0.19
DSN [15]	**46.25** ± 0.53	47.50 ± 0.60	**62.15** ± 0.58	39.72 ± 0.55	45.85 ± 0.48	56.65 ± 0.50	56.5 ± 0.38	42.87 ± 0.43
c-DSN	45.82 ± 0.53	44.64 ± 0.42	45.60 ± 0.48	46.32 ± 0.52	45.18 ± 0.47	45.52 ± 0.55	44.79 ± 0.53	47.37 ± 0.48
1p-DSN	44.12 ± 0.73	44.14 ± 0.20	45.00 ± 0.38	45.62 ± 0.42	44.78 ± 0.47	45.02 ± 0.65	44.21 ± 0.48	46.97 ± 0.38
mp-DSN	42.19 ± 0.46	44.70 ± 0.53	42.47 ± 0.48	40.50 ± 0.39	45.00 ± 0.51	43.80 ± 0.50	45.79 ± 0.48	42.39 ± 0.49
s-MTDA-ITA	44.77 ± 0.19	45.61 ± 0.18	60.00 ± 0.27	46.70 ± 0.28	49.06 ± 0.24	55.33 ± 0.22	59.90 ± 0.30	50.64 ± 0.26
c-MTDA-ITA	44.35 ± 0.27	42.67 ± 0.24	58.90 ± 0.26	44.32 ± 0.26	46.74 ± 0.22	54.11 ± 0.21	56.89 ± 0.23	49.64 ± 0.19
MTDA-ITA	**46.30** ± 0.25	**60.60** ± 0.18	60.50 ± 0.19	**50.40** ± 0.20	**55.59** ± 0.25	57.80 ± 0.21	**64.20** ± 0.18	**56.34** ± 0.20

Ablation Studies

We performed an ablation study on the proposed model measuring impact of various terms on the model's performance. To this end, we conducted additional experiments for the digit datasets with different components ablation, i.e., training without the reconstruction loss (denoted as **MTDA-woR**) by setting $\lambda_r = 0$, training without the classifier entropy loss (denoted as **MTDA-woE**) by setting $\lambda_c = 0$, training without the multi-domain separation loss (denoted as **MTDA-woD**) by setting $\lambda_d = 0$.

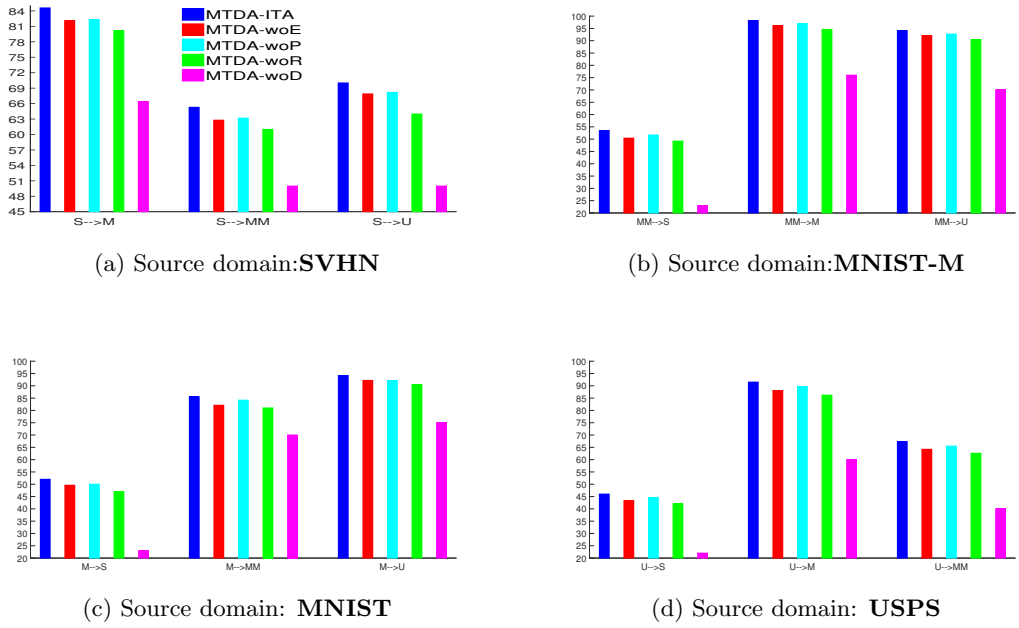
As can be seen from Figure 3.16, disabling each of the above components leads to degraded performance. More precisely, the average drop by disabling the classifier entropy loss is $\approx 3.5\%$. Similarly, by disabling the reconstruction loss and the multi-domain separation loss, we have $\approx 4.5\%$ and $\approx 22\%$ average drop in performance,

