Transferable Representation Learning with Deep Adaptation Networks

Mingsheng Long, Yue Cao, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan

Abstract—Domain adaptation studies learning algorithms that generalize across source domains and target domains that exhibit different distributions. Recent studies reveal that deep neural networks can learn transferable features that generalize well to similar novel tasks. However, as deep features eventually transition from general to specific along the network, feature transferability drops significantly in higher task-specific layers with increasing domain discrepancy. To formally reduce the effects of this discrepancy and enhance feature transferability in task-specific layers, we develop a novel framework for deep adaptation networks that extends deep convolutional neural networks to domain adaptation problems. The framework embeds the deep features of all task-specific layers into reproducing kernel Hilbert spaces (RKHSs) and optimally matches different domain distributions. The deep features are made more transferable by exploiting low-density separation of target-unlabeled data in very deep architectures, while the domain discrepancy is further reduced via the use of multiple kernel learning that enhances the statistical power of kernel embedding matching. The overall framework is cast in a minimax game setting. Extensive empirical evidence shows that the proposed networks yield state-of-the-art results on standard visual domain-adaptation benchmarks.

Index Terms—Domain adaptation, deep learning, convolutional neural network, two-sample test, multiple kernel learning

1 Introduction

The generalization error of supervised learning systems that are deployed in domains different from the domain in which they are trained can be unsatisfyingly large. Moreover, it is rarely realistic to simply assume sufficient training data for each new domain. Rather, there is a need for versatile algorithms that reduce the need for large labeled data sets across multiple domains. *Domain adaptation* addresses this need in the setting in which we have data from two or more related domains characterized by distributions that are broadly similar but different in significant ways. For example, an object detection model trained on manually annotated images may not generalize well to new images obtained under substantial variations in pose, occlusion, or illumination. Domain adaptation aims to enable knowledge transfer from a labeled source domain to an unlabeled target domain by exploring domain-invariant structures that bridge different domains under substantial distributional discrepancy [1], [2].

One important strategy for domain adaptation is to learn domain-invariant models from data which bridge source and target domains via an isomorphism-inducing latent feature space. In this direction, a fruitful line of prior work has focused on learning shallow features by jointly minimizing a distance metric of domain discrepancy [3], [4], [5], [6], [7]. However, recent studies have shown that deep networks can learn much more transferable features for domain adaptation [8], [9], [10], producing breakthrough results on some domain-adaptation datasets. Deep networks are able to disentangle meaningful factors of variation underlying data and group features hierarchically in accordance with their relatedness to the invariant factors, yielding robustness to cross-domain variation.

While deep networks are able to learn transferable features, recent findings reveal that deep features must eventually transition from general to specific along the network, and the feature transferability drops substantially in higher task-specific layers under increasing domain discrepancy. In other words, the features in the higher layers of the deep network will depend heavily on specific datasets and tasks [10], which are task-specific and not safely transferable to new tasks. Another curious phenomenon is that the disentangling of the factors of variation in higher layers of deep networks may further increase domain discrepancy, as different domains will become more distinguishable in higher layers [8]. Though deep features are more discriminative for classification, increasing dataset shift may deteriorate domain adaptation performance, resulting in statistically *unbounded* error for the target task [2], [11], [12].

This paper is motivated from recent work on the transferability of deep neural networks [10]. We propose a new framework for deep adaptation networks which generalizes deep convolutional networks to the domain adaptation scenario. The main idea is to formally reduce the dataset shift and enhance the feature transferability in task-specific layers of deep networks. To achieve this goal, the deep

- *M. Long, Y. Cao, Z. Cao, and J. Wang are with the School of Software, Tsinghua University, Beijing 100084, China. E-mail: {mingsheng, jimwang}@tsinghua.edu.cn, {yue-cao14, caozj14}@mails.tsinghua.edu.cn.*
- *M.I. Jordan is with the Department of EECS, Department of Statistics, University of California, Berkeley, Berkeley, CA 94720-4600. E-mail: jordan@berkeley.edu.*

Manuscript received 5 Jan. 2016; revised 3 July 2018; accepted 11 Aug. 2018. Date of publication 5 Sept. 2018; date of current version 31 Oct. 2019. (Corresponding author: Jianmin Wang.) Recommended for acceptance by F. Sha. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TPAMI.2018.2868685

features of task-specific higher layers are embedded in reproducing kernel Hilbert spaces (RKHSs) in which the kernel embeddings of different domain distributions are well matched. Since kernel-embedding-based matching is sensitive to kernel choices, an optimal multi-kernel learning method [13] and a maximal testing power method [14] are employed in a *minimax* game to further reduce the domain discrepancy. The deep features are made more transferable by employing a low-density separation (entropy minimization) criterion for target-unlabeled data [15], in the context of the very deep GoogLeNet [16] and ResNet [17]. This leads to a minimax-game learning framework that learns transferable features with statistical guarantees. We propose a linear-time unbiased estimate of the kernel mean embedding to enable scalable training, which is very desirable for deep domain adaptation. Finally, since deep models that are pre-trained on large repositories such as ImageNet [18] are effective for general perception tasks [10], [19], [20], [21], the proposed deep adaptation networks are trained by fine-tuning from pre-trained AlexNet [22], GoogLeNet [16] and ResNet [17], using ImageNet as the training data. Extensive empirical evidence shows that the proposed models yield state-of-the-art results on standard domain adaptation benchmarks.

This work invovles a substantial extension to an earlier conference paper [23]. The contributions are as follows:

- We propose a novel framework—deep adaptation networks (DAN)—for domain adaptation, where *multiple* task-specific layers are adapted, benefitting from "deep adaptation."
- We explore *multiple* kernels for matching deep representations across domains, benefiting from "optimal matching."
- We further learn *distinguishable* test locations for a new two-sample test statistic to maximize the distinguishability of distributions, benefitting from "adversarial matching."
- We finally exploit the *low-density separation* criterion using entropy minimization on the target-unlabeled data, benefitting from "semi-supervised adaptation."

2 Related Work

This work is related to transfer learning [1], which builds models that can bridge different domains or tasks, explicitly taking the cross-domain discrepancy into account. Transfer learning mitigates the burden of manual labeling in machine learning [3], [5], [6], [7], [24], [25], computer vision [4], [20], [26], [27], [28], [29] and natural language processing [30], [31]. There is a consensus that cross-domain discrepancy in distributions of different domains should be formally reduced. The major bottleneck is how to match different domain distributions effectively. Most existing methods learn a new shallow representation model in which the domain discrepancy can be explicitly minimized. However, without learning deep features to suppress domain-specific exploratory factors of variations, the transferability of shallow features is restricted by task-specific structures. There are several very recent attempts to learn domain-invariant features in the context of shallow networks [32], but these proposals generally underperform deep networks.

Deep networks can learn abstract representations that disentangle different explanatory factors of variation [33]. The learned deep representations manifest invariant factors underlying different populations and are transferable from the original tasks to similar future tasks [10]. Hence, deep networks have been well explored for domain adaptation [8], [19], [20], multimodal and multi-task learning problems [30], [34], where significant performance gains have been witnessed. These methods depend on the assumption that deep networks can learn the desired transferable representations across different tasks. In reality, the domain discrepancy can be reduced, but not removed, by deep networks [8], [35]. Dataset shift has posed a general bottleneck to the transferability of deep representations, resulting in statistically *unbounded* risk for target tasks [2], [11], [12].

This work is primarily motivated by Yosinski et al. [10], which comprehensively quantifies the feature transferability of deep convolutional neural networks. That method focuses on a different scenario where the learning tasks are different across domains, hence it requires sufficient target-labeled examples such that the source network can be fine-tuned to the target task. In many real problems, labeled data is usually limited especially for a new target task, hence the method cannot be directly applicable to domain adaptation. Nonetheless, [10] reveals that deep features must eventually transition from general to specific along the network, and feature transferability drops substantially in multiple higher layers, providing the motivation for our work.

Domain discrepancy should be formally reduced to achieve smaller transfer errors [2], [11], [12]. Several lines of work [29], [35], [36] extend deep convolutional networks (CNN) to domain adaptation either by adding an adaptation layer through which the means of distributions are matched [35], or by adding a fully connected subnetwork as a domain discriminator which the deep features are learned to confuse in a domain-adversarial training paradigm [29], [36]. In particular, adversarial domain adaptation [29], [36], [37], [38], [39] combines adversarial learning and domain adaptation, which performs strongly on many datasets. These methods function in a two-player minimax game as generative adversarial networks (GANs) [40]. A domain discriminator, e.g., an MLP, is learned by minimizing the classification error for distinguishing the source domain from a target domain, while a deep classification model, e.g., a CNN, learns transferable representations indistinguishable to the domain discriminator. Despite their efficacy, the adversarial matching of a single network layer is not sufficient to close the domain discrepancy, since the dataset shifts may linger in multiple task-specific network layers [10].

The above adversarial domain adaptation methods apply adversarial losses in the feature space. To aim at multi-layer adaptation, there has recently been significant interest in performing adaptation by applying the adversarial losses in the pixel space. Such methods primarily use generative models such as GANs [40] to generate synthetic images in the pixel space and perform cross-domain image translation. Coupled GAN (CoGAN) [41] trains a coupled generative model that learns the joint data distribution across the two domains. A domain invariant classifier is learned by sharing the weights with the discriminator of the CoGAN network. Bousmalis et al. [42] present a pixel-level method that learns a

transformation in the pixel space from one domain to the other in an unsupervised manner. The GAN-based method adapts source images to appear as if drawn from the target domain. In particular, Shrivastava et al. [43] modifies GANs to improve the realism of a simulator's output using unlabeled real data while preserving the annotations from the simulator. These pixel-level methods are complementary to the feature-level methods, but are subject to the risk that the generative model may generate fake images.

While performance has been improved, the above state-of-the-art methods still suffer from several limitations. First, despite the fact that the pixel-level methods may jointly adapt the source and target domains in both feature space and pixel space, the transferability of features are shown to decrease in multiple *higher* layers, while lower-layer features generally transfer well [10]. Second, they add new adaptation layers or new parametric subnetworks, which may be difficult to train given the limited data setting that characterizes domain adaptation [29], [44]. Third, they do not fully exploit the unlabeled target data via a low-density separation principle to refine the source classifier, hence the source classifier may still misspecify the target data.

3 Deep Domain Adaptation Networks

In deep domain adaptation problems, we are provided with a *source* domain $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ of n_s labeled examples and a *target* domain $\mathcal{D}_t = \{\mathbf{x}_i^t\}_{i=1}^{n_t}$ of n_t examples, where the target domain $\mathcal{D}_t = \mathcal{D}_l \cup \mathcal{D}_u$ may contain both labeled data $\mathcal{D}_l = \{(\mathbf{x}_i^l, y_i^l)\}_{i=1}^{n_l}$ and unlabeled data $\mathcal{D}_u = \{\mathbf{x}_i^u\}_{i=1}^{n_u}$, and where $n_l \ll n_s$. Denote by $\mathcal{D}_a = \{(\mathbf{x}_i^a, y_i^a)\}_{i=1}^{n_a}$ all available labeled examples of the source and target domains, $\mathcal{D}_a = \mathcal{D}_s \cup \mathcal{D}_l$. We consider two common scenarios: *unsupervised* domain adaptation where the target domain is fully unlabeled ($\mathcal{D}_l = \emptyset$) and *semi-supervised* domain adaptation where the target domain is partially labeled ($\mathcal{D}_l \neq \emptyset$). The source domain and target domain are sampled from probability distributions p and q respectively, and $p \neq q$. The goal of this paper is to design a deep neural network, $y = f(\mathbf{x})$, which enables the learning of transferable representations to close the source-target discrepancy, such that the target risk $R_t(f) = \Pr_{(\mathbf{x},y)\sim q}[f(\mathbf{x}) \neq y]$ can be minimized by leveraging the source domain supervision.

