# Unsupervised Visual Domain Adaptation: A Deep Max-Margin Gaussian Process Approach

Minyoung Kim[1,2,3], Pritish Sahu[1], Behnam Gholami[1], and Vladimir Pavlovic[1,3]

[1]Dept. of Computer Science, Rutgers University, NJ, USA
[2]Dept. of Electronic Engineering, Seoul National University of Science & Technology, South Korea
[3]Samsung AI Center, Cambridge, UK
mikim21@gmail.com, {ps851,bb510,vladimir}@cs.rutgers.edu, v.pavlovic@samsung.com

## Abstract

*For unsupervised domain adaptation, 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. Very recently it has been shown that it is not only critical to match the marginal input distributions, but also align the output class distributions. The latter can be achieved by minimizing the maximum discrepancy of predictors. In this paper, we take this principle further by proposing a more systematic and effective way to achieve hypothesis consistency using Gaussian processes (**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, significantly easier to solve than previous approaches based on adversarial minimax optimization. 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 domainEmpirical results demonstrate that our approach leads to state-to-the-art performance superior to existing methods on several challenging benchmarks for domain adaptation.*

## 1. Introduction

The success of deep visual learning largely relies on the abundance of data annotated with ground-truth labels where the main assumption is that the training and test data follow from the same underlying distribution. However, in real-world problems this presumption rarely holds due to artifacts such as the different types of noise or sensors, changes in object view or context, resulting in degradation of performance during inference on test data. One way to address this problem would be to collect labeled data in the test domain and learn a test-specific classifier while possibly leveraging the model estimated from the training data. Nevertheless, this would typically be a highly costly effort.

Domain adaptation, a formalism to circumvent the aforementioned problem, is the task of adapting a model trained in one domain, called the *source*, to another *target* domain, where the source domain data is typically fully labeled but we only have access to images from the target domain with no (or very few) labels. Although there are several slightly different setups for the problem, in this paper we focus on the *unsupervised domain adaptation* (**UDA**) with *classification* of instances as the ultimate objective. That is, given the fully labeled data from the source domain and unlabeled data from the target, the goal is to learn a classifier that performs well on the target domain itself.

One mainstream direction to tackle **UDA** is the shared space embedding process. 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 [3], 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 [21, 19, 14, 1, 16, 35, 15, 28, 38, 18] leverage this idea (see Sec. 4 for more details). 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 [56], can be reduced without using the target domain labels.

To address this issue, as effective solution was proposed in [51], which aims to take into account the class-specific decision boundary. Its motivation follows the theorem in [2] relating the target domain error to the maximal disagreement between any two classifiers, tighter than the former bound in [3]. It implies that a provably small target error is achievable by minimizing the maximum classifier discrepancy (**MCD**). The approach in [51], the **MCD** Algorithm

Pritish Sahu and Behnam Gholami contributed equally to this work.

(**MCDA** for short), attempted to minimize **MCD** directly using adversarial learning similar to GAN training [20], i.e., through solving a minimax problem that finds the pair of most discrepant classifiers and reduces their disagreement.

In this paper 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**) [47] 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 [63], a significantly easier-to-solve[1] task than the minimax optimization of [51] 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 [47], we utilize recent deep kernel techniques [27, 64] 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. Our extensive experimental results on several standard benchmarks demonstrate that the proposed approach achieves state-of-the-art prediction performance, outpacing recent **UDA** methods including **MCDA** [51].

## 2. Problem Setup and Preliminaries

We begin with the formal description of the **UDA** task for a multi-class classification problem.

**Unsupervised domain adaptation:** *Consider the joint space of inputs and class labels, $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{Y} = \{1, \dots, K\}$ for ($K$-way) classification. Suppose we have two domains on this joint space, **source (S)** and **target (T)**, defined by unknown distributions $p_S(\mathbf{x}, y)$ and $p_T(\mathbf{x}, y)$, respectively. We are 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 assume the shared set of class labels between the two domains. The goal is to assign the correct class labels $\{y_i^T\}$ to target data points $\mathcal{D}_T$.*

To tackle the problem in the shared latent space framework, we seek to learn the embedding function $\mathbf{G} : \mathcal{X} \to \mathcal{Z}$ and a classifier $h : \mathcal{Z} \to \mathcal{Y}$ in the *shared latent space* $\mathcal{Z}$. The embedding function $\mathbf{G}(\cdot)$ and the classifier $h(\cdot)$ are shared across both domains and will be applied to classify samples in the target domain using the composition $y = h(\mathbf{z}) = h(\mathbf{G}(\mathbf{x}))$.