Table 3.14: Classification results on Multi-PIE dataset.

method	C05 → C08	C05 → C09	C05 → C13	C05 → C14
Source Only	31.56 ± 0.40	40.67 ± 0.36	39.89 ± 0.22	54.70 ± 0.25
1-NN	27.28	31.22	33.66	47.04
CORAL [110]	36.55 ± 0.66	38.60 ± 0.67	40.60 ± 0.58	55.29 ± 0.47
DANN [35]	40.30 ± 0.60	41.20 ± 0.65	40.12 ± 0.60	58.90 ± 0.38
ADDA [117]	33.21 ± 0.81	30.86 ± 0.90	52.44 ± 0.80	70.18 ± 0.60
c-ADDA	46.88 ± 0.65	36.38 ± 0.88	39.14 ± 0.85	65.41 ± 0.69
DTN [129]	38.50 ± 0.51	30.56 ± 0.46	55.78 ± 0.36	68.90 ± 0.31
c-DTN	41.70 ± 0.42	31.10 ± 0.48	50.19 ± 0.45	60.34 ± 0.35
PixelDA [14]	44.93 ± 0.42	44.75 ± 0.45	45.18 ± 0.45	46.88 ± 0.49
UNIT ([72])	44.47 ± 0.21	44.47 ± 0.21	44.47 ± 0.20	44.51 ± 0.28
DSN [15]	45.12 ± 0.46	44.35 ± 0.49	48.12 ± 0.53	75.00 ± 0.39
c-DSN	42.52 ± 0.48	38.54 ± 0.64	34.15 ± 0.64	69.45 ± 0.55
1p-DSN	41.64 ± 0.58	37.84 ± 0.63	34.65 ± 0.44	68.75 ± 0.85
mp-DSN	41.30 ± 0.28	35.14 ± 0.35	34.40 ± 0.35	65.70 ± 0.27
s-MTDA-ITA	44.40 ± 0.23	44.60 ± 0.25	47.65 ± 0.27	80.20 ± 0.13
c-MTDA-ITA	40.49 ± 0.25	40.70 ± 0.25	42.80 ± 0.25	71.60 ± 0.10
MTDA-ITA	**49.01** ± 0.20	**48.20** ± 0.27	**53.13** ± 0.22	**84.29** ± 0.10

respectively. Clearly, by disabling the multi-domain separation loss, the accuracy drops significantly due to the severe data distribution mismatch between different domains. The figure also demonstrates that leveraging the unlabeled data from multiple target domains during training enhances the generalization ability of the model that leads to higher performance. In addition, the performance drop caused by removing the reconstruction loss , i.e., without the private encoder/decoder, indicates (i) the benefit of modeling the latent features as the combination of shared and private features, (ii) the ability of the model's domain adversarial loss to effectively learn those features.

In order to examine the effect of the private features on the model's classification performance, we took the **MTDA-ITA** and trained it without the private encoder (denoted as **MTDA-woP**). As Figure 3.16 shows, without the private features, the model performed consistently worse ($\approx 2\%$ average drop in performance) in all scenarios. This demonstrates explicitly modeling what is unique to each domain can improve the model's ability to extract domain–invariant features. In summary, this ablation study showed that the individual components bring complimentary information to achieve the



(a) Source domain:**SVHN** (b) Source domain:**MNIST-M**

(c) Source domain: **MNIST** (d) Source domain: **USPS**

Figure 3.16: Ablation of **MTDA-ITA** on Digit dataset. We show that each component of our method, Reconstruction loss, Classifier entropy loss with separating shared/private features, contributes to the overall performance.

best classification results.