3.1 Two-Sample Test Statistics

The challenge of domain adaptation mainly arises when the target domain has no or only limited labeled information. To approach this problem, many existing methods aim to bound the target error by the source error plus a discrepancy measure between the source and target distributions p and q [12]. In particular, two-sample testing methods have been adapted for this purpose. These methods accept or reject a null hypothesis ($p = q$) based on the two samples generated from p and q. Examples include *Maximum Mean Discrepancy* (MMD) [13], [45] and *Mean Embedding Test* (ME) [14], [46]. This paper will explore these two testing methods for learning transferable features and closing domain discrepancy.

3.1.1 MMD Test

First, we describe the *generalized* MMD method [13], [47], [48], which maximizes the two-sample test power; i.e., it minimizes the Type-II error, which is the failure to reject a false null hypothesis ($p = q$). Technically, minimizing the Type-II error involves maximizing a *normalized* variant of MMD between domains [13], equivalent to maximizing the distinguishability of data distributions.

Let $\mathcal{H}_k$ be a reproducing kernel Hilbert space (RKHS) endowed with a *characteristic kernel* k. Then the *kernel mean embedding* of a distribution p in $\mathcal{H}_k$ is a unique element $\mu_k(p)$ such that the expectation $\mathbb{E}_{\mathbf{x}\sim p} f(\mathbf{x}) = \langle f(\mathbf{x}), \mu_k(p) \rangle_{\mathcal{H}_k}$ for all $f \in \mathcal{H}_k$. That is, crucial statistical features of the distribution p are encoded into the embedding $\mu_k(p)$ so that we can learn through $\mu_k(p)$ instead of p, removing the need for density estimation of p. The multi-kernel maximum mean discrepancy (MK-MMD) between distributions p and q is defined as the squared distance between kernel mean embeddings of p and q in the RKHS $\mathcal{H}_k$

$$M_k(p, q) \triangleq \left\| \mathbb{E}_{\mathbf{x}^s\sim p}[\boldsymbol{\phi}(\mathbf{x}^s)] - \mathbb{E}_{\mathbf{x}^t\sim q}[\boldsymbol{\phi}(\mathbf{x}^t)] \right\|_{\mathcal{H}_k}^2, \tag{1}$$

where $\boldsymbol{\phi}(\cdot)$ is a nonlinear feature mapping that induces $\mathcal{H}_k$. Given $\mathcal{D}_s$ and $\mathcal{D}_t$ as the samples drawn from distributions p and q respectively, the empirical estimate of MK-MMD is

$$\begin{aligned} M_k(\mathcal{D}_s, \mathcal{D}_t) \triangleq & \frac{1}{n_s^2} \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} k(\mathbf{x}_i^s, \mathbf{x}_j^s) + \frac{1}{n_t^2} \sum_{i=1}^{n_t} \sum_{j=1}^{n_t} k(\mathbf{x}_i^t, \mathbf{x}_j^t) \\ & - \frac{2}{n_s n_t} \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} k(\mathbf{x}_i^s, \mathbf{x}_j^t). \end{aligned} \tag{2}$$

The most important property is that $p = q$ iff $M_k(p, q) = 0$ [45]. The characteristic kernel $k(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}') \rangle$ is defined as the convex combination of m kernels $\{k_u\}$,

$$\mathcal{K} \triangleq \left\{ k = \sum_{u=1}^{m} \beta_u k_u : \sum_{u=1}^{m} \beta_u = 1, \beta_u \geqslant 0, \forall u \right\}, \tag{3}$$

where the constraints on the coefficients $\{\beta_u\}$ are imposed to guarantee that the composed multi-kernel k is characteristic. As theoretically studied in [13], the kernel adopted for mean embeddings of p and q is critical in ensuring high test power; i.e., minimizing the probability of degenerated two-sample tests $M_k(p, q) \to 0$ when $p \neq q$. The kernel mean embeddings by different kernels $\{k_u\}$ with different bandwidths can characterize the distributions at different scales and thus match different orders of moments. By minimizing the Type-II error, we can automatically learn an optimal kernel for cross-domain distribution matching.

3.1.2 ME Test

The ME test [14] is a form of Hotelling's T-squared statistic to compare distributions. The ME test is formally defined as

$$M_k(\mathcal{D}_s, \mathcal{D}_t) \triangleq \max_{\mathcal{V}} \; n \mathbf{z}_n^{\mathsf{T}} \mathbf{S}_n^{-1} \mathbf{z}_n, \tag{4}$$

where $\mathbf{z}_n \triangleq \frac{1}{n} \sum_{i=1}^{n} \mathbf{z}_i$, $\mathbf{S}_n \triangleq \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{z}_i - \mathbf{z}_n)(\mathbf{z}_i - \mathbf{z}_n)^{\mathsf{T}}$, and $\mathbf{z}_i \triangleq \left(k(\mathbf{x}_i^s, \mathbf{v}_j) - k(\mathbf{x}_i^t, \mathbf{v}_j) \right)_{j=1}^{J} \in \mathbb{R}^J$. The ME-test statistic depends on a characteristic kernel k and J test locations $\mathcal{V} = \{\mathbf{v}_j\}_{j=1}^{J} \subset \mathbb{R}^J$ chosen by

$$\mathcal{V} = \arg\max_{\mathcal{V}} \; n \mathbf{z}_n^{\mathsf{T}} \mathbf{S}_n^{-1} \mathbf{z}_n, \tag{5}$$

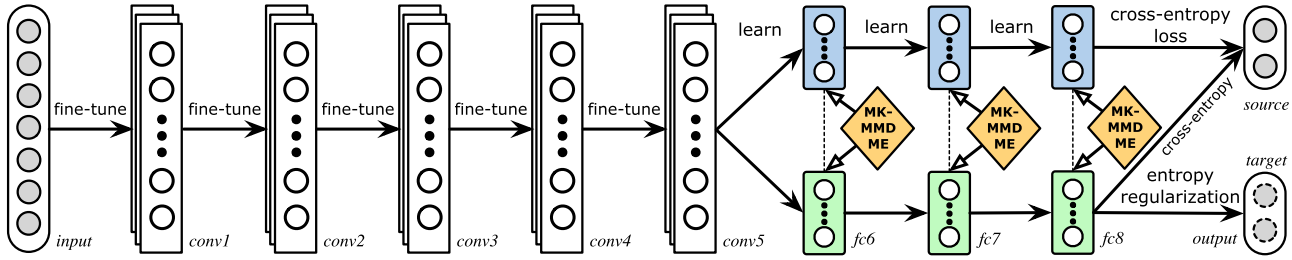


Fig. 1. The *deep adaptation network* (DAN) based on AlexNet [22]. Since deep features transition from general to specific along the network: 1) the features extracted by convolutional layers $conv1$–$conv5$ are transferable, hence these layers are learned via fine-tuning; 2) fully connected layers $fc6$–$fc8$ are tailored to fit task-specific structures, hence they are not safely transferable and should be adapted with MK-MMD/ME minimization. To mitigate the misspecification of the source classifier to the target data, the low-density separation of target data is achieved by entropy minimization.

which maximizes the ME test power for parsimonious and interpretable indication of how and where two distributions differ locally [46]. In this paper, we adopt the *unnormalized* ME test—i.e., without $\mathbf{S}_n^{-1}$—which has been shown by [14] to be a metric on the space of probability measures for any $\mathcal{V}$ and behaves similarly for two-sample testing as the normalized ME in (4). To compare domains of different sizes n_s and n_t, we need to sample the target domain such that the number n of points is the same as the source domain. It has been shown that ME test can be computed in linear time and has greater test power than MMD [46].

A successful strategy to control the domain discrepancy is to find an invariant representation through which the source domain and target domain are made similar [2], [11], [12]. MMD has been extensively explored for this purpose [3], [4], [6], [7], [25], but to date there has been no attempt for learning transferable features via MK-MMD or ME in deep networks. In summary, previous shallow transfer-learning methods may be restricted by weak representation power [33], while previous standard deep-learning methods may be restricted by weak adaptation efficacy [8], [13], [35].

3.2 Deep Adaptation Networks

Deep learning [33] has the ability to extract distributed representations for natural signals such as images and videos. In this paper, we will exploit the above test statistics MK-MMD and ME in deep networks for the learning of transferable features. We start with convolutional neural networks (CNN) [22], a strong model for learning high-quality deep features that can be fine-tuned to novel tasks [9], [19], [20]. The main challenge resides in the fact that the target domain contains no or very few labeled data points, hence directly adapting CNN to the target domain through fine-tuning [19] is infeasible or is prone to over-fitting. In view of our goal of domain adaptation, we design a *deep adaptation network* that can transfer knowledge from a labeled source domain to an unlabeled target domain. Fig. 1 gives an illustration of the proposed DAN model.

We extend the classic AlexNet architecture [22], which consists of five convolutional layers ($conv1$–$conv5$) and three fully connected layers ($fc6$–$fc8$), while $conv1$–$fc7$ are feature layers and $fc8$ is the classifier layer. Specifically, each fully connected layer ℓ learns a nonlinear mapping $\mathbf{h}_i^\ell = a^\ell(\mathbf{W}^\ell \mathbf{h}_i^{\ell-1} + \mathbf{b}^\ell)$, where $\mathbf{h}_i^\ell$ is the ℓ^{th}-layer representation of example $\mathbf{x}_i$, $\mathbf{W}^\ell$ and $\mathbf{b}^\ell$ are the ℓ^{th}-layer weight and bias parameters, and a^ℓ is the ℓ^{th}-layer activation function, taken as Rectifier Linear Units (ReLU) $a^\ell(\mathbf{x}) = \max(\mathbf{0}, \mathbf{x})$ for the hidden layers and Softmax units $a^\ell(\mathbf{x}) = e^{\mathbf{x}} / \sum_{j=1}^{|\mathbf{x}|} e^{x_j}$ for the output layer. Denote by $\mathcal{F} = \{\mathbf{W}^\ell, \mathbf{b}^\ell\}_{\ell=1}^{l}$ the set of network parameters (l layers in total). The empirical error of a CNN classifier f on labeled data $\mathcal{D}_a = \{(\mathbf{x}_i^a, y_i^a)\}_{i=1}^{n_a}$ is

$$\min_{f \in \mathcal{F}} \frac{1}{n_a} \sum_{i=1}^{n_a} L\big(f\big(\mathbf{x}_i^a\big), y_i^a\big), \tag{6}$$

where $f\big(\mathbf{x}_i^a\big)$ is the probability distribution that the CNN assigns point $\mathbf{x}_i^a$ to all labels, $L(\cdot,\cdot)$ is the cross-entropy loss function, $L\big(f\big(\mathbf{x}_i^a\big), y_i^a\big) = -\sum_{j=1}^{c} 1\{y_i^a = j\} \log f_j\big(\mathbf{x}_i^a\big)$, c is the number of classes, and $f_j\big(\mathbf{x}_i^a\big) = e^{h_{ij}^{a,l}} / \sum_{j'} e^{h_{ij'}^{a,l}}$ is a softmax function on the last layer activation $\mathbf{h}_i^{a,l}$ computing the probability of assigning point $\mathbf{x}_i^a$ to class j.

Based on the quantification study of feature transferability [10], the convolutional layers can learn generic features that are transferable in layers $conv1$–$conv3$ and slightly specific in layers $conv4$–$conv5$ [10]. Hence, when adapting the AlexNet model from the source domain to the target domain, we opt to fine-tune $conv1$–$conv5$ such that the efficacy of feature co-adaptation can be maximally preserved [49]. Since we will perform distribution matching only for fully connected layers and pooling layers, we will not elaborate on the computational details of the convolutional layers.

3.2.1 Distribution Matching

The deep features learned by CNNs can disentangle the factors of variation underlying data distributions and facilitate knowledge transfer [19], [33]. However, recent findings also reveal that the deep features can reduce, but not remove, the cross-domain distribution discrepancy [10], [35]. The deep features in standard CNNs must eventually transition from general to specific along the network, and the transferability gap grows with the cross-domain discrepancy and becomes intolerably large when transferring the higher layers $fc6$–$fc8$ [10]. In other words, the fc layers are tailored to their original task at the expense of degraded performance on the target task. Hence they cannot transfer safely to the target domain by fine-tuning under little target supervision.