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

$$(h^*, \mathbf{G}^*) = \arg\min_{h,\mathbf{G}} e_T(h, \mathbf{G}) \tag{1}$$

$$= \arg\min_{h,\mathbf{G}} \mathbb{E}_{(\mathbf{x},y)\sim p_T(\mathbf{x},y)}[I(h(\mathbf{G}(\mathbf{x})) \neq y)], \tag{2}$$

with $I(\cdot)$ the $1/0$ indicator function. Optimizing $e_T$ directly is typically infeasible. Instead, one can exploit the upper bounds proposed in [2] and [3], which we restate, without loss of generality, for the case of fixed $\mathbf{G}$.

**Theorem 1** *[2, 3] 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]:*

$$e_T(h) \leq e_S(h) + \sup_{h,h' \in \mathcal{H}} \left| d_S(h, h') - d_T(h, h') \right| + e^* \tag{3}$$

$$\leq e_S(h) + \sup_{h \in \mathcal{H}} \left| d_S(h, +1) - d_T(h, +1) \right| + e^* \tag{4}$$

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 $d_S(h, h') := \mathbb{E}_{\mathbf{z}\sim S}[\mathbb{I}(h(\mathbf{z}) \neq h'(\mathbf{z}))]$ denotes the discrepancy between two classifiers $h$ and $h'$ on the source domain $S$, and similarly for $d_T(h, h')$. We use $\mathbf{z} \sim S$ to denote the distribution of $\mathbf{z}$ in the latent space induced by $\mathbf{G}$ and $p_S(\mathbf{x}, y)$.

**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 (4), the supremum term is, up to a constant, equivalent to $\sup_{h \in \mathcal{H}} \mathbb{E}_{\mathbf{z}\sim S}[I(h(\mathbf{z}) = +1)] + \mathbb{E}_{\mathbf{z}\sim T}[I(h(\mathbf{z}) = -1)]$, 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 $\mathbf{G}$ 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 [61, 17, 7, 60], and typically accomplished through adversarial learning [20] or the maximum mean discrepancy [23].

**Tighter bound.** Recently, [51] exploited the tighter bound (3) 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 (3), interpreted as the *Maximum Classifier Discrepancy* (***MCD***), reduces to:

$$\sup_{h,h' \in \mathcal{H}} \mathbb{E}_{(\mathbf{x},y)\sim p_T(\mathbf{x},y)}[\mathbb{I}(h(\mathbf{z}) \neq h'(\mathbf{z}))]. \tag{5}$$

Named **MCDA**, [51] aims to minimize (5) directly via adversarial-cooperative learning of two deep classifier networks $h(\mathbf{z})$ and $h'(\mathbf{z})$. For the source domain data, these

[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.

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

two classifiers and $\mathbf{G}$ 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 $\mathbf{G}$ seeks to minimize the discrepancy[3].

## 3. Our Approach

**Overview.** We adopt the **MCD** principle, but propose a more 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 [47] 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 $\mathbf{G}$ 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 (5). Our approach is denoted by **GPDA**, and its details are described below.

### 3.1. GP-endowed Maximum Separation Model

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

$$P(\mathbf{f}) = \prod_{j=1}^{K} P(f_j), \quad f_j \sim \mathcal{GP}\big(0, k_j(\cdot,\cdot)\big), \tag{6}$$

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

$$\text{class}(\mathbf{z}) = \arg \max_{1 \le j \le K} f_j(\mathbf{z}). \tag{7}$$

We use the softmax likelihood model,

$$P(y = j|\mathbf{f}(\mathbf{z})) = \frac{e^{f_j(\mathbf{z})}}{\sum_{r=1}^{K} e^{f_r(\mathbf{z})}}, \quad \text{for } j = 1, \dots, K. \tag{8}$$

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

$$p(\mathbf{f}|\mathcal{D}_S) \propto p(\mathbf{f}) \cdot \prod_{i=1}^{N_S} P(y_i^S|\mathbf{f}(\mathbf{z}_i^S)), \tag{9}$$

where $\mathbf{z}_i^S = \mathbf{G}(\mathbf{x}_i^S)$. The key idea is to use (9) 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 $S$ 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 (8), exact posterior inference is intractable, and one has to resort to approximate inference. We will discuss an approach for efficient variational approximate inference in Sec. 3.2. 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 $\mathbf{z} = \mathbf{G}(\mathbf{x})$, $\mathbf{x} \sim T$, the latent function values $\mathbf{f}(\mathbf{z})$ sampled from the posterior (9) should lead to the class prediction (made by (7)) that is as consistent as possible across the samples.