Analysis of shared/private space embedding

In the experiments conducted, we showed that our approach is able to achieve better performance than the competing methods including the extended **DSN** with one private encoder (**1p-DSN**) which is the most similar method to ours.

Indeed, Figure 3.17 depicts the embedding of the **MTDA-ITA** learned private/shared features, those of **1p-DSN** and the original features from different domains for Digit datasets (**SVHN** is the source). Notice that both **MTDA-ITA** and **1p-DSN** reduces the domain mismatch for the shared features (circle markers in Figure 3.17) and separate the shared features from private features. On the other hand, **MTDA-ITA** increases the domain separation for the private features (triangle markers, pure and well-separated domain clusters in (c)) while **1p-DSN** is unable to enforce the private representation of different domains to be different (e) that may result in redundancy of different private

spaces. This is partially due to the proposed multi-domain separation loss through the use of the domain classifier D, which penalizes the domain mismatch for the shared features and rewards the mismatch for the private features, something the **1p-DSN** fails to account for. Moreover, as supported by the quantitative results in Tables 3.10 and 3.11, the class label separation in the shared space for **1p-DSN** (f), is still evident but not as strong as in the **MTDA-ITA** (d). This can be attributed to the lack of redundancy in the private space that helps **MTDA-ITA** to learn more disentangled shared features and usage of the target samples during training, something the **1p-DSN** fails to account for.

3.4.5 Summary

In this section, we presented an information theoretic end-to-end approach to **uDA** in the context of a single source and multiple target domains that share a common task or properties. The proposed method learns feature representations invariant under multiple domain shifts and simultaneously discriminative for the learning task. This is accomplished by explicitly separating representations private to each domain and shared between source and target domains using a novel discrimination strategy. Our use of a single private domain encoder results in a highly scalable model, easily optimized using established back-propagation approaches. Results on three benchmark datasets for image classification show superiority of the proposed method compared to the state-of-the-art methods for unsupervised domain adaptation of visual domain categories.