To promote transferability, this paper fine-tunes a CNN on labeled examples from the source and target domains (if they are available in the target domain), and requires the distributions of the source and target domains to become similar under the representations of fully connected layers $\mathcal{L} = \{fc6, fc7, fc8\}$. We minimize a *multi-layer* MK-MMD or

ME penalty by substituting the representations $\{\mathbf{h}_i^\ell\}_{\ell\in\mathcal{L}}$ into test statistics MK-MMD (2) or ME-test (4) respectively

$$\min_{f\in\mathcal{F}}\max_{k\in\mathcal{K}}\sum_{\ell\in\mathcal{L}}M_k\big(\mathcal{D}_s^\ell,\mathcal{D}_t^\ell\big), \tag{7}$$

where $\mathcal{D}_s^\ell=\{\mathbf{h}_i^{s\ell}\}$ and $\mathcal{D}_t^\ell=\{\mathbf{h}_i^{t\ell}\}$ are the ℓ^{th}-layer hidden representations of the source and target points respectively, and $M_k\big(\mathcal{D}_s^\ell,\mathcal{D}_t^\ell\big)$ is the MK-MMD/ME between the source and target data evaluated by the ℓ^{th}-layer representations. $\mathcal{L}$ is the index of layers over which the MMD/ME penalty is taken, whose setting depends on the size of training data and the number of parameters to be learned.

Note that maximizing MK-MMD/ME with respect to the kernel $k\in\mathcal{K}$ is equivalent to maximizing the two-sample test power when the two test statistics (1) and (4) have isotropic covariance structure [13], [47]. This enables deep features to minimize an upper bound on the cross-domain discrepancy, which guarantees the goodness of distribution matching for domain adaptation. To learn smoother kernel combination coefficients, we also consider the non-isotropic covariance structure when deriving learning algorithms in Section 4. This is similar to GANs [40] that optimize the discriminator and generator using different losses. Note also that for the ME-test, we need to further choose the optimal test locations $\mathcal{V}$ by maximizing Equation (5).

3.2.2 Entropy Minimization

Since it is generally impossible to achieve zero cross-domain distribution discrepancy—i.e., the MK-MMD/ME in Equation (7) do not vanish—the source and target domains may still be somewhat different after optimizing Equations (6) and (7). In that case, the deep CNN classifier still fails to classify the target-unlabeled data accurately. Due to the distribution discrepancy, the boundary of the deep classifier must be able to pass through the low-density regions of the target-unlabeled data in order to perform well on the target domain. From the perspective of semi-supervised learning, the target-unlabeled data will be an informative prior to the source classifier because the source and target are different distributionally [50]. In contrast to semi-supervised learning where the labeled data are limited, in domain adaptation we have rich source-labeled data but they are not identically distributed with the target-unlabeled data.

In this paper, we further exploit the *entropy minimization* principle [15] in the deep adaptation networks, which favors the low-density separation between classes through minimizing the conditional-entropy of the class distribution $p(y_i^u=j|\mathbf{x}_i^u;f)=f_j(\mathbf{x}_i^u)$ on unlabeled target data $\mathcal{D}_u$

$$\min_{f\in\mathcal{F}}\frac{1}{n_u}\sum_{i=1}^{n_u}H\big(f\big(\mathbf{x}_i^u\big)\big), \tag{8}$$

where $f\big(\mathbf{x}_i^u\big)$ is the probability distribution that the CNN assigns an unlabeled point $\mathbf{x}_i^u$ to all classes, and $H(\cdot)$ is the entropy loss, $H\big(f\big(\mathbf{x}_i^u\big)\big)=-\sum_{j=1}^{c}f_j\big(\mathbf{x}_i^u\big)\log f_j\big(\mathbf{x}_i^u\big)$, c is the number of classes, and $f_j\big(\mathbf{x}_i^u\big)$ is the probability of predicting point $\mathbf{x}_i^u$ to class j. By minimizing the entropy penalty (8), equivalent to minimizing the *uncertainty* of predicting the labels of the target data, the classifier f is directly accessible to the target-unlabeled data and will adjust itself to pass through the low-density region of the target domain.

3.2.3 Minimax Optimization

As we have emphasized, the invariance of deep features, the adaptation of domain distributions, and the separation of low-density regions all contribute critically to the domain adaptation performance. Hence, we integrate (6), (7) and (8) in a unified *deep adaptation network* as follows:

$$\begin{aligned}\min_{f\in\mathcal{F}}\max_{k\in\mathcal{K}}\ &\frac{1}{n_a}\sum_{i=1}^{n_a}L\big(f\big(\mathbf{x}_i^a\big),y_i^a\big)+\frac{\gamma}{n_u}\sum_{i=1}^{n_u}H\big(f\big(\mathbf{x}_i^u\big)\big)\\&+\lambda\sum_{\ell\in\mathcal{L}}M_k\big(\mathcal{D}_s^\ell,\mathcal{D}_t^\ell\big),\end{aligned} \tag{9}$$

where γ and λ are the tradeoff parameters for the conditional-entropy penalty (8) and the multi-layer MK-MMD/ME penalty (7) respectively. We refer to the DAN variant using MMD as *DAN* and the one using ME as *DAN (ME)*. As training deep CNNs requires a large amount of labeled data that is prohibitive for many domain adaptation applications, we start with the AlexNet model pre-trained on ImageNet 2012 dataset and fine tune it as in [10], [35]. With the proposed DAN model (9), we can learn transferable features that are both discriminative due to entropy minimization and properties of the CNN, and domain-unbiased stemming from MK-MMD or ME minimization.

It is noteworthy that the final formulation of DAN is a *minimax* optimization problem, in the spirit of generative adversarial networks [40]. We maximize the two-sample test statistics with respect to the kernels $k\in\mathcal{K}$ and test locations $\mathcal{V}$ for maximal test power, which makes any subtle hidden difference between the source and target domains distinguishable by the test statistics. This is similar to the domain discriminator that tries to classify the source from the target as accurately as possible. In the two-player game, we minimize the the test statistics with respect to the learned deep features, which aligns the source and target domains and suppresses undesirable hidden differences. However, unlike previous adversarial domain adaptation methods [36], [37], [44], we adopt a *nonparametric* max game with very few parameters k and $\mathcal{V}$ to be optimized. In other words, we enable *adversarial* matching of different domains without a large parametric network, making it easier to reach equilibrium in our minimax game.

The proposed DAN framework has several advantages:

- *Multi-layer* adaptation. As feature transferability gets worse in middle layers and drops significantly in higher layers [10], it is critical to align multiple layers instead of only one layer. Another benefit of multi-layer adaptation is—by jointly adapting the representation layers and the classifier layer—we essentially close the cross-domain gap underlying *both* the marginal distribution and conditional distribution, which is important for domain adaptation [6].
- *Multi-kernel* distribution matching. The choice of kernel is critical to the testing power of MMD/ME since different kernels embed the probability distributions in different RKHSs where different orders of sufficient statistics can be emphasized [13]. This is crucial

for moment matching, which has not been explored by previous domain adaptation methods.
- *Adversarial* matching. By jointly learning the optimal test locations that maximally distinguish two distributions, our method forms a minimax optimization game, where the test locations are learned to maximally discriminate the source and target domains while the deep features are learned to maximally confuse the two-sample test statistics.
- *Low-density* separation. As the target-unlabeled data are not identically distributed to the source-labeled data, the source classifier should pass through the low-density regions of the target data. Distribution adaptation and low-density separation should reinforce each other for better domain adaptation performance, since they are clearly complementary.

3.2.4 Architectural Options

By going even deeper with convolutions, the very deep convolutional networks including GoogLeNet [16] and ResNet [17] achieved state-of-the-art results in the ImageNet Large-Scale Visual Recognition Challenge 2015 (ILSVRC) [18]. The transferability of AlexNet features has been extensively quantified [10] and enhanced by the proposed DAN model. We further enhance the transferability of very deep neural networks within the DAN framework.

- *GoogLeNet:* We extend the GoogLeNet architecture [16], which contains nine *inception* layers ($in1$–$in9$) and three fully connected layers ($fc1$–$fc3$) for three softmax classifiers, which are plugged in to layers $in3$, $in6$ and $in9$ to mitigate vanishing gradients. We add the *multi-layer* MK-MMD/ME penalty on fully connected layers (fc), pooling layers (pl), and concat layers (cc). Denote by $cc\ell$ the concat layer of the ℓ^{th} inception layer. We specify the last softmax classifier $fc3$ as the deep network classifier f in the cross-entropy loss (6) and conditional-entropy penalty (8), and $\mathcal{L} = \{cc8, cc9, fc3\}$ as adaptation layers for the *multi-layer* MK-MMD/ME penalty.
- *ResNet:* We extend the ResNet architecture [17] that contains 50 convolution-pooling layers $conv1$–$pool5$ as feature layers and one fully connected layer fc as classifier layer. We add the *multi-layer* MK-MMD/ME penalty on the fully connected layers (fc) and pooling layers (pl) layers. We specify the fully connected layer fc as the deep network classifier f in the cross-entropy loss (6) and conditional-entropy penalty (8), and $\mathcal{L} = \{pl5, fc\}$ as adaptation layers for the *multi-layer* MK-MMD/ME penalty.

4 Algorithm and Analysis

We derive linear-time algorithms and provide a sketch of the theoretical analysis of the generalization bound for deep domain adaptation.

4.1 Learning Network Parameters

MK-MMD. Using the kernel trick, $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle = k(\mathbf{x}, \mathbf{x}')$, which avoids the explicit manipulation of the infinite-dimensional feature mapping, MK-MMD in (1) is computed as the expectation of kernels: $M_k(p,q) = \mathbb{E}_{\mathbf{x}^s \mathbf{x}'^s} k(\mathbf{x}^s, \mathbf{x}'^s) + \mathbb{E}_{\mathbf{x}^t \mathbf{x}'^t} k(\mathbf{x}^t, \mathbf{x}'^t) - 2\mathbb{E}_{\mathbf{x}^s \mathbf{x}^t} k(\mathbf{x}^s, \mathbf{x}^t)$, where $\mathbf{x}^s, \mathbf{x}'^s \overset{\text{iid}}{\sim} p$ and $\mathbf{x}^t, \mathbf{x}'^t \overset{\text{iid}}{\sim} q$, $k \in \mathcal{K}$. However, this computation incurs a complexity of $O(n^2)$, which is undesirable in the deep-learning setting, as the power of deep networks largely derives from scalable training on large-scale datasets. Moreover, the summation over pairwise similarities between all data points makes *mini-batch* stochastic gradient descent (SGD) more difficult, whereas the use of mini-batch SGD is crucial to the training of deep networks. While prior MMD-based works [3], [25], [35] rarely address this issue, we believe it is critical for deep domain adaptation.

In this paper, we adopt the linear-time unbiased estimate of MK-MMD [13], which can be computed in linear time as

$$\hat{M}_k(\mathcal{D}_s, \mathcal{D}_t) = \frac{2}{n_s} \sum_{i=1}^{n_s/2} d_k(\mathbf{z}_i). \tag{10}$$

Denote a quad-tuple by $\mathbf{z}_i \triangleq (\mathbf{x}^s_{2i-1}, \mathbf{x}^s_{2i}, \mathbf{x}^t_{2i-1}, \mathbf{x}^t_{2i})$, and evaluate the multi-kernel function k on each quad-tuple $\mathbf{z}_i$ by $d_k(\mathbf{z}_i) \triangleq k(\mathbf{x}^s_{2i-1}, \mathbf{x}^s_{2i}) + k(\mathbf{x}^t_{2i-1}, \mathbf{x}^t_{2i}) - k(\mathbf{x}^s_{2i-1}, \mathbf{x}^t_{2i}) - k(\mathbf{x}^s_{2i}, \mathbf{x}^t_{2i-1})$. This unbiased estimate computes MK-MMD as the expectation of independent variables in (1) with $O(n)$ cost. By computing the linear-time estimate of MK-MMD as the sum of $n_s/2$ quad-tuple functions $d_k(\mathbf{z}_i)$, it fits well into the mini-batch stochastic gradient descent.