This is illustrated in Fig. 1. Consider two different $\mathcal{H}$ priors $p_A$ and $p_B$ at a point $\mathbf{z}$, $p_A(\mathbf{f}(\mathbf{z}))$ and $p_B(\mathbf{f}(\mathbf{z}))$, where for brevity we drop the conditioning on $\mathcal{D}_S$ in notation. The class cardinality is $K = 3$. For simplicity, we assume that the latent functions $f_j$'s are independent from each other. Fig. 1 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[4]. For any target domain point $\mathbf{z} \sim T$ and each $j = 1, \dots, K$ let the mean and the variance of the $\mathcal{H}$ prior in (9) be:

$$\mu_j(\mathbf{z}) := \int f_j(\mathbf{z})\, p\big(f_j(\mathbf{z})|\mathcal{D}_S, \mathbf{z}\big)\, df_j(\mathbf{z}), \tag{10}$$

$$\sigma_j^2(\mathbf{z}) := \int (f_j(\mathbf{z}) - \mu_j(\mathbf{z}))^2\, p\big(f_j(\mathbf{z})|\mathcal{D}_S, \mathbf{z}\big)\, df_j(\mathbf{z}). \tag{11}$$

The maximum-a-posterior (MAP) class prediction by the model is denoted by $j^* = \arg\max_{1 \le j \le K} \mu_j(\mathbf{z})$. 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 $f_j(\mathbf{z})$, $j$ other than $j^*$, cannot overtake $\mu_{j^*}(\mathbf{z})$. That is, we seek

$$\mu_{j^*}(\mathbf{z}) - \alpha\sigma_{j^*}(\mathbf{z}) \ge \max_{j \ne j^*} \big(\mu_j(\mathbf{z}) + \alpha\sigma_j(\mathbf{z})\big), \tag{12}$$

where $\alpha$ 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[5], we impose slightly stricter constraint than (12), lead-

[3] See the Supplementary Material for further technical details.

[4] This choice conforms to the variational density family we choose in Sec. 3.2.

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

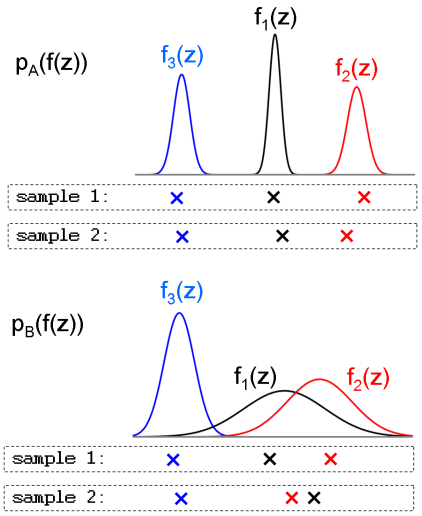



**Fig. 1:** Illustration of ideal ($p_A$) and problematic ($p_B$) posteriors at some fixed point $\mathbf{z}$ in the target domain. For each posterior, we also depict two plausible samples (marked as crosses). In $p_A$, most samples $\mathbf{f}(\mathbf{z})$, including the two shown, are consistent in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

ing to the final constraint:

$$\max_{1\le j\le K} \mu_j(\mathbf{z}) \ge 1 + \max_{j\neq j^*} \mu_j(\mathbf{z}) + \alpha \max_{1\le j\le K} \sigma_j(\mathbf{z}) - \xi(\mathbf{z}). \tag{13}$$

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

Our objective now is to find such embedding $\mathbf{G}$, **GP** kernel parameters $k$, $k(\mathbf{z},\mathbf{z}') = \boldsymbol{\phi}(\mathbf{z})^\top \boldsymbol{\phi}(\mathbf{z}')$, and minimal slack $\xi$, to impose (13). Equivalently, we pose it as the following optimization problem, for each $\mathbf{z} \sim T$:

$$\min_{\mathbf{G},k} \left( \max_{j\neq j^*} \mu_j(\mathbf{z}) - \max_{1\le j\le K} \mu_j(\mathbf{z}) + 1 + \alpha \max_{1\le j\le K} \sigma_j(\mathbf{z}) \right)_+ \tag{14}$$

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

Note that (13) and (14) are reminiscent of the large-margin classifier learning in traditional supervised learning [62]. 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 [22, 71, 58, 8].

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 (14).