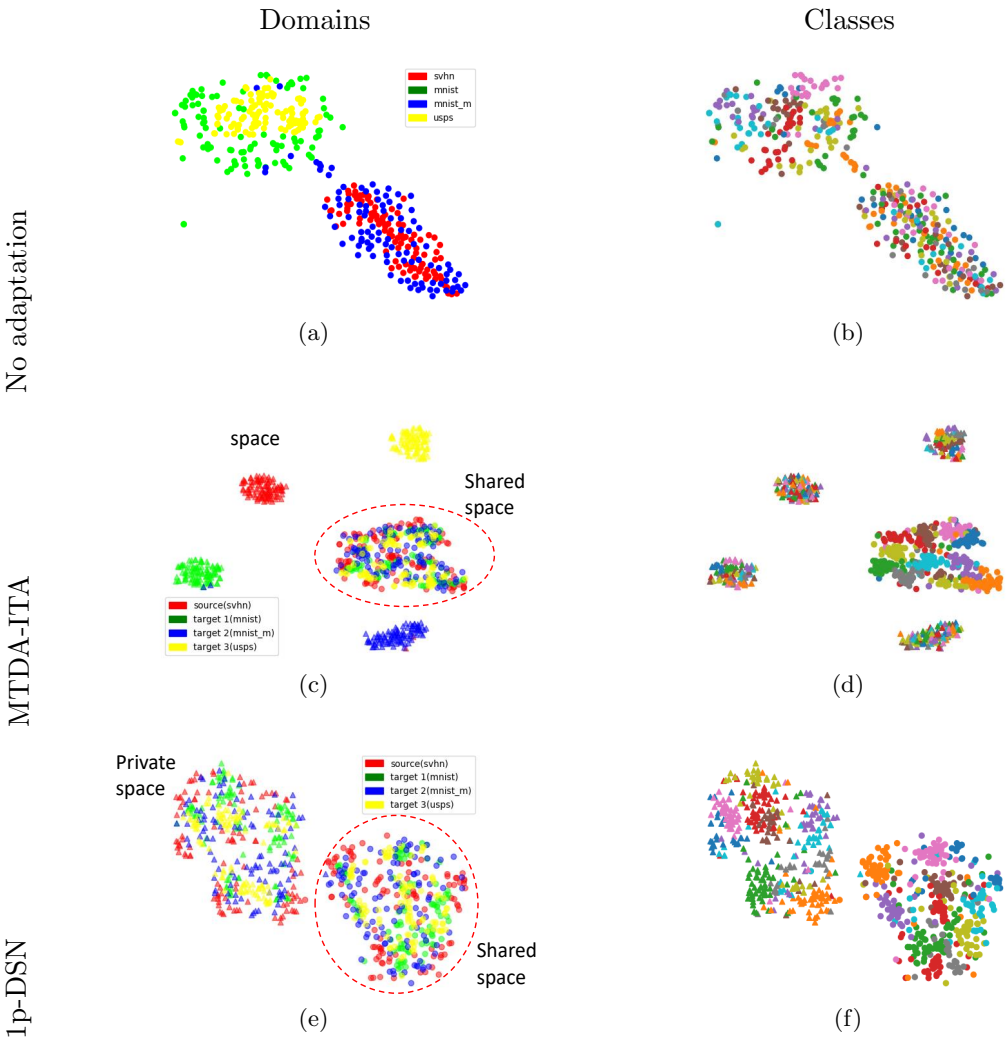


Figure 3.17: Feature visualization for embedding of digit datasets using t-SNE algorithm. The first and the second columns show the domains and classes, respectively, with color indicating domain and class membership. (a),(b) Original features. (c),(d) learned features for **MTDA-ITA** (triangle marker: private features, circle marker: shared features). Large clusters in the right column represent points from the shared space, while the smaller ones are from the private spaces. (e),(f) learned features for **1p-DSN**.

Chapter 4

Conclusions

This chapter summarizes the main results and contributions presented in this thesis, discusses the open issues and sketches possible future directions of research.

4.1 Summary

A quick overview of the current state-of-the-art image (object) classification methods shows us that all approaches based on a large amount of labeled training data perform reasonably well on difficult datasets [55]. However, most of them provide very few guarantees when only a small amount of labeled training data points is available, or more in general, if there is a mismatch between the training and the testing distribution [115].

The purpose of this thesis has been to understand how to best transfer what we know to a new domain, where data is scarce. We have shown that there are multiple ways of addressing this, depending on the availability of auxiliary data. More specifically, we tackled the problem as an unsupervised domain adaptation problem, in chapter 3, and proposed multiple solutions: we showed that we can reduce the domain gap by aligning the feature distributions of source (train) and target (test) domains using a Probabilistic MMD measure(PUnDA, 3.1) or by using adversarial methods (3.3, 3.4). In section 3.2 we showed a complex approach using Gaussian process that, by utilizing max-margin principle, reaches even better performance.

In conclusion, our work demonstrated that, by properly exploiting auxiliary knowledge, we can train effective shallow/deep models even when annotated data is scarce. For this purpose we presented a number of alternative solutions, ranging from classical domain adaptation approaches to innovative, across domains, transfer learning techniques, each suitable for different learning conditions.

4.2 Future Research

All the proposed methods for domain adaptation have been presented together with an analysis of their properties, and evaluating their performances. We briefly describe in the following a few aspects of this work relevant for future work.

All of our proposed methods are designed to work in the unsupervised domain adaptation setting, with the assumption that no labels are available for the target domain. Clearly this hypothesis does not always hold and it is likely that, if (a few) target labels are available, a better alignment between domains can be achieved. A potential future work would be to investigate how to modify our models to best adapt to this semi-supervised setting.

Likewise, our DA methods could be adapted to work on the recent problem of *open set* [86] domain adaptation, a more realistic scenario where only a few categories of interest are shared between source and target.