When training with mini-batch SGD, we only consider the gradient of objective (9) with respect to each point $\mathbf{x}_i$ in a mini-batch $\mathcal{B} \triangleq \{\mathbf{x}_i\}_{i=1}^{|\mathcal{B}|}$ of batch-size $|\mathcal{B}|$. Note that, $\mathcal{B} \subset \mathcal{D}_a \cup \mathcal{D}_u = \mathcal{D}_s \cup \mathcal{D}_t$, that is, each mini-batch may contain both labeled and unlabeled examples from both source and target domains. We maintain a labeled list $\mathcal{B}_a$ and an unlabeled list $\mathcal{B}_u$, $\mathcal{B} = \mathcal{B}_a \cup \mathcal{B}_u$. For each point $\mathbf{x}_i \in \mathcal{B}$, if $\mathbf{x}_i \in \mathcal{B}_a$, the gradient of the empirical error $\frac{\partial L(\mathbf{x}_i, y_i)}{\partial \mathcal{F}^\ell}$ is computed, otherwise $\mathbf{x}_i \in \mathcal{B}_u$, and the gradient of the conditional-entropy penalty $\frac{\partial H(\mathbf{x}_i)}{\partial \mathcal{F}^\ell}$ is computed, both through back-propagation.

Since the linear-time MK-MMD (10) can be decoupled into the sum of quad-tuple functions $d_k(\mathbf{z}_i)$'s, we only need to compute the gradients $\frac{\partial d_k(\mathbf{z}_i^\ell)}{\partial \mathcal{F}^\ell}$ for each quad-tuple $\mathbf{z}_i^\ell = (\mathbf{h}^{s\ell}_{2i-1}, \mathbf{h}^{s\ell}_{2i}, \mathbf{h}^{t\ell}_{2i-1}, \mathbf{h}^{t\ell}_{2i})$ of the ℓ^{th}-layer hidden representation with respect to the network parameters $\mathcal{F}^\ell = \{\mathbf{W}^\ell, \mathbf{b}^\ell\}$. For each mini-batch $\mathcal{B}$, we go through it to sample the quad-tuples $\mathbf{z}_i^\ell$'s until all source points in $\mathcal{B}$ have been accessed. Specifically, we maintain two lists $\mathcal{B}_s$ and $\mathcal{B}_t$ from $\mathcal{B}$, one for the source points and the other for the target points, and $\mathcal{B} = \mathcal{B}_s \cup \mathcal{B}_t$. For each quad-tuple $\mathbf{z}_i^\ell$, we sample two consecutive points $\{\mathbf{h}^{s\ell}_{2i-1}, \mathbf{z}^{s\ell}_{2i}\}$ from source list $\mathcal{B}_s$, and two consecutive points $\{\mathbf{h}^{t\ell}_{2i-1}, \mathbf{z}^{t\ell}_{2i}\}$ from the target list $\mathcal{B}_t$. When we traverse the source list $\mathcal{B}_s$, if we reach the end of the target list $\mathcal{B}_t$, we will shuffle the target list and go through it again until the source list is run out. With this procedure, we can compute the gradients of linear-time MK-MMD (10) using standard mini-batch SGD.

Finally, for each mini-batch $\mathcal{B}$, we combine the gradients of the empirical error (6), entropy penalty (8) and MK-MMD penalty (7) together. For each layer's parameters $\mathcal{F}^\ell$, this yields

$$\nabla_{\mathcal{F}^\ell}(\mathcal{B}) = \frac{1}{|\mathcal{B}_a|}\sum_{\mathbf{x}_i\in\mathcal{B}_a}\frac{\partial L(\mathbf{x}_i,y_i)}{\partial \mathcal{F}^\ell} + \frac{\gamma}{|\mathcal{B}_u|}\sum_{\mathbf{x}_i\in\mathcal{B}_u}\frac{\partial H(\mathbf{x}_i)}{\partial \mathcal{F}^\ell} + \frac{2\lambda}{|\mathcal{B}_s|}\sum_{\mathbf{z}_i^\ell\in\mathcal{B}}\frac{\partial d_k(\mathbf{z}_i^\ell)}{\partial \mathcal{F}^\ell}. \tag{11}$$

With the resulting mini-batch gradient (11), we can perform each mini-batch update by gradient descent for each layer $\mathcal{F}^\ell \leftarrow \mathcal{F}^\ell - \eta\nabla_{\mathcal{F}^\ell}(\mathcal{B})$, and η is the learning rate. This can be easily implemented in the Caffe framework for CNNs [51]. Given the kernel k is the linear combination of m Gaussian kernels, $\{k_u(\mathbf{x}_i,\mathbf{x}_j) = e^{-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma_u}\}$, the gradient $\frac{\partial d_k(\mathbf{z}_i^\ell)}{\partial \mathcal{F}^\ell}$ can be computed by back-propagation. For example,

$$\frac{\partial k(\mathbf{h}_{2i-1}^{s\ell},\mathbf{h}_{2i}^{t\ell})}{\partial \mathbf{W}^\ell} = \frac{\partial k(\mathbf{h}_{2i-1}^{s\ell},\mathbf{h}_{2i}^{t\ell})}{\partial \boldsymbol{\mathit{h}}_{2i-1}^{s\ell}}\left(\mathbf{h}_{2i-1}^{s(\ell-1)}\right)^\mathsf{T} + \frac{\partial k(\mathbf{h}_{2i-1}^{s\ell},\mathbf{h}_{2i}^{t\ell})}{\partial \boldsymbol{\mathit{h}}_{2i}^{t\ell}}\left(\mathbf{h}_{2i}^{t(\ell-1)}\right)^\mathsf{T}, \tag{12}$$

and the residual terms $\frac{\partial k(\mathbf{h}_{2i-1}^{s\ell},\mathbf{h}_{2i}^{t\ell})}{\partial \boldsymbol{\mathit{h}}_{2i-1}^{s\ell}}$ that back-propagate to influence all the previous layers are further computed by

$$\frac{\partial k(\mathbf{h}_{2i-1}^{s\ell},\mathbf{h}_{2i}^{t\ell})}{\partial \tilde{\mathbf{h}}_{2i-1}^{s\ell}} = -\sum_{u=1}^{m}\frac{2\beta_u}{\sigma_u}k_u(\mathbf{h}_{2i-1}^{s\ell},\mathbf{h}_{2i}^{t\ell}) \times\left[(\mathbf{h}_{2i-1}^{s\ell}-\mathbf{h}_{2i}^{t\ell})\odot\dot{a}^\ell(\boldsymbol{\mathit{h}}_{2i-1}^{s\ell})\right], \tag{13}$$

where $\dot{a}^\ell(\cdot)$ is a derivative of activation function $a^\ell(\cdot)$, $\boldsymbol{\mathit{h}}_i^\ell = \mathbf{W}^\ell\mathbf{h}^{\ell-1}+\mathbf{b}^\ell$ is the layer output before activation.

ME Test. Given J test locations $\mathcal{V} = \{\mathbf{v}_j\}_{j=1}^J \subset \mathbb{R}^J$, the ME test in (4) itself can be computed in linear time. The computation of gradients of ME with respect to network parameters is similar to that of MK-MMD, which is omitted. The J test locations are learned by *maximizing* the ME test as $\max_{\mathcal{V}} n\mathbf{z}_n^\mathsf{T}\mathbf{S}_n^{-1}\mathbf{z}_n$ using gradient ascent, done in each iteration or epoch of the back-propagation. The gradient of ME-test with respect to each test location $\mathbf{v}_j$ is computed as

$$\frac{\partial M_k(\mathcal{D}_s^\ell,\mathcal{D}_t^\ell)}{\partial \mathbf{v}_j} = -\frac{2}{n}\sum_{i=1}^{n}\left(k(\mathbf{h}_i^{s\ell},\mathbf{v}_j)-k(\mathbf{h}_i^{t\ell},\mathbf{v}_j)\right) \times\left(\sum_{i=1}^{n}\sum_{u=1}^{m}\frac{2\beta_u}{\sigma_u}k_u(\mathbf{h}_i^{s\ell},\mathbf{v}_j)\times(\mathbf{h}_i^{s\ell}-\mathbf{v}_j) - \sum_{i=1}^{n}\sum_{u=1}^{m}\frac{2\beta_u}{\sigma_u}k_u(\mathbf{h}_i^{t\ell},\mathbf{v}_j)\times(\mathbf{h}_i^{t\ell}-\mathbf{v}_j)\right). \tag{14}$$

4.2 Learning Kernel Parameters

Theoretically, the optimal kernel parameter $\boldsymbol{\beta}$ for MK-MMD can be learned by minimizing the Type-II error [13]. The proposed multi-layer adaptation regularizer performs layerwise distribution matching by minimizing MK-MMD with respect to the network parameter $\mathcal{F}$, while we seek to learn the optimal kernel parameter $\{\boldsymbol{\beta}_\ell\}$ for all the adaptation layers $\ell\in\mathcal{L}$ by minimizing the Type-II error, equivalent to maximizing a *normalized* variant of MMD [13]

$$\max_{k\in\mathcal{K}} M_k(\mathcal{D}_s^\ell,\mathcal{D}_t^\ell)\boldsymbol{\Sigma}_k^{-2}, \tag{15}$$

where $\boldsymbol{\Sigma}_k^2 = \mathbb{E}_\mathbf{z}d_k(\mathbf{z}) - [\mathbb{E}_\mathbf{z}d_k(\mathbf{z})]^2$ is estimation covariance. Note that when the covariance is isotropic—i.e., $\boldsymbol{\Sigma}_k$ equals to the identity matrix—minimizing the Type-II error reduces to maximizing MK-MMD with respect to kernel parameter $\boldsymbol{\beta}$, which is consistent with Equation (7). For generality, we derive the algorithm based on general covariance structure, which includes (7) as a special case. Note that non-isotropic covariance structure results in smoother kernel parameters.

Denote the m MMDs by $\mathbf{M} = (M_1, M_2, \ldots, M_m)^\mathsf{T}$, where each M_u is the MMD for kernel k_u. The covariance matrix $\mathbf{Q} = \mathrm{cov}(M_k) \in \mathbb{R}^{m\times m}$ is computed in $O(m^2n)$ time by $\mathbf{Q}_{uu'} = \frac{4}{n_s}\sum_{i=1}^{n_s/4} d_{k_u}^\Delta(\mathbf{z}_i)d_{k_{u'}}^\Delta(\mathbf{z}_i)$, where an eight-tuple is $\mathbf{z}_i \triangleq (\mathbf{z}_{2i-1},\mathbf{z}_{2i})$ and $d_{k_u}^\Delta(\mathbf{z}_i) \triangleq d_{k_u}(\mathbf{z}_{2i-1}) - d_{k_u}(\mathbf{z}_{2i})$. Problem (15) can be reduced to a quadratic program (QP),

$$\min_{\mathbf{M}^\mathsf{T}\boldsymbol{\beta}_\ell=1,\boldsymbol{\beta}_\ell\geqslant\mathbf{0}} \boldsymbol{\beta}_\ell^\mathsf{T}(\mathbf{Q}+\varepsilon\mathbf{I})\boldsymbol{\beta}_\ell, \tag{16}$$

where $\varepsilon = 10^{-3}$ is a small penalty to stabilize the problem. This problem can be efficiently solved by standard QP packages. By solving (16), we obtain a multi-kernel $k = \sum_{u=1}^m \beta_u k_u$ that minimizes the Type-II test error. For the ME test, we adopt a fixed kernel combination parameter $\boldsymbol{\beta}$. We note that the DAN objective (9) is essentially a *minimax* problem, i.e., $\min_{f\in\mathcal{F}}\max_{k\in\mathcal{K}} M_k(\mathcal{D}_s^\ell,\mathcal{D}_t^\ell)\boldsymbol{\Sigma}_k^{-2}$. We adopt an alternating optimization that updates $\mathcal{F}$ by mini-batch SGD (11) and $\boldsymbol{\beta}$ by QP (16) iteratively. Both updates cost $O(n)$ and are scalable to large-scale data.