### 3.2. Variational Inference with Deep Kernels

We describe our scalable variational inference approach to approximate the posterior (9). Although there are scalable **GP** approximation schemes based on the random feature expansion [46] and the pseudo/induced inputs [45, 54, 59, 13], here we adopt the *deep kernel* trick [27, 64] 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 $k(\mathbf{z},\mathbf{z}') := \boldsymbol{\phi}(\mathbf{z})^\top \boldsymbol{\phi}(\mathbf{z}')$, 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 $f_j(\mathbf{z}) = \mathbf{w}_j^\top \boldsymbol{\phi}(\mathbf{z})$ with $\mathbf{w}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ independently for $j = 1, \ldots, K$. We let $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K]^\top$. 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 $\mathrm{Cov}(f(\mathbf{z}), f(\mathbf{z}')) = \mathrm{Cov}(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{z}), \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{z}')) = \boldsymbol{\phi}(\mathbf{z})^\top \boldsymbol{\phi}(\mathbf{z}') = k(\mathbf{z},\mathbf{z}')$. Consequently, the source-driven $\mathcal{H}$ prior (9) becomes

$$p(\mathbf{W}|\mathcal{D}_S) \propto \prod_{j=1}^{K} \mathcal{N}(\mathbf{w}_j; \mathbf{0}, \mathbf{I}) \cdot \prod_{i=1}^{N_S} P(y_i^S | \mathbf{W}\boldsymbol{\phi}(\mathbf{z}_i^S)). \tag{15}$$

Since computing (15) is intractable, we introduce a variational density $q(\mathbf{W})$ to approximate it. We assume a fully factorized Gaussian,

$$q(\mathbf{W}) = \prod_{j=1}^{K} \mathcal{N}(\mathbf{w}_j; \mathbf{m}_j, \mathbf{S}_j), \tag{16}$$

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:

$$\log P\Big(\{y_i^S\}_{i=1}^{N_S} \,\Big|\, \{\mathbf{z}_i^S\}_{i=1}^{N_S}, \boldsymbol{\phi}(\cdot)\Big) \ge \mathrm{ELBO}, \tag{17}$$

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

$$\mathrm{ELBO} := \sum_{i=1}^{N_S} \mathbb{E}_{q(\mathbf{W})}\big[\log P(y_i^S | \mathbf{W}\boldsymbol{\phi}(\mathbf{z}_i^S))\big] - \sum_{j=1}^{K} \mathrm{KL}\big(q(\mathbf{w}_j) \,||\, \mathcal{N}(\mathbf{w}_j; \mathbf{0}, \mathbf{I})\big), \tag{18}$$

with the likelihood stemming from (8). As the gap in (17) 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 $\phi$) and the embedding[6] $\mathbf{G}$ can potentially improve the marginal likelihood, i.e., the left hand side in (17).

In optimizing the ELBO (18), the KL term (denoted by KL) can be analytically derived as

$$\text{KL} = \frac{1}{2}\sum_{j=1}^{K}\big(\,\text{Tr}(\mathbf{S}_j) + ||\mathbf{m}_j||_2^2 - \log\det(\mathbf{S}_j) - d\,\big). \quad (19)$$

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 [30]) to facilitate optimization. That is, for each $j$ and $m$,

$$\mathbf{w}_j^{(m)} = \mathbf{m}_j + \mathbf{S}_j^{1/2}\boldsymbol{\epsilon}_j^{(m)}, \quad \boldsymbol{\epsilon}_j^{(m)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (20)$$

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:

$$\text{LL} = \frac{1}{M}\sum_{m=1}^{M}\frac{N_S}{|B_S|}\sum_{i\in B_S}\log P(y_i^S|\mathbf{W}^{(m)}\boldsymbol{\phi}(\mathbf{z}_i^S)). \quad (21)$$

### 3.3. Optimization Strategy

We combine the maximum posterior separation criterion in (14) with the variational inference of the previous section to arrive at the comprehensive optimization task.

Our approximate posterior (16) leads to closed-form expressions for $\mu_j(\mathbf{z})$ and $\sigma_j(\mathbf{z})$ in (10–11) as follows:

$$\mu_j(\mathbf{z}) \approx \mathbf{m}_j^\top\boldsymbol{\phi}(\mathbf{z}), \quad \sigma_j(\mathbf{z}) \approx \big(\boldsymbol{\phi}(\mathbf{z})^\top\mathbf{S}_j\boldsymbol{\phi}(\mathbf{z})\big)^{1/2}. \quad (22)$$

With $q(\mathbf{W})$ fixed, we rewrite our posterior maximum separation loss in (14) as follows. We consider stochastic optimization with a random mini-batch $B_T \subset \mathcal{D}_T = \{\mathbf{z}_i^T\}_{i=1}^{N_T}$ sampled from the target domain data.

$$\text{MS} := \frac{1}{|B_T|}\sum_{i\in B_T}\Bigg(\max_{j\neq j^*}\mathbf{m}_j^\top\boldsymbol{\phi}(\mathbf{z}_i^T) - \max_{1\le j\le K}\mathbf{m}_j^\top\boldsymbol{\phi}(\mathbf{z}_i^T) + 1 + \alpha\max_{1\le j\le K}\big(\boldsymbol{\phi}(\mathbf{z}_i^T)^\top\mathbf{S}_j\boldsymbol{\phi}(\mathbf{z}_i^T)\big)^{1/2}\Bigg)_+ \quad (23)$$

Combining all objectives thus far, our **GPDA** algorithm[7] can be summarized as the following two optimizations alternating with each other:

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

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

[6] Note that the inputs $\mathbf{z}$ also depend on $\mathbf{G}$.