Our models in this thesis has been evaluated on the task of image classification (object recognition). There are no specific requirements in the algorithm which limit its applicability to this setting, and it would be very interesting to see how much we can transfer to more complicated task (e.g., object detection or semantic segmentation). In fact, such domain adaptation tasks are quite challenging as there usually exists a significant gap between source and target domains [96, 132].

Bibliography

[1] M. Abavisani and V. Patel. Domain adaptive subspace clustering. In *British Machine Vision Conference*. BMVA, 2016.

[2] M. Abavisani and V. M. Patel. Adversarial domain adaptive subspace clustering. In *IEEE International Conference on Identity, Security, and Behavior Analysis*, 2018.

[3] M. E. Abbasnejad, A. Dick, and A. van den Hengel. Infinite variational autoencoder for semi-supervised learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 781–790. IEEE, 2017.

[4] A. Anoosheh, E. Agustsson, R. Timofte, and L. Van Gool. Combogan: Unrestrained scalability for image domain translation. *ICLR Workshops*, 2017.

[5] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *CoRR*, abs/1701.04862, 2017.

[6] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *IEEE International Conference on Computer Vision (ICCV)*, pages 769–776. IEEE, 2013.

[7] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Domain adaptation on the statistical manifold. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2481–2488, 2014.

[8] D. Barber and F. Agakov. The im algorithm: a variational approach to information maximization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 201–208. MIT Press, 2003.

[9] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision (ECCV)*, pages 404–417, 2006.

[10] 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(1–2):151–175, 2010.

[11] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation, 2007. In Advances in Neural Information Processing Systems.

[12] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[13] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

[14] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 7, 2017.

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

[16] S. Branson, G. Van Horn, C. Wah, P. Perona, and S. Belongie. The ignorant led by the blind: A hybrid human–machine vision system for fine-grained categorization. *International Journal of Computer Vision*, 108(1-2):3–29, 2014.

[17] J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer, 1990.

[18] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò. Just dial: Domain alignment layers for unsupervised domain adaptation. In *International Conference on Image Analysis and Processing*, pages 357–369. Springer, 2017.

[19] W.-L. Chao, B. Gong, K. Grauman, and F. Sha. Large-Margin Determinantal Point Processes, 2015. Uncertainty in AI.

[20] C. Chen, W. Xie, W. Huang, Y. Rong, X. Ding, Y. Huang, T. Xu, and J. Huang. Progressive feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 627–636, 2019.

[21] J. Chen, E. Gan, K. Rong, S. Suri, and P. Bailis. Crosstrainer: Practical domain adaptation with loss reweighting. *arXiv preprint arXiv:1905.02304*, 2019.

[22] E. Choi and C. Lee. Feature extraction based on the bhattacharyya distance. *Pattern Recognition*, 36(8):1703–1709, 2003.

[23] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1711, 2017.

[24] S. Cicek and S. Soatto. Unsupervised domain adaptation via regularized conditional alignment. *International Conference on Computer Vision (ICCV)*, 2019.

[25] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pages 6510–6520, 2017.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[27] Z. Deng, Y. Luo, and J. Zhu. Cluster alignment with a teacher for unsupervised domain adaptation. *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[28] K. G. Derpanis. The bhattacharyya measure. *Mendeley Computer*, 1(4):1990–1992, 2008.

[29] A. Dezfouli and E. V. Bonilla. Scalable inference for Gaussian process models with black-box likelihoods, 2015. In Advances in Neural Information Processing Systems.

[30] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[31] S. R. Fanello, C. Ciliberto, L. Natale, and G. Metta. Weakly supervised strategies for natural object recognition in robotics. In *2013 IEEE International Conference on Robotics and Automation*, pages 4223–4229. IEEE, 2013.

[32] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 524–531, 2005.

[33] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2960–2967, 2013.

[34] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. *International Conference on Machine Learning (ICML)*, 2015.

[35] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[36] 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.

[37] Z. Ghahramani, M. J. Beal, et al. Variational inference for bayesian mixtures of factor analysers. In *Advances in Neural Information Processing Systems (NIPS)*, volume 12, pages 449–455, 1999.

[38] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *Euroupean Conference on Computer Vision (ECCV)*, pages 597–613, 2016.