Since the covariance $\boldsymbol{\Sigma}_k$ cannot be estimated accurately with typically small-scale datasets in domain adaptation, we find that maximizing (9) and (15) with $\boldsymbol{\Sigma}_k$ fixed to identity yields similar results, and will report it in our experiments. The CNN parameter $\mathcal{F}$ is learned by minimizing MK-MMD as a domain discrepancy, while the MK-MMD parameter $\boldsymbol{\beta}$ is learned by minimizing the Type-II error. Both criteria are dedicated to an effective adaptation of domain distributions, targeting to consolidate the transferability of DAN features.

4.3 Generalization Error Analysis

We give a sketch of an analysis of a bound on target risk, by making use of the theory of domain adaptation [2], [11], [12] and the theory of kernel embedding of distributions [13], [45], [47].

Theorem 1 ([11], [12]). *Letting $f\in\mathcal{H}$ be a hypothesis, $R_s(f)$ and $R_t(f)$ be the expected risks of source and target respectively, then*

$$R_t(f) \leqslant R_s(f) + d_\mathcal{H}(p,q) + C, \tag{17}$$

where C is the expected risk of ideal hypothesis for both domains, and $d_\mathcal{H}(p,q)$ is the $\mathcal{H}$-divergence between distributions p and q,

$$d_\mathcal{H}(p,q) \triangleq 2\sup_{\eta\in\mathcal{H}}\left|\Pr_{\mathbf{x}^s\sim p}[\eta(\mathbf{x}^s)=1] - \Pr_{\mathbf{x}^t\sim q}[\eta(\mathbf{x}^t)=1]\right|. \tag{18}$$

Hypothesis $\rho\in\mathcal{H}$ is essentially a *two-sample* classifier that discriminates the source and target. The $\mathcal{H}$-divergence largely depends on the capacity of the hypothesis space $\mathcal{H}$ to characterize the discrepancy between distributions p and

q. The hypothesis space $\mathcal{H}$ should be rich enough so that any key difference underlying p and q can be captured. We choose a (kernel) Parzen window classifier as the two-sample classifier ρ [47], which is *nonparametric* and is rich enough to capture the distributional difference. We show that $d_{\mathcal{H}}(p, q)$ can be bounded by the empirical error of the (kernel) Parzen window classifier, which is equivalent to the MK-MMD [47]

$$\begin{aligned} d_{\mathcal{H}}(p,q) &\leqslant \hat{d}_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) + C_{\mathcal{H}} \\ &\leqslant 2 - 2 \inf_{\rho \in \mathcal{H}} \left(\sum_{i=1}^{n_s} \frac{L\big[\rho(\mathbf{x}_i^s) = 1\big]}{n_s} + \sum_{j=1}^{n_t} \frac{L\Big[\rho(\mathbf{x}_j^t) = -1\Big]}{n_t} \right) + C_{\mathcal{H}} \\ &= 2(1 + M_k(\mathcal{D}_s, \mathcal{D}_t)) + C_{\mathcal{H}}, \end{aligned} \tag{19}$$

where $C_{\mathcal{H}}$ is a complexity term that bounds the $\mathcal{H}$-divergence from its empirical estimate, $L(\cdot)$ is the linear loss function of the (kernel) Parzen window classifier ρ, where $L[\rho = 1] \triangleq -\rho, L[\rho = -1] \triangleq \rho$.

The deep features learned by the DAN models to explicitly minimize the MK-MMD can decrease the upper bound of the target-domain risk. Note that we maximize MK-MMD w.r.t. $\boldsymbol{\beta}$ in Equation (15), which helps the Parzen window classifier achieve minimal error for the two-sample discrimination in (19). An important advantage of choosing *nonparametric* Parzen window as the two-sample classifier is that it is almost parameter-free (except a few kernel-combination parameters $\boldsymbol{\beta}$) and no new network parameters are introduced, making the DAN models easy to train. In contrast, domain-adversarial neural network [36] shooses a three-layer perceptron as the domain discriminator, which has more parameters and is not easy to drive to equilibrium.

5 Experiments

We compare our deep adaptation networks to state-of-the-art transfer learning and deep learning methods for unsupervised and semi-supervised domain adaptation. Code and datasets are available at github.com/thuml/xlearn.

5.1 Datasets

We adopt three benchmark datasets: *Office-31*, *Office-Caltech* and *ImageCLEF-DA*. For all deep methods, we use original images as input. For all shallow methods, we use the deep features of AlexNet [22] and GoogLeNet [16].

Office-31 [26] is an open dataset for domain adaptation, consisting of 4,652 images in 31 classes collected from three domains: *Amazon* (**A**), which contains images downloaded from amazon.com, *Webcam* (**W**) and *DSLR* (**D**), which contain images taken by a web camera and a digital SLR camera, respectively, in an office environment. We evaluate all methods across three transfer tasks, $\mathbf{A} \rightarrow \mathbf{W}$, $\mathbf{D} \rightarrow \mathbf{W}$ and $\mathbf{W} \rightarrow \mathbf{D}$, which are commonly adopted in deep learning studies [9], [35], [36], and across the other three transfer tasks $\mathbf{A} \rightarrow \mathbf{D}$, $\mathbf{D} \rightarrow \mathbf{A}$ and $\mathbf{W} \rightarrow \mathbf{A}$ as studied in [29].

Office-Caltech [28], a dataset widely studied in domain adaptation research [31], [52], is built by selecting the 10 common categories shared by *Office-31* and *Caltech-256* (**C**) [53]. We consider all domain combinations and construct 12 transfer tasks: $\mathbf{A} \rightarrow \mathbf{W}$, $\mathbf{D} \rightarrow \mathbf{W}$, $\mathbf{W} \rightarrow \mathbf{D}$, $\mathbf{A} \rightarrow \mathbf{D}$, $\mathbf{D} \rightarrow \mathbf{A}$, $\mathbf{W} \rightarrow \mathbf{A}$, $\mathbf{A} \rightarrow \mathbf{C}$, $\mathbf{W} \rightarrow \mathbf{C}$, $\mathbf{D} \rightarrow \mathbf{C}$, $\mathbf{C} \rightarrow \mathbf{A}$, $\mathbf{C} \rightarrow \mathbf{W}$, and $\mathbf{C} \rightarrow \mathbf{D}$. While *Office-31* has more categories and is more difficult for domain adaptation, *Office-Caltech* comprises more transfer tasks and provides a way to assess dataset bias [54].

ImageCLEF-DA [1] is a benchmark dataset for the ImageCLEF 2014 domain adaptation challenge which is formed by selecting the 12 common categories shared by four public datasets, each considered as a domain: *Caltech-256* (**C**), *ImageNet ILSVRC 2012* (**I**), *Pascal VOC 2012* (**P**), and *Bing* (**B**). We consider all domain combinations and build 12 transfer tasks: $\mathbf{C} \rightarrow \mathbf{I}$, $\mathbf{C} \rightarrow \mathbf{P}$, $\mathbf{C} \rightarrow \mathbf{B}$, $\mathbf{I} \rightarrow \mathbf{C}$, $\mathbf{I} \rightarrow \mathbf{P}$, $\mathbf{I} \rightarrow \mathbf{B}$, $\mathbf{P} \rightarrow \mathbf{C}$, $\mathbf{P} \rightarrow \mathbf{I}$, $\mathbf{P} \rightarrow \mathbf{B}$, $\mathbf{B} \rightarrow \mathbf{C}$, $\mathbf{B} \rightarrow \mathbf{I}$, and $\mathbf{B} \rightarrow \mathbf{P}$. It is worth noting that some domains (e.g., **B**) are low-quality images that are more difficult to categorize than other domains (e.g., **C**). This makes it a good complement to the *Office-31* dataset.

5.2 Experimental Setup

5.2.1 Comparison Methods

We compare with state-of-the-art transfer learning and deep learning methods: Transfer Component Analysis (*TCA*) [3], Geodesic Flow Kernel (*GFK*) [28], Subspace Alignment (*SA*) [55], Deep Domain Confusion (*DDC*) [35], Reverse Gradient (*RevGrad*) and [36]. TCA learns a shared feature space by MMD-penalized Kernel PCA. GFK interpolates across an infinite number of intermediate subspaces to bridge the source and target subspaces. SA seeks a domain invariant feature space by aligning the source subspace with the target subspace. For these three methods, we adopt SVM as a base classifier. DDC maximizes domain confusion by adding an adaptation layer that is regularized by linear-kernel MMD. RevGrad enables domain adversarial learning by adapting a single network layer which matches source and target by making them indistinguishable for a domain discriminator.

We examine the influence of deep representations for domain adaptation, by studying *AlexNet* [22], *GoogLeNet* [16], and *ResNet* [17] as the base architectures. For shallow methods, we follow the DeCAF method [9] and use as deep image representations the activations of the $fc7$ (AlexNet), $in9$ (GoogLeNet) and $pool5$ (ResNet) layers.

We study variants of our models to explore their properties. To study *multi-layer* adaptation, we run DAN with only one adaptation layer, $fc7$ of AlexNet or $in9$ of GoogLeNet, respectively termed *DAN-fc7* and *DAN-in9*. To study *multi-kernel* MMD, we evaluate DAN using single-kernel MMD, termed *DAN-sk*. To study *entropy minimization*, we evaluate DAN by removing the entropy penalty, termed *DAN-ent*. Our preliminary paper [23] is the DAN-ent not using the entropy penalty. All the above variants are based on MMD, while the one based on ME is termed *DAN (ME)*.

5.2.2 Evaluation Protocols

We follow standard evaluation protocols [5], [26]. In the *non-sampling* protocol [5], we utilize all labeled source examples and all unlabeled target examples for unsupervised domain adaptation. For semi-supervised domain adaptation, we further sample three labeled examples per category from the target domain [26]. In the *down-sampling* protocol [26] for *Office-31*, we randomly sample as the source domain 20 labeled examples per category from Amazon (**A**) and 8 labeled

1. http://imageclef.org/2014/adaptation

TABLE 1
Accuracy on *Office-31* Dataset Under Non-Sampling Protocol [5] for Unsupervised and Semi-Supervised Domain Adaptation (AlexNet)

Method	Unsupervised Domain Adaptation							Semi-Supervised Domain Adaptation						
	A → W	D → W	W → D	A → D	D → A	W → A	Avg	A → W	D → W	W → D	A → D	D → A	W → A	Avg
AlexNet [22]	60.6±0.5	95.4±0.3	99.0±0.3	64.2±0.4	45.5±0.6	48.3±0.5	68.8	81.6±0.3	97.0±0.2	99.6±0.2	80.7±0.3	64.3±0.4	64.1±0.3	81.2
GFK [28]	58.4±0.0	93.6±0.0	91.0±0.0	58.6±0.0	**52.4**±0.0	46.1±0.0	66.7	81.0±0.3	94.2±0.2	96.3±0.4	81.0±0.5	65.5±0.4	64.5±0.6	80.4
SA [55]	58.5±0.0	92.0±0.0	98.8±0.0	60.1±0.0	50.1±0.0	47.3±0.0	67.9	80.2±0.6	95.8±0.5	98.8±0.4	78.9±0.3	65.2±0.4	63.3±0.4	80.4
TCA [3]	59.0±0.0	90.2±0.0	88.2±0.0	57.8±0.0	51.6±0.0	47.9±0.0	65.8	82.5±0.3	94.9±0.2	97.3±0.3	80.4±0.4	65.8±0.4	65.1±0.5	81.0
DDC [35]	61.0±0.5	95.0±0.3	98.5±0.3	64.9±0.4	47.2±0.5	49.4±0.6	69.3	85.9±0.4	96.5±0.3	99.2±0.2	80.8±0.4	64.3±0.4	64.5±0.4	81.9
RevGrad [36]	73.0±0.5	96.4±0.3	98.5±0.3	-	-	-	-	-	-	-	-	-	-	-
DAN-fc7	70.2±0.4	96.5±0.2	**99.6**±0.2	71.3±0.3	47.4±0.3	51.3±0.4	72.7	88.4±0.3	97.0±0.1	99.9±0.1	83.5±0.2	65.9±0.3	67.8±0.3	83.8
DAN-sk	70.4±0.7	96.7±0.4	99.6±0.4	70.1±0.5	48.1±0.6	50.2±0.6	72.5	89.5±0.3	97.4±0.2	**100.0**±0.0	83.7±0.4	66.5±0.5	**68.3**±0.4	84.2
DAN-ent [23]	68.5±0.5	96.0±0.3	99.0±0.3	66.8±0.4	50.0±0.5	49.8±0.5	71.7	86.3±0.4	97.2±0.2	99.6±0.2	82.1±0.4	64.6±0.3	65.2±0.3	82.5
DAN	**73.9**±0.5	**96.8**±0.3	**99.6**±0.2	**71.7**±0.4	50.0±0.6	**51.4**±0.6	**73.9**	**89.7**±0.4	**97.5**±0.2	**100.0**±0.0	**84.1**±0.3	**67.8**±0.4	**68.3**±0.4	**84.6**

examples per category from Webcam (**W**) and DSLR (**D**). We compare the average classification accuracy and standard deviation on all random experiments.