[7] 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 $\mathcal{Z}$. See Supplement for details.

## 4. Related Work

There has been extensive prior work on domain adaptation [10]. Recent approaches have focused on transferring deep neural network representations from a labeled source dataset to an unlabeled target domain by matching the distributions of features between different domains, aiming to extract domain-invariant features [48, 4, 9, 40, 50, 67, 65, 6, 37, 49]. To this end, it is critical to first define a measure of distance (divergence) between source and target distributions. One popular measure is the non-parametric Maximum Mean Discrepancy (**MMD**) (adopted by [6, 66, 35]), which measures the distance between the sample means of the two domains in the reproducing Kernel Hilbert Space (**RKHS**) induced by a pre-specified kernel. The deep Correlation Alignment (**CORAL**) method [57] attempted to match the sample mean and covariance of the source/target distributions, while it was further generalized to potentially infinite-dimensional feature spaces in [69] to effectively align the **RKHS** covariance descriptors across domains.

The Deep Adaptation Network (**DAN**) [34] applied **MMD** to layers embedded in a **RKHS** to match higher order moments of the two distributions more effectively. The Deep Transfer Network (**DTN**) [68] achieved alignment of source and target distributions using two types of network layers based on the **MMD** distance: the shared feature extraction layer that can learn a subspace that matches the marginal distributions of the source and the target samples, and the discrimination layer that can match the conditional distributions by classifier transduction.

Many recent **UDA** approaches leverage deep neural networks with the adversarial training strategy [48, 4, 9, 40, 50, 67], which allows the learning of feature representations to be simultaneously discriminative for the labeled source domain data and indistinguishable between source and target domains. For instance, [16] proposed a technique called the Domain-Adversarial Training of Neural Networks (**DANN**), which allows the network to learn domain invariant representations in an adversarial fashion by adding an auxiliary domain classifier and back-propagating inverse gradients. The Adversarial Discriminative Domain Adaptation (**ADDA**) [60] first learns a discriminative feature subspace using the labeled source samples. Then, it encodes the target data to this subspace using an asymmetric transformation learned through a domain-adversarial loss. The **DupGAN** [26] proposed a **GAN**-like model [20] with duplex discriminators to restrict the latent representation to be domain invariant but category preserving.

In parallel, within the shared-latent space framework, [32] proposed an unsupervised image-to-image translation (**UNIT**) framework based on the **Coupled GAN**s [33]. Another interesting idea is the pixel-level domain adaptation method (**PixelDA**) [6] where they imposed alignment of distributions not in the feature space but directly in the raw

**(a)** Digits. **(b)** Traffic Signs. **(c)** VisDA.

**Fig. 2:** Example images from benchmark datasets. (a) Samples from MNIST, USPS, and SVHN datasets. (b) Samples from SYN SIGNS (first two rows), and GTSRB.

pixel space via the adversarial approaches. The intention is to adapt the source samples as if they were drawn from the target domain, while maintaining the original content. Similarly, [41] utilized the **CycleGAN** [70] to constrain the features extracted by the encoder network to reconstruct the images in both domains. In [52], they proposed a joint adversarial discriminative approach that can transfer the information of the target distribution to the learned embedding using a generator-discriminator pair.

## 5. Experimental Results

We compare the proposed method[8] with state-of-the-art on standard benchmark datasets[9]. Digit classification task consists of three datasets, containing ten digit classes: **MNIST** [31], **SVHN** [42], **USPS** [60]. We also evaluated our method on the traffic sign datasets, Synthetic Traffic Signs (SYN SIGNS) [39] and the German Traffic Signs Recognition Benchmark [55] (GTSRB), which contain 43 types of signs. Finally, we report performance on **VisDA** object classification dataset [43] with more than 280K images across twelve categories. The details of the datasets as well as additional experiments on the Office-31 dataset are available in the Supplementary Material. Fig. 2 illustrates image samples from different datasets and domains.

We evaluate the performance of all methods with the classification accuracy score. We used ADAM [29] 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 $\lambda$ and $\alpha$ will be discussed in Sec. 5.3. We also used the same network structure as [51]. Specifically, we employed the CNN architecture used in [16] and [6] for digit and traffic sign datasets and used ResNet101 [25] model pre-trained on Imagenet [11]. 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. The training details for comparing methods are available in our Supplementary material due to the space limit.