[39] B. Gholami, V. Pavlovic, et al. Punda: Probabilistic unsupervised domain adaptation for knowledge transfer across visual categories. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3581–3590, 2017.

[40] B. Gholami, P. Sahu, M. Kim, and V. Pavlovic. Task-discriminative domain alignment for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision(ICCV)*, 2019.

[41] B. Gholami, P. Sahu, O. Rudovic, K. Bousmalis, and V. Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *IEEE Transaction on Image processing (TIP)*, 2019.

[42] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, pages 222–230, 2013.

[43] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073, 2012.

[44] 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.

[45] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets, 2014. In Advances in Neural Information Processing Systems.

[46] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *IEEE International Conference on Computer Vision (ICCV)*, pages 999–1006, 2011.

[47] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization, 2004. In Proc. of Advances in Neural Information Processing Systems.

[48] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.

[49] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520, 2007.

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

[51] T. L. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *Advances in Neural Information Processing Systems (NIPS)*, volume 18, pages 475–482, 2005.

[52] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.

[53] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2765–2773, 2017.

[54] G.-Y. Hao, H.-X. Yu, and W.-S. Zheng. Mixgan: Learning concepts from different domains for mixture generation. *arXiv preprint arXiv:1807.01659*, 2018.

[55] 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.

[56] S. Herath, M. Harandi, and F. Porikli. Learning an invariant hilbert space for domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[57] L. Hu, M. Kan, S. Shan, and X. Chen. Duplex generative adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1498–1507, 2018.

[58] W. Huang, D. Zhao, F. Sun, H. Liu, and E. Chang. Scalable Gaussian process regression using deep neural networks, 2015. Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI).

[59] H. Kameoka, T. Kaneko, K. Tanaka, and N. Hojo. Stargan-vc: Non-parallel many-to-many voice conversion with star generative adversarial networks. *arXiv preprint arXiv:1806.02169*, 2018.

[60] M. Kan, S. Shan, and X. Chen. Bi-shifting auto-encoder for unsupervised domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3846–3854, 2015.

[61] M. Kim, P. Sahu, B. Gholami, and V. Pavlovic. Unsupervised visual domain adaptation: A deep max-margin gaussian process approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4380–4390, 2019.

[62] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representation (ICLR)*, 2015.

[63] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[64] D. P. Kingma and M. Welling. Auto-encoding variational Bayes, 2014. In Proceedings of the Second International Conference on Learning Representations, ICLR.

[65] 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.

[66] S. Lee, D. Kim, N. Kim, and S.-G. Jeong. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 91–100, 2019.

[67] C. Li, A. Reiter, and G. D. Hager. Beyond spatial pooling: fine-grained representation learning in multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4913–4922, 2015.

[68] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales. Deeper, broader and artier domain generalization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 5543–5551. IEEE, 2017.

[69] S. Li, S. Song, and G. Huang. Prediction reweighting for domain adaptation. *IEEE transactions on neural networks and learning systems*, 28(7):1682–1695, 2016.

[70] Y. Li, K. Swersky, and R. Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727, 2015.

[71] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

[72] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 700–708, 2017.

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

[74] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 469–477. Curran Associates, Inc., 2016.

[75] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *International Conference on Machine Learning (ICML)*, 2015.

[76] M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu. Transfer sparse coding for robust image representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 407–414, 2013.

[77] M. Long, J. Wang, Y. Cao, J. Sun, and S. Y. Philip. Deep learning of transferable representation for scalable domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2027–2040, 2016.

[78] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu. Transfer joint matching for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1410–1417, 2014.

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

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

[81] M. Mancini, L. Porzi, S. Rota Bulò, B. Caputo, and E. Ricci. Boosting domain adaptation by discovering latent domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3771–3780, 2018.

[82] T. Ming Harry Hsu, W. Yu Chen, C.-A. Hou, Y.-H. Hubert Tsai, Y.-R. Yeh, and Y.-C. Frank Wang. Unsupervised domain adaptation with imbalanced cross-domain data. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4121–4129, 2015.