For all baseline methods, we follow their original model selection procedures, or conduct *transfer cross-validation* [56] if their model selection strategies are not specified. We can also adopt transfer cross-validation [56] to select parameters λ and γ for DAN, but since DAN performs stably under different parameters, we fix $\lambda = 1$, $\gamma = 0.1$ in all experiments. For MMD-based methods (TCA, DDC, and DAN), we use the Gaussian (RBF) kernel $k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma}$ with the bandwidth σ set to the median pairwise squared distances on training data, i.e., the *median heuristic* [13]. We use multi-kernel MMD and multi-kernel ME for DAN, and a family of m Gaussian kernels $\{k_u\}_{u=1}^{m}$ varying bandwidth $\sigma_u \in [2^{-5}\sigma, 2^5\sigma]$ with a multiplicative step-size of $2^{0.5}$ [13].

We implement all deep methods in *Caffe* [51] and *PyTorch*, and fine-tune from AlexNet [22], GoogLeNet [16] and ResNet [17] models pre-trained on the ImageNet dataset [18]. Since the classifier is trained from scratch, we set its learning rate to be ten times that of the lower layers. We employ mini-batch stochastic gradient descent with momentum of 0.9 and the learning rate strategy implemented in RevGrad [36]: the learning rate is not selected by a grid search due to high computational cost—it is adjusted during SGD using the following formula: $\eta_p = \frac{\eta_0}{(1+\alpha p)^\beta}$, where p is the training progress linearly changing from 0 to 1, $\eta_0 = 0.01, \alpha = 10$ and $\beta = 0.75$, which is optimized to promote convergence and low error on the source domain. To suppress noisy activations at the early stages of training, instead of fixing parameters λ and γ, we gradually change them from 0 to $\lambda = 1, \gamma = 0.1$ by multiplying $\frac{2}{1+\exp(-\delta p)} - 1$, where $\delta = 10$ [36]. This new parameter strategy significantly stabilizes the parameter sensitivity of the proposed models.

5.3 Results and Discussion

We present results for unsupervised adaptation on all three datasets, and for semi-supervised adaptation on *Office-31*.

5.3.1 Results on Office-31 with AlexNet

We compare all methods using AlexNet features as input (i.e., DeCAF features [9]) or using AlexNet as the base architecture. Table 1 reports the classification accuracy results of both unsupervised domain adaptation and semi-supervised domain adaptation under the non-sampling protocol, where the results of RevGrad are directly reported from its original paper [36]. The DAN model outperforms all comparison methods on most transfer tasks, and significantly improves classification accuracy on three out of six transfer tasks. For unsupervised domain adaptation, DAN achieves an absolute accuracy increase of 4.6 percent against the best baseline DDC. For semi-supervised domain adaptation, DAN outperforms the best baseline DDC by an absolute accuracy increase of 2.7 percent. This shows that DAN can learn more transferable features for more effective domain adaptation.

From these experimental results, we make several observations: Traditional transfer-learning methods with deep features as input may either underperform or outperform standard deep-learning methods. Traditional transfer-learning methods may benefit from a domain adaptation procedure while standard deep-learning methods may take advantage of fine-tuning. Transfer learning and deep learning cannot reinforce each other in separated pipelines; instead, we require an end-to-end architecture to unify them. Deep-transfer learning methods generally outperform traditional transfer-learning methods with deep features as input. This confirms that incorporating domain-adaptation modules to deep networks can improve feature transferability. However, different layers of deep networks extract features at different abstraction levels, and thus the extracted features may not transfer safely in multiple layers. By adapting the cross-domain distributions in multiple task-specific layers using optimal multi-kernel two-sample matching and exploiting the low-density separation of the target-unlabeled data, the proposed DAN model establishes a state-of-the-art result on the *Office-31* dataset.

Ablation Study. To explore the properties of DAN, we show the results of its variants: DAN-fc7 (single-layer), DAN-sk (single-kernel), and DAN-ent (no entropy penalty). The results in Table 1 yield several conclusions.

(1) DAN-fc7 achieves better results than DDC, while substantially underperforming DAN. This validates the importance of *multi-layer* adaptation of all task-specific layers for learning transferable features.
(2) DAN-sk achieves better accuracies than DDC, while substantially underperforming DAN. This shows that *multi-kernel* reduces cross-domain discrepancy more effectively than single-kernel. DAN-sk

TABLE 2
Accuracy on *Office-31* Dataset Under Down-Sampling [26] and Non-Sampling [5] Protocols for Unsupervised Domain Adaptation (GoogLeNet)

Method	Down-Sampling Protocol							Non-Sampling Protocol						
	A → W	D → W	W → D	A → D	D → A	W → A	Avg	A → W	D → W	W → D	A → D	D → A	W → A	Avg
GoogLeNet [16]	70.1±0.9	**95.1**±0.5	**98.2**±0.3	71.1±0.5	59.9±0.5	59.4±0.6	75.6	71.4±0.5	95.8±0.2	98.3±0.2	72.2±0.3	61.0±0.3	60.5±0.3	76.5
GFK [28]	63.4±0.6	90.1±0.4	92.6±0.5	65.9±0.6	59.0±0.6	56.8±0.5	71.3	71.2±0.0	96.4±0.0	**99.0**±0.0	70.3±0.0	62.7±0.0	60.7±0.0	76.7
TCA [3]	63.5±0.4	87.1±0.2	92.3±0.4	66.2±0.4	59.2±0.5	58.2±0.5	71.1	68.6±0.0	94.0±0.0	97.4±0.0	69.5±0.0	61.7±0.0	61.4±0.0	75.4
DDC [35]	70.0±0.7	93.8±0.5	96.2±0.4	71.0±0.5	62.8±0.6	61.9±0.6	76.0	72.5±0.4	95.5±0.2	98.1±0.1	73.2±0.3	61.6±0.3	61.6±0.3	77.1
DAN-in9	71.8±0.6	94.0±0.4	96.4±0.3	72.2±0.5	64.2±0.6	63.8±0.6	77.0	74.8±0.3	96.0±0.2	98.5±0.1	74.7±0.2	62.4±0.3	63.8±0.3	78.4
DAN-sk	72.2±0.6	93.5±0.4	97.6±0.3	72.9±0.5	**65.7**±0.6	61.4±0.3	77.2	72.6±0.2	95.8±0.2	99.4±0.1	74.9±0.2	65.9±0.2	63.0±0.3	78.6
DAN-ent [23]	73.2±0.5	94.3±0.4	97.2±0.3	72.9±0.5	64.7±0.5	62.3±0.6	77.4	76.0±0.3	95.9±0.2	98.6±0.1	74.4±0.2	61.5±0.3	60.3±0.2	77.8
DAN	**73.6**±0.7	94.5±0.4	96.6±0.4	**73.5**±0.6	65.4±0.7	**64.4**±0.6	**78.0**	**80.6**±0.3	**96.2**±0.2	98.6±0.1	**75.8**±0.3	**66.2**±0.2	**66.5**±0.3	**80.6**

outperforming DAN-fc7 shows multi-layer adaptation is more important than optimal kernel selection.

(3) DAN-ent achieves higher accuracies than DDC, but underperforms DAN-fc7 and DAN-sk. Hence, low-density separation by entropy minimization may be more important than distribution matching.

(4) The full DAN model achieves the best results. By jointly adapting the feature layers and the classifier layer, we can further improve domain adaptation by reducing the cross-domain discrepancy underlying *both* the marginal distribution of features and conditional distribution of labels given features [6].

Cross-Domain Discrepancy. Theoretical results on domain adaptation [2], [12] show that cross-domain discrepancy plays a key role in bounding the target risk. The *Office-31* dataset embodies cross-domain discrepancy. By construction, the domains **W** and **D** are very similar to each other, while they are significantly dissimilar to domain **A**. The results in Table 1 clearly support the above theory: when the cross-domain discrepancy is small (**W** → **D** and **D** → **W**), the transfer performance is good; when the cross-domain discrepancy is large (**A** → **W** and **A** → **D**), the transfer performance drops significantly.

It is desirable that the DAN model improves the transfer performance more significantly for transfer tasks with larger cross-domain discrepancy (e.g., task **A** → **W**). Another interesting observation is the *asymmetric* property of domain adaptation: the difficulty of transferring from domain **S** to **T** differs from that of **T** to **S**. Transferring from a large domain to a small domain is easier; e.g., **A** → **W** is easier than **W** → **A**. Given the same cross-domain discrepancy between **S** to **T** and **T** to **S**, the risk of a large source domain is lower and results in lower target risk [12].

Semi-Supervised Adaptation. In real applications, a few target-labeled examples may be available. We report semi-supervised domain adaptation results under the non-sampling protocol [26] in Table 1. We conclude that:

(1) Semi-supervised adaptation methods outperform unsupervised adaptation counterparts. This implies that target-labeled examples are much more "valuable" than source-labeled examples.

(2) We should not depend only on the target-labeled examples if they are very scarce, since limited target data cannot induce a reliable target classifier.

(3) The source-labeled data gives low variance but a highly biased prior for the target classifier. Correcting the domain bias by DAN can make a source classifier better adapted to the target domain.

(4) The target-labeled examples boost transfer significantly, by specifying where a good target classifier resides in the source-constrained hypothesis space.

5.3.2 Results on Office-31 with GoogLeNet

We examine all methods using GoogLeNet features as input (similar to the off-the-shelf DeCAF features [9]) or using GoogLeNet as their underlying architecture (DDC and DAN). Table 2 reports unsupervised domain adaptation results on the *Office-31* dataset under both down-sampling [26] and non-sampling [5] protocols. For the down-sampling protocol, DAN outperforms the best baseline DDC by 2.0 percent. For the non-sampling protocol, DAN outperforms the best baseline DDC by 3.5 percent. Compared with the results of AlexNet-based methods in Table 1, we can make several interesting observations.

(1) GoogLeNet-based methods outperform AlexNet-based counterparts by a huge margin (∼6 percent), showing that very deep neural networks can learn higher-quality features than shallower ones.

(2) DAN significantly outperforms GoogLeNet, showing that even very deep networks can only reduce, but not remove, the domain discrepancy.

(3) The boost of DAN (GoogLeNet) over GoogLeNet is less than DAN (AlexNet) over AlexNet, showing that GoogLeNet learns more transferable features.