[8] Our code is publicly available in https://seqam-lab.github.io/GPDA/

[9] While we focus on classification tasks, our approach can be readily applied to and formulated for other related tasks, e.g., semantic segmentation treated as multiple concurrent classification problems.

### 5.1. Results on Digit and Traffic Signs datasets

We show the accuracy of different methods in Tab. 1. 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 [5], 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.

### 5.2. Results on VisDA dataset

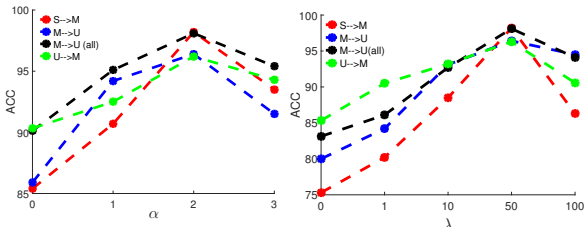
Results for this experiment are summarized in Tab. 2. We observe that our **GPDA** achieved, on average, the best performance compared to other competing methods. Due to vastly varying difficulty of classifying different categories of objects, in addition to reporting the average classification accuracy 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, Eq. 23. 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.

### 5.3. Ablation Studies

Two complementary studies are conducted to investigate the impact of two hyper-parameters $\alpha$ and $\lambda$, 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. $\alpha$ and $\lambda$, with results depicted in Fig. 3a and 3b, 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 Fig. 3b, we see that when $\lambda = 0$ (no **MCD** regularization term), the

**Tab. 1:** Classification results on the digits and traffic signs datasets (best viewed in color). The best score is in bold red, second best in light red. Results are cited from each study. The score of MMD is cited from DSN [7]. † 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 † [34] | 71.1 | 91.1 | - | 81.1 | - |
| DANN † [16] | 71.1 | 88.7 | $77.1^{1.8}$ | 85.1 | $73.0^{0.2}$ |
| DSN † [7] | 82.7 | 93.1 | 91.3 | - | - |
| ADDA [60] | $76.0^{1.8}$ | - | $89.4^{0.2}$ | - | $90.1^{0.8}$ |
| CoGAN [33] | - | - | $91.2^{0.8}$ | - | $89.1^{0.8}$ |
| PixelDA [6] | - | - | - | 95.9 | - |
| ATDA † [50] | 86.2 | 96.1 | - | - | - |
| ASSC [24] | $95.7^{1.5}$ | $82.8^{1.3}$ | - | - | - |
| DRCN [18] | $82.0^{0.1}$ | - | $91.8^{0.09}$ | - | $73.7^{0.04}$ |
| G2A [53] | 92.4 | - | 92.8 | 95.3 | 90.8 |
| SimNet [44] | - | - | - | 96.4 | 95.6 |
| MCDA ($n=2$) | $94.2^{2.6}$ | $93.5^{0.4}$ | $92.1^{0.8}$ | $93.1^{1.9}$ | $90.0^{1.4}$ |
| MCDA ($n=3$) | $95.9^{0.5}$ | $94.0^{0.4}$ | $93.8^{0.8}$ | $95.6^{0.9}$ | $91.8^{0.9}$ |
| MCDA ($n=4$) | $96.2^{0.4}$ | $94.4^{0.3}$ | $94.2^{0.7}$ | $96.5^{0.3}$ | $94.1^{0.3}$ |
| GPDA | **$98.2^{0.1}$** | **$96.19^{0.2}$** | **$96.45^{0.15}$** | **$98.11^{0.1}$** | **$96.37^{0.1}$** |




**(a)** Sensitivity to $\alpha$ **(b)** Sensitivity to $\lambda$

**Fig. 3:** 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.

performance drops considerably. As $\lambda$ increases from 0 to 50, the performance also 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 Fig. 3a, 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.

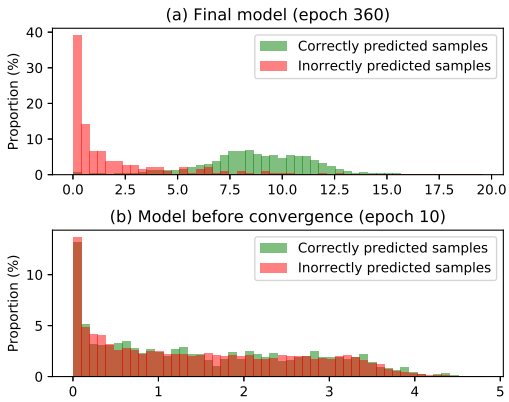



**Fig. 4:** Histograms of prediction (un)certainty for our models: (a) after convergence, (b) at an early stage of training. Abscissa is the Bhattacharyya distance b/w two largest mean posteriors, an indication of *prediction certainty*; the higher the distance, the more certain the prediction is. For 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).