[83] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin. Evaluation of traffic sign recognition methods trained on synthetically generated data. In *International*

Conference on Advanced Concepts for Intelligent Vision Systems, pages 576–583. Springer, 2013.

[84] 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.

[85] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[86] P. Panareda Busto and J. Gall. Open set domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 754–763, 2017.

[87] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.

[88] P. O. Pinheiro. Unsupervised domain adaptation with similarity learning. *CVPR*, 2018.

[89] Y. Pu, W. Wang, R. Henao, L. Chen, Z. Gan, C. Li, and L. Carin. Adversarial symmetric variational autoencoder. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4330–4339, 2017.

[90] J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

[91] A. Rahimi and B. Recht. Random features for large-scale kernel machines, 2008. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), Advances in Neural Information Processing Systems 20.

[92] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[93] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[94] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision (ECCV)*, pages 213–226, 2010.

[95] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. *International Conference on Machine Learning (ICML)*, 2017.

[96] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018.

[97] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. *Computer Vision and Pattern Recognition*, 2018.

[98] M. Salzmann, C. H. Ek, R. Urtasun, and T. Darrell. Factorized orthogonal latent spaces. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 701–708, 2010.

[99] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2018.

[100] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. *CVPR*, 2018.

[101] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2110–2118, 2016.

[102] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304. Ieee, 2011.

[103] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.

[104] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[105] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs, 2006. In Advances in Neural Information Processing Systems.

[106] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11(Apr):1517–1561, 2010.

[107] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1453–1460. IEEE, 2011.

[108] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1433–1440, 2008.

[109] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[110] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision (ECCV)*, pages 443–450. Springer, 2016.

[111] B. Taskar, C. Guestrin, and D. Koller. Max-Margin Markov Networks, 2003. In Advances in Neural Information Processing Systems.

[112] R. Thibaux and M. I. Jordan. Hierarchical beta processes and the indian buffet process. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 2, pages 564–571, 2007.

[113] N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE, 2015.

[114] M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes, 2009. In Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics.

[115] A. Torralba, A. A. Efros, et al. Unbiased look at dataset bias. In *CVPR*, volume 1, page 7. Citeseer, 2011.

[116] Y.-H. H. Tsai, C.-A. Hou, W.-Y. Chen, Y.-R. Yeh, and Y.-C. F. Wang. Domain-constraint transfer coding for imbalanced unsupervised domain adaptation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[117] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2017.

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

[119] S. Uguroglu and J. Carbonell. Feature selection for transfer learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 430–442, 2011.

[120] L. Van der Maaten. A new benchmark dataset for handwritten character recognition. *Tilburg University*, pages 2–5, 2009.

[121] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[122] X. Wang, B. Wang, X. Bai, W. Liu, and Z. Tu. Max-margin multiple-instance dictionary learning. In *International Conference on Machine Learning*, pages 846–854, 2013.

[123] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning, 2016. AI and Statistics (AISTATS).

[124] R. Xu, G. Li, J. Yang, and L. Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1426–1435, 2019.

[125] 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. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2272–2281, 2017.

[126] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon. Pixel-level domain transfer. In *European Conference on Computer Vision (ECCV)*, pages 517–532, 2016.

[127] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811*, 2017.

[128] W. Zhang, W. Ouyang, W. Li, and D. Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3801–3809, 2018.

[129] X. Zhang, F. X. Yu, S.-F. Chang, and S. Wang. Deep transfer network: Unsupervised domain adaptation. *arXiv preprint arXiv:1503.00591*, 2015.

[130] H. Zhao, S. Zhang, G. Wu, J. P. Costeira, J. Moura, and G. J. Gordon. Multiple source domain adaptation with adversarial training of neural networks. *International Conference on Learning Representation (ICLR) workshops*, 05 2017.

[131] X. Zhu and A. B. Goldberg. *Introduction to semi-supervised learning*. Morgan & Claypool, 2009.

[132] X. Zhu, J. Pang, C. Yang, J. Shi, and D. Lin. Adapting object detectors via selective cross-domain alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 687–696, 2019.