Market Values of Source Labeled Samples. It is interesting to study how the transfer performance will change with respect to the size of source-labeled examples, which can reflect the "market value" of the source dataset [28], [54]. To this end, Table 2 compares the results of unsupervised domain adaptation using both down-sampling [26] and non-sampling protocols [5]. The average accuracy under the non-sampling protocol is 80.6 percent, which is merely 2.6 percent higher than that under the down-sampling protocol [26]. This implies the "marginal utility" of the source-labeled examples saturates quickly and adding more source-labeled examples can hardly improve the transfer performance. It is worth noting that, by adding only 3 target-labeled examples, the transfer accuracy is dramatically boosted by $\sim 11\%$ (Table 1). This reveals an *intrinsic* limitation of unsupervised domain adaptation for practical

TABLE 3
Accuracy on *Office-31* Dataset Under Non-Sampling [5] Protocol for Unsupervised Domain Adaptation (ResNet)

Method	A → W	D → W	W → D	A → D	D → A	W → A	Avg
ResNet [17]	68.4±0.3	96.7±0.2	99.3±0.1	68.9±0.3	62.5±0.4	60.7±0.4	76.1
RevGrad [36]	82.0±0.5	96.9±0.4	99.1±0.3	79.7±0.2	**68.2**±0.6	67.4±0.5	82.2
DAN	86.3±0.3	**97.2**±0.2	99.6±0.1	**82.1**±0.3	64.6±0.4	65.2±0.3	82.5
DAN (ME)	**89.2**±0.3	96.6±0.2	**100.0**±0.0	82.0±0.3	66.1±0.4	**67.6**±0.4	**83.6**

applications, and calls for semi-supervised domain adaptation for better performance.

5.3.3 Results on Office-31 with ResNet

We evaluate the most competitive method RevGrad using ResNet [17] as the base architecture. We also investigate the new interpretable ME test (4) in this experiment. Table 3 reports unsupervised adaptation results on *Office-31* dataset under the non-sampling [5] protocol. RevGrad and DAN significantly outperform the standard ResNet by large margins ($\sim$ **6**%). This reveals that even when extracting highly abstract features using extremely deep networks, the cross-domain discrepancy still lingers in deep features. This validates our claim that domain adaptation is a very challenging problem, which cannot be solved by deep learning alone.

While DAN performs comparably with RevGrad, DAN (ME) significantly outperforms RevGrad, especially on task **A** → **W**. The ME test can learn interpretable features so as to maximize the distinguishability of the source and target distributions. Taking advantage of the proposed *minimax* game, we adapt domain distributions against the features where their distributions maximally differ. Similar to adversarial learning, the transferability of features can be maximized.

5.3.4 Results on Office-Caltech with AlexNet

These tasks are easier than *Office-31*, so we only report the results with AlexNet. Table 4 lists the results of unsupervised domain adaptation under the non-sampling protocol [5]. DAN outperforms baseline methods on most transfer tasks, and substantially improves the classification accuracy on 8 out of 12 transfer tasks. The average classification accuracy of DAN on all transfer tasks is 92.9 percent, and the absolute accuracy increase is 4.7 percent against the best competitor DDC. By comparing Tables 1, 2, 3, and 4, we find that traditional transfer-learning methods with deep features as input cannot outperform standard deep-learning methods consistently.

5.3.5 Results on ImageCLEF-DA with GoogLeNet

The four domains in *ImageCLEF-DA* are balanced but some domains (e.g., **B**) contain low-quality images much more difficult to categorize. With more difficult transfer tasks, we expect to study if transfer learning helps for harder problems. As the transfer tasks are more difficult, we only report the results with GoogLeNet. Table 5 shows the results of unsupervised domain adaptation under the non-sampling protocol [5]. DAN outperforms all baseline methods on most transfer tasks, and substantially improves the classification accuracy on 7 out of 12 transfer tasks. The average classification accuracy of DAN on all 12 transfer tasks is 79.7 percent, attaining an absolute accuracy improvement of 3.6 percent against the best baseline DDC. This evidence validates that DAN can learn transferable features on hard problems.

Contradiction between theory and results. An interesting result that we reproduce is the *asymmetric* property of domain adaptation: the difficulty of transferring from easy domain **S** to hard domain **T** is significantly lower than transferring from **T** to **S**. For example, based on the quality of images, the difficulty degrees of the four domains are **B** > **P** > **I** > **C**. However, when **B** is used as the target domain, the classification accuracy is much lower than that of when **B** is used as the source domain.

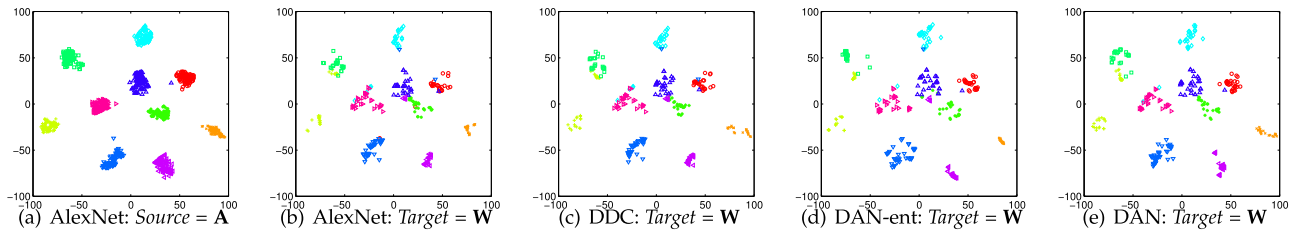
Note that the cross-domain discrepancy (MMD) between a particular source-target pair is the same while the expected risk of an easy source domain may be lower. According to existing domain adaptation theory [2], [12], the expected risk of a hard target domain should be bounded by the cross-domain discrepancy and the expected risk of an easy source domain. Hence it should be lower, but this contradicts with our empirical results. This contradiction points out the need for addressing the asymmetric case with a new theory.

TABLE 4
Classification Accuracy on *Office-Caltech* Dataset Under Non-Sampling Protocol [5] for Unsupervised Domain Adaptation (AlexNet)

Method	A → W	D → W	W → D	A → D	D → A	W → A	A → C	W → C	D → C	C → A	C → W	C → D	Avg
AlexNet [22]	83.1±0.2	97.7±0.2	**100.0**±0.0	88.5±0.3	89.3±0.2	83.8±0.3	84.6±0.2	77.7±0.3	80.9±0.3	91.8±0.2	83.1±0.2	89.0±0.2	87.5
GFK [28]	89.5±0.0	97.0±0.0	98.1±0.0	86.0±0.0	89.8±0.0	88.5±0.0	76.2±0.0	77.1±0.0	77.9±0.0	90.7±0.0	78.0±0.0	77.1±0.0	85.5
SA [55]	81.7±0.0	97.0±0.0	**100.0**±0.0	87.3±0.0	84.6±0.0	82.1±0.0	82.1±0.0	74.1±0.0	76.6±0.0	92.1±0.0	84.1±0.0	86.6±0.0	85.7
TCA [3]	84.4±0.0	96.9±0.0	99.4±0.0	82.8±0.0	90.4±0.0	85.6±0.0	81.2±0.0	75.5±0.0	79.6±0.0	92.1±0.0	88.1±0.0	87.9±0.0	87.0
DDC [35]	86.1±0.3	98.2±0.1	**100.0**±0.0	89.0±0.2	89.5±0.2	84.9±0.3	85.0±0.2	78.0±0.3	81.1±0.3	91.9±0.2	85.4±0.2	88.8±0.3	88.2
DAN-fc7	92.2±0.2	98.9±0.1	**100.0**±0.0	90.3±0.2	92.6±0.1	88.8±0.2	86.3±0.3	83.7±0.3	82.3±0.2	93.1±0.2	94.3±0.1	90.5±0.2	91.1
DAN-sk	94.2±0.2	98.9±0.1	**100.0**±0.0	91.7±0.2	94.0±0.1	92.7±0.2	86.2±0.3	83.0±0.3	81.9±0.3	92.4±0.2	95.0±0.2	90.5±0.2	91.7
DAN-ent [23]	93.8±0.2	**99.0**±0.1	**100.0**±0.0	92.4±0.2	92.0±0.2	92.1±0.2	85.1±0.3	84.3±0.3	**82.4**±0.4	92.0±0.2	90.6±0.2	90.5±0.2	91.2
DAN	**96.1**±0.1	**99.0**±0.1	**100.0**±0.0	**92.8**±0.2	**94.3**±0.2	**93.4**±0.2	88.0±0.3	**87.3**±0.3	**82.4**±0.3	**93.5**±0.2	**96.3**±0.1	**91.4**±0.3	**92.9**

TABLE 5
Classification Accuracy on *ImageCLEF-DA* Dataset Under Non-Sampling Protocol [5] for Unsupervised Domain Adaptation (GoogLeNet)

Method	C → I	C → P	C → B	I → C	I → P	I → B	P → C	P → I	P → B	B → C	B → I	B → P	Avg
GoogLeNet [16]	85.2±0.2	66.8±0.3	58.6±0.3	92.2±0.2	76.5±0.3	57.2±0.3	91.3±0.2	85.0±0.3	**58.7**±0.3	87.0±0.2	81.3±0.3	65.2±0.3	75.4
GFK [28]	84.2±0.0	69.8±0.0	58.3±0.0	92.5±0.0	75.8±0.0	57.3±0.0	82.2±0.0	78.7±0.0	48.8±0.0	81.2±0.0	73.5±0.0	61.5±0.0	72.0
TCA [3]	83.2±0.0	69.8±0.0	58.8±0.0	93.7±0.0	76.0±0.0	60.2±0.0	93.9±0.0	84.0±0.0	56.8±0.0	87.0±0.0	80.3±0.0	67.0±0.0	75.9
DDC [35]	86.3±0.2	71.5±0.3	58.9±0.3	92.8±0.2	77.7±0.3	59.6±0.3	89.9±0.3	83.7±0.2	55.5±0.3	87.6±0.2	80.4±0.2	69.1±0.3	76.1
DAN-in9	90.5±0.2	**73.6**±0.3	**62.4**±0.2	**97.0**±0.2	78.5±0.3	60.8±0.3	92.6±0.2	86.2±0.2	56.6±0.3	91.2±0.2	84.0±0.2	73.5±0.2	78.9
DAN-sk	89.5±0.2	71.8±0.2	60.5±0.2	96.0±0.2	76.4±0.3	51.2±0.3	**95.3**±0.2	**89.3**±0.2	55.4±0.3	92.6±0.2	**85.6**±0.2	72.7±0.2	78.0
DAN-ent [23]	90.0±0.1	73.5±0.3	60.8±0.3	96.1±0.1	78.9±0.2	58.6±0.2	92.7±0.2	84.7±0.3	56.3±0.2	90.0±0.3	82.4±0.2	**73.7**±0.3	78.1
DAN	**91.2**±0.2	72.3±0.2	61.3±0.2	96.6±0.1	**79.3**±0.2	**61.5**±0.3	95.0±0.1	89.2±0.2	58.3±0.2	**93.8**±0.2	85.2±0.3	73.3±0.2	**79.7**



(a) AlexNet: *Source* = **A** (b) AlexNet: *Target* = **W** (c) DDC: *Target* = **W** (d) DAN-ent: *Target* = **W** (e) DAN: *Target* = **W**

Fig. 2. Visualization analysis: (a) t-SNE of source features by AlexNet; (b)–(e) t-SNE of target features by AlexNet, DDC, DAN-ent, and DAN.

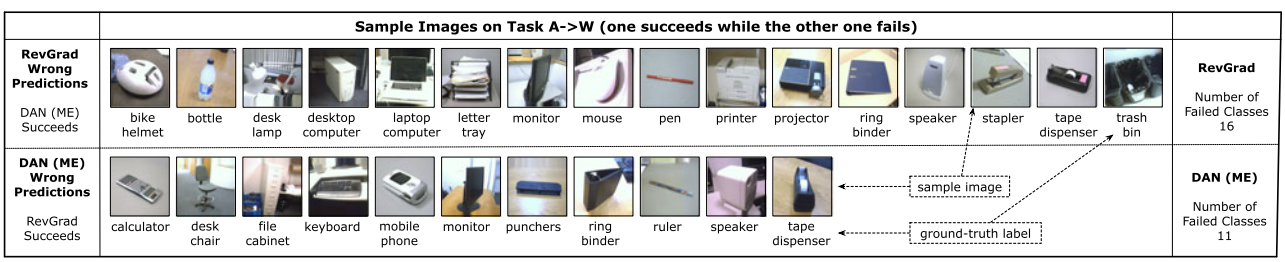


Fig. 3. Sample images where our method succeeds while the baseline method fails or vice versa.