## 5.4. 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, $p(f_{j^*}(\mathbf{z})|\mathcal{D}_S)$ and $p(f_{j^\dagger}(\mathbf{z})|\mathcal{D}_S)$, where $j^*$ and $j^\dagger$ are the indices of the largest and the second largest among the posterior means $\{\mu_j(\mathbf{z})\}_{j=1}^K$, respectively (c.f., (10)). 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 [12] between the two cohorts. In our variational Gaussian approximation (22), the Bhattacharyya distance can be computed in a closed form; Supplement for details.

The histograms of the distances are depicted in Fig. 4, where we contrast the two models, one at an early stage of training and the other after convergence. Our final model in Fig. 4(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, Fig. 4(b), the two posteriors overlap strongly (small distances along hor-

**Tab. 2:** Accuracy of the ResNet model fine-tuned on the VisDA dataset. All models adopt ResNet101 except for [44] which used ResNet152. Last column shows the average rank of each method over all classes. The best (in bold red), the 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 | 7.25 |
| MMD [34] | 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 [16] | 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 [44] | **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** |

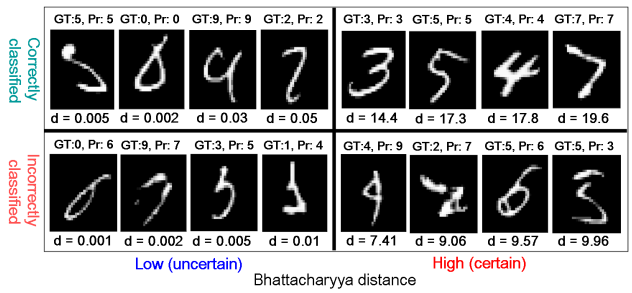


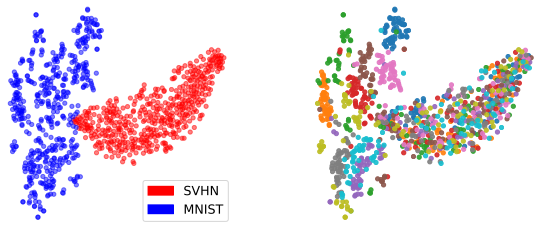

**Fig. 5:** Selected test (**MNIST**) images according to the Bhattacharyya distances. Right: samples with low distances (uncertain prediction). Left: high distances (certain prediction). Top: correctly classified by our model. Bottom: incorrectly classified. For each image, GT, Pr, and $d$ means ground-truth label, predicted label, and the distance, respectively.

izontal axis) for most samples regardless of the correctness of prediction. This confirms our algorithm enforces posterior separation by large margin during the training process.

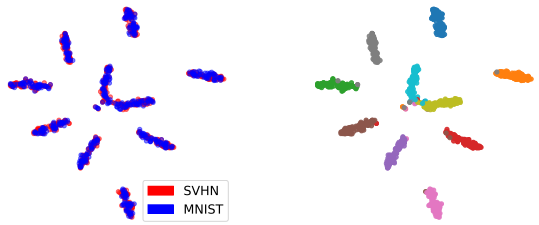
This analysis also suggests that the measure of prediction uncertainty provided by our **GPDA** model, can be used as an *indicator of prediction quality*, namely whether the prediction made by our model is trustworthy or not. To verify this, we depict some sample test images in Fig. 5. We differentiate 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.

### 5.5. Analysis of Shared Space Embedding

We use t-SNE [36] on **VisDA** dataset to visualize the feature representations from different classes. Fig. 6 depicts the embedding of the learned features $G(\mathbf{x})$, and the original features $\mathbf{x}$. Colors indicate source (red) and target (blue) domains. Notice that **GPDA** significantly reduces the domain mismatch, resulting in the expected tight clustering. This is partially due to the use of the proposed probabilistic **MCD** approach, which shrinks the classifier hypothesis




**(a)** Original (by domain) **(b)** Original (by classes)




**(c)** GPDA (by domain) **(d)** GPDA (by classes)

**Fig. 6:** 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**.

class to contain only consistent classifiers on target samples while exploiting the uncertainty in the prediction.

## 6. Conclusion

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.

## References


[1] 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. 1

[2] 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. 1, 2

[3] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation, 2007. In Advances in Neural Information Processing Systems. 1, 2

[4] S. Benaim and L. Wolf. One-sided unsupervised domain mapping. In *Advances in Neural Information Processing Systems (NIPS)*, pages 752–762, 2017. 5

[5] 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. 6

[6] 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. 5, 6, 7

[7] 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. 2, 7

[8] W.-L. Chao, B. Gong, K. Grauman, and F. Sha. Large-Margin Determinantal Point Processes, 2015. Uncertainty in AI. 4

[9] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3733–3742, 2017. 5

[10] G. Csurka. A comprehensive survey on domain adaptation for visual applications. pages 1–35. Springer, 2017. 5

[11] 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. 6

[12] K. G. Derpanis. The bhattacharyya measure. *Mendeley Computer*, 1(4):1990–1992, 2008. 7

[13] A. Dezfouli and E. V. Bonilla. Scalable inference for Gaussian process models with black-box likelihoods, 2015. In Advances in Neural Information Processing Systems. 4

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

[15] M. Gabavisani and V. M. Patel. Adversarial domain adaptive subspace clustering. In *IEEE International Conference on Identity, Security, and Behavior Analysis*, pages 1–8. IEEE, 2018. 1

[16] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. *International Conference on Machine Learning (ICML)*, 2015. 1, 5, 6, 7, 8

[17] 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. 2

[18] 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. 1, 7

[19] 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. 1

[20] 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. 2, 5

[21] 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. 1

[22] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization, 2004. In Proc. of Advances in Neural Information Processing Systems. 4

[23] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schlkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(1):723–773, 2012. 2

[24] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *International Conference on Computer Vision (ICCV)*, volume 2, page 6, 2017. 7

[25] 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. 6

[26] 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. 5

[27] 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). 2, 4

[28] 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. 1

[29] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representation (ICLR)*, 2015. 6

[30] D. P. Kingma and M. Welling. Auto-encoding variational Bayes, 2014. In Proceedings of the Second International Conference on Learning Representations, ICLR. 5

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

[32] 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. 5

[33] 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. 5, 7

[34] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *International Conference on Machine Learning (ICML)*, 2015. 5, 7, 8

[35] 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. 1, 5

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

[37] M. Mancini, L. Porzi, S. R. Bulò, B. Caputo, and E. Ricci. Boosting domain adaptation by discovering latent domains. *arXiv preprint arXiv:1805.01386*, 2018. 5

[38] 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. 1

[39] 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. 6

[40] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto. Few-shot adversarial domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 6673–6683, 2017. 5

[41] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim. Image to image translation for domain adaptation. *arXiv preprint arXiv:1712.00479*, 13, 2017. 6

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

[43] 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. 6

[44] P. O. Pinheiro. Unsupervised domain adaptation with similarity learning. *CVPR*, 2018. 7, 8

[45] 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. 4

[46] 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. 4

[47] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006. 2, 3

[48] S.-A. Rebuffi, H. Bilen, and A. Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems (NIPS)*, pages 506–516, 2017. 5

[49] A. Rozantsev, M. Salzmann, and P. Fua. Residual parameter transfer for deep domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, 2018. 5

[50] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. *International Conference on Machine Learning (ICML)*, 2017. 5, 7

[51] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. *Computer Vision and Pattern Recognition*, 2018. 1, 2, 6

[52] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. *ArXiv e-prints, abs/1704.01705*, 2017. 6

[53] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. *CVPR*, 2018. 7

[54] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs, 2006. In Advances in Neural Information Processing Systems. 4

[55] 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. 6

[56] 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. 1

[57] 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. 5

[58] B. Taskar, C. Guestrin, and D. Koller. Max-Margin Markov Networks, 2003. In Advances in Neural Information Processing Systems. 4

[59] 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. 4

[60] 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. 2, 5, 6, 7

[61] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance, 2014. arXiv:1412.3474. 2

[62] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998. 4

[63] 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. 2

[64] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning, 2016. AI and Statistics (AISTATS). 2, 4

[65] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 5

[66] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *International Conference on Learning Representation (ICLR)*, 2017. 5

[67] J. Zhang, W. Li, and P. Ogunbona. Joint geometrical and statistical alignment for visual domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 5

[68] X. Zhang, F. X. Yu, S.-F. Chang, and S. Wang. Deep transfer network: Unsupervised domain adaptation. *arXiv preprint arXiv:1503.00591*, 2015. 5

[69] Z. Zhang, M. Wang, Y. Huang, and A. Nehorai. Aligning infinite-dimensional covariance matrices in reproducing kernel hilbert spaces for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3437–3445, 2018. 5

[70] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017. 6

[71] X. Zhu and A. B. Goldberg. *Introduction to semi-supervised learning*. Morgan & Claypool, 2009. 4