5.4 Empirical Analysis

5.4.1 Feature Visualization

We demonstrate the feature transferability by visualizing in Figs. 2a, 2b, 2c, 2d, and 2e the t-SNE embeddings [9] of the images in transfer task **A** → **W** with the deep features by AlexNet (**A** & **W**), DDC (**W**), DAN-ent (**W**), and DAN (**W**), respectively. We make the following observations. The source and target are not aligned well with AlexNet, and are better aligned with DDC but categories are not discriminated well. And they are aligned better and categories are discriminated better by DAN-ent, while DAN is clearly better than DAN-ent. This shows the benefit of learning transferable features through deep adaptation of multiple layers, optimal multi-kernel distribution matching and entropy minimization.

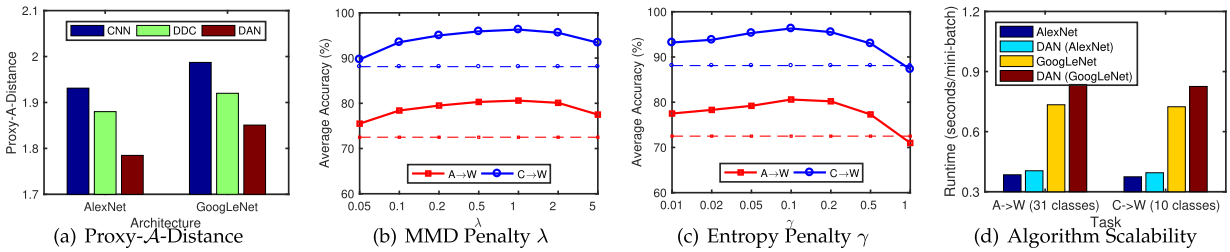
5.4.2 Case Study

To enable a more concrete understanding about which kind of errors the algorithms are making, we present a case study that shows sample images where our method DAN (ME) succeeds while the strongest competitor RevGrad fails and vice versa. The sample images along with their ground truth labels are illustrated in Fig. 3, where the images classified simultaneously right or wrong by both methods are omitted. It is interesting that the number of classes that RevGrad fails but DAN (ME) succeeds is 16, while in the other direction this number is only 11. This shows that DAN (ME) performs more stably than RevGrad under diverse variations in image classes. Note that most failure predictions are made for images of complex backgrounds or of similar appearances between classes (e.g., "stapler" and "tape dispenser").

5.4.3 Proxy-A-Distance

The theory of domain adaptation [2], [12] suggests the $\mathcal{A}$-distance as a measure of cross-domain discrepancy, which, together with the source risk, will bound the target risk. As computing the exact $\mathcal{A}$-distance is intractable, the proxy $\mathcal{A}$-distance (PAD) is defined as $\hat{d}_{\mathcal{A}} = 2(1 - 2\epsilon)$, where ϵ is the generalization error of a two-sample classifier (kernel SVM) trained on the binary problem of distinguishing the input samples between the source and target domains.

Fig. 4a shows the PADs on task **A** → **W** (10 classes) with features of CNN, DDC and DAN based respectively on AlexNet (left bars) and GoogLeNet (right bars). Since deep features can be discriminative for classifying categories and domains [8], they may hurt domain adaptation if enlarging the cross-domain discrepancy [12]. It is desirable that the PADs on DDC and DAN features are much smaller than those on CNN features, which guarantees more transferable features. Counterintuitively, the PADs of GoogLeNet-based methods (22 layers) are even larger than those of AlexNet-based methods (8 layers). This confirms that the features of very deep networks are also not safely transferable.



(a) Proxy-$\mathcal{A}$-Distance (b) MMD Penalty λ (c) Entropy Penalty γ (d) Algorithm Scalability

Fig. 4. Empirical analysis: (a) Proxy-$\mathcal{A}$-Distance of different features; (b)–(c) Sensitivity of λ and γ (dashed lines show best baselines); (d) Scalability.

5.4.4 Parameter Sensitivity

The DAN models have two hyper-parameters for the MMD penalty λ and entropy penalty γ. We conduct parameter sensitivity experiment on transfer tasks $\mathbf{A} \rightarrow \mathbf{W}$ (31 classes) and $\mathbf{C} \rightarrow \mathbf{W}$ (10 classes). When testing a specific parameter, we fix the other parameters and vary the parameter of interest in $\{0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 5\}$. The results are shown in Figs. 4b and 4c, with the best results of the baseline methods shown as dashed lines. One can see that the accuracy of DAN first increases and then decreases as hyper-parameters λ, γ vary and shows stable accuracy curves, which consistently outperform the best baseline.

5.4.5 Algorithm Scalability

Finally, we empirically verify that the proposed models can scale linearly to large sample size and achieve comparable computational cost with the standard deep networks such as AlexNet and GoogLeNet. Fig. 4d shows that the running times of DAN are roughly $1.2\times$ that of AlexNet and GoogLeNet, respectively. This highlights the potential value of applying DAN in large domain-adaptation applications.

6 Conclusions

We have presented a novel learning framework for deep adaptation networks to enhance feature transferability from all task-specific layers of deep convolutional networks. We confirm that while general features can generalize well to a novel target task, specific features tailored to an original task cannot bridge the domain discrepancy effectively. We show that the feature transferability can be enhanced substantially by matching kernel embeddings of multi-layer representations across domains in reproducing kernel Hilbert spaces. The very deep architectures and the low-density separation of target-unlabeled data substantially enhance the feature transferability, while an optimal multi-kernel learning strategy improve the effectiveness of distribution matching. A new interpretable test statistic with distinguishable test locations leads to a nonparametric minimax game, enabling adversarial learning of more transferable features. The unbiased estimate of the kernel embedding establishes an efficient linear-time algorithm implemented through standard back-propagation. Comprehensive empirical evaluation on standard domain adaptation datasets validates the efficacy of the deep adaptation networks against the state-of-the-art.

Acknowledgments

This work was supported by National Key R&D Program of China (No. 2016YFB1000701), the National Natural Science Foundation of China (No. 61772299, 71690231, 61502265) and the DARPA Program on Lifelong Learning Machines.

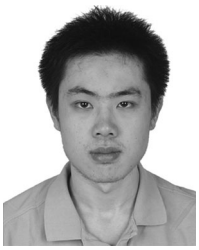
References

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[2] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. Conf. Learn. Theory*, 2009.
[3] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw. Learn. Sys.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.
[4] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 2200–2207.
[5] B. Gong, K. Grauman, and F. Sha, "Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. I-222–I-230.
[6] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang, "Domain adaptation under target and conditional shift," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. III-819–III-827.
[7] X. Wang and J. Schneider, "Flexible transfer learning under support and model shift," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 1898–1906.
[8] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 513–520.
[9] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. I-647–I-655.
[10] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.
[11] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 137–144.
[12] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, no. 1/2, pp. 151–175, 2010.
[13] A. Gretton, B. Sriperumbudur, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, and K. Fukumizu, "Optimal kernel choice for large-scale two-sample tests," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1205–1213.
[14] K. P. Chwialkowski, A. Ramdas, D. Sejdinovic, and A. Gretton, "Fast two-sample testing with analytic representations of probability measures," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 1981–1989.
[15] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2004, pp. 529–536.
[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.
[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[19] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1717–1724.
[20] J. Hoffman, S. Guadarrama, E. Tzeng, R. Hu, J. Donahue, R. Girshick, T. Darrell, and K. Saenko, "LSDA: Large scale detection through adaptation," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 3536–3544.
[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.
[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[23] M. Long, J. Wang, Y. Cao, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 97–105.
[24] W. Li, L. Duan, D. Xu, and I. W. Tsang, "Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1134–1148, Jun. 2014.
[25] L. Duan, I. W. Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 465–479, Mar. 2012.
[26] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 213–226.
[27] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 999–1006.
[28] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2066–2073.
[29] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Simultaneous deep transfer across domains and tasks," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 4068–4076.
[30] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.
[31] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2058–2065.
[32] H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, and M. Marchand, "Domain-adversarial neural networks," in *Proc. NIPS Workshop Transfer Multi-task Learn.: Theory Meets Practice*, 2014.
[33] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[34] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 689–696.
[35] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," arXiv preprint arXiv:1412.3474, 2014.
[36] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.
[37] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2962–2971.
[38] M. Long, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 2208–2217.
[39] Z. Luo, Y. Zou, J. Hoffman, and L. F. Fei-Fei, "Label efficient learning of transferable representations acrosss domains and tasks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 165–177.
[40] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[41] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 469–477.
[42] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 95–104.
[43] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2242–2251.
[44] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.
[45] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.
[46] W. Jitkrittum, Z. Szabó, K. P. Chwialkowski, and A. Gretton, "Interpretable distribution features with maximum testing power," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 181–189.
[47] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, G. Lanckriet, and B. Schölkopf, "Kernel choice and classifiability for RKHS embeddings of probability distributions," in *Proc. 22nd Int. Conf. Neural Inf. Process. Syst.*, 2009, pp. 1750–1758.
[48] D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu, "Equivalence of distance-based and RKHS-based statistics in hypothesis testing," *The Ann. Statist.*, vol. 41, no. 5, pp. 2263–2291, 2013.
[49] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," arXiv preprint arXiv:1207.0580, 2012.
[50] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised Learning*. Cambridge, MA, USA: MIT Press, 2006.
[51] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.
[52] M. Long, J. Wang, J. Sun, and P. S. Yu, "Domain invariant transfer kernel learning," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 6, pp. 1519–1532, Jun. 2015.
[53] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology, 7694, Pasadena, CA, 2007.
[54] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 1521–1528.
[55] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2960–2967.
[56] M. Sugiyama, M. Krauledat, and K.-R. Muller, "Covariate shift adaptation by importance weighted cross validation," *J. Mach. Learn. Res.*, vol. 8, no. May, pp. 985–1005, 2007.

Mingsheng Long received the BE degree in electrical engineering and the PhD degree in computer science, in 2008 and 2014 respectively, both from Tsinghua University. He is an assistant professor with the School of Software, Tsinghua University. He was a visiting researcher with the AMPLab, UC Berkeley from 2014 to 2015. His research interests include machine learning, computer vision, deep learning, and transfer learning.

Yue Cao received the BE degree in computer software from Tsinghua University, China, in 2014. He is working toward the PhD degree in computer software at Tsinghua University. His research interests include computer vision and machine learning.



Zhangjie Cao is working toward the BE degree in computer software at Tsinghua University, China. His research interests include computer vision and machine learning.

Jianmin Wang received the graduate degree from Peking University, China, in 1990, and the ME and PhD degrees in computer software from Tsinghua University, China, in 1992 and 1995, respectively. He is a full professor with the School of Software, Tsinghua University. His research interests include big data management systems and large-scale data analytics. He led to develop a product data & lifecycle management system, which has been deployed in hundreds of enterprises in China. He is leading to develop a big data management system in the National Engineering Lab for Big Data Software.

Michael I. Jordan is the Pehong Chen distinguished professor with the Department of Electrical Engineering and Computer Science and the Department of Statistics, University of California, Berkeley. His research interests bridge the computational, statistical, cognitive and biological sciences, and have focused in recent years on Bayesian nonparametric analysis, probabilistic graphical models, spectral methods, kernel machines and applications to problems in distributed computing systems, natural language processing, signal processing and statistical genetics. He is a member of the National Academy of Sciences, a member of the National Academy of Engineering and a member of the American Academy of Arts and Sciences. He received the David E. Rumelhart Prize in 2015 and the ACM/AAAI Allen Newell Award in 2009. He is a fellow of the AAAI, ACM, ASA, CSS, IEEE, IMS, ISBA and SIAM.