Deep Domain Fusion for Adaptive Image Classification

by

Andrew Dudley

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved July 2019 by the
Graduate Supervisory Committee:

Sethuraman Panchanathan, Chair
Hemanth Venkateswara
Troy McDaniel

ARIZONA STATE UNIVERSITY

August 2019

ABSTRACT

Endowing machines with the ability to understand digital images is a critical task for a host of high-impact applications, including pathology detection in radiographic imaging, autonomous vehicles, and assistive technology for the visually impaired. Computer vision systems rely on large corpora of annotated data in order to train task-specific visual recognition models. Despite significant advances made over the past decade, the fact remains collecting and annotating the data needed to successfully train a model is a prohibitively expensive endeavor. Moreover, these models are prone to rapid performance degradation when applied to data sampled from a different domain. Recent works in the development of deep adaptation networks seek to overcome these challenges by facilitating transfer learning between source and target domains. In parallel, the unification of dominant semi-supervised learning techniques has illustrated unprecedented potential for utilizing unlabeled data to train classification models in defiance of discouragingly meager sets of annotated data.

In this thesis, a novel domain adaptation algorithm – Domain Adaptive Fusion (DAF) – is proposed, which encourages a domain-invariant linear relationship between the pixel-space of different domains and the prediction-space while being trained under a domain adversarial signal. The thoughtful combination of key components in unsupervised domain adaptation and semi-supervised learning enable DAF to effectively bridge the gap between source and target domains. Experiments performed on computer vision benchmark datasets for domain adaptation endorse the efficacy of this hybrid approach, outperforming all of the baseline architectures on most of the transfer tasks.

Angela,

For all the mountains we will climb,

this one I dedicate to you.

ACKNOWLEDGMENTS

The list of individuals to which I owe the utmost gratitude for supporting me on the path to completing this thesis couldn't possibly be contained within a single page. To every person that I have had the privilege of learning from on this fascinating pursuit of knowledge, please know that you have helped mold me into who I am today, and I am eternally indebted to you.

Dr. Panch, you gave me a home when I was lost, the opportunity to succeed, and even in the final days have returned to me a confidence that was long forgotten. Thank you for founding CUbiC, and for taking the role as my graduate advisor.

Dr. Venkateswara and Dr. McDaniel, you've shown unwaivering faith in me to triumph over even the most daunting of challenges. Thank you for your endless expertise and advise.

Within the walls of CUbiC and the Yochan Laboratory, I have had the pleasure of working side-by-side with more relentlessly driven and intelligent individuals than I could ever have hoped to meet in five lifetimes;
Tathagata, Sarath, Sailik, Sachin, Gabe, Lydia, Yantian, Ram, Aditya, and Anagha;
Badri, Meredith, Bijan, Bryan, and Abhik;
Your hard work and dedication will never cease to inspire.

ASU and CIDSE, you have given me more than I ever expected to find on this academic journey;
For my role as a TA: Dr. Doupe, Dr. Kambhampati, Dr. Dougherty;
For unparalleled academic advising: Allison and Christina;
Thank you for everything!

To my father for setting the example of everything I hope to be, my mother for her eternal love and support, and my sister for leading the charge back into academia, I love you, I appreciate you, and I am so proud to call you my family.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Modern advances in machine learning have ignited imaginations around the world with visions of superhuman artificial intelligence seemingly just around the corner. With the latest technology enabling computers to drive cars, defeat grand champions at their own games, model languages, write articles, produce music, and so much more, these high expectations of the imminent future may not appear entirely unwarranted. However, with human intellect as the benchmark, it is necessary to appreciate the abilities that humans possess and which machines must be able to emulate in order to make these sci-fi fantasies become a reality. In particular, the human brain's ability adapt and transfer knowledge across domains is a fundamental feature of human intelligence, and has proven to be particularly challenging to replicate in silicon and software.

This concept of domain adaptation can be elucidated by example. Imagine a pathology diagnosis system developed to identify disease and injury in chest x-ray images. The model employed by this application was trained on hundreds of thousands of images taken from an x-ray machine at the local hospital. With access to expert annotations provided by a staff of radiologists, the system has learned to effectively identify an array of common diseases. Impressed by these results, another hospital is quick to adopt the diagnosis system in hopes of reducing their backlog of x-rays. However, after a series of tests using images taken at their location, they conclude that the predicted pathologies are not reliable. The system's inability to produce the same performance at the new hospital can be attributed a *domain gap* that exists between the images produced by the two x-ray machines. Variation in the x-ray tube voltage, display window, and dynamic range of the machine can impact the appearance of the imaging, and a difference in patient demographics will result in a different distributions of physical characteristics and pathologies captured in the images. In order to improve the performance, the system could be retrained on a dataset of x-rays from the new hospital, but doing so would also require ground-truth pathology annotations for that dataset, which are prohibitively expensive and time consuming to collect.

Domain adaptation algorithms seek to resolve these issues of adapting models from a source domain to a target domain by reducing the distribution discrepancy of the domains. This is most often accomplished without the need of collecting any additional labels by using both the labeled data from the source domain *and* the unlabeled data from the target domain to improve the model performance.

The goal of this thesis is to explore new methods of unsupervised domain adaptation (UDA) in computer vision. With an abundance of raw visual data now available through a host of internet services, the lack of labels to use this data in a supervised training setting has become one of the key obstacles to fully realizing the data's potential for training state-of-the-art classification models. UDA algorithms present an opportunity to sidestep this obstacle entirely. The ubiquity of the domain gap problem when training with limited datasets is a testament to the potential impact and importance of finding effective adaptation solutions.

1.1 Contributions

The contributions of this thesis are as follows:

1. We assert that the successful alignment of two domains effectively reduces an unsupervised domain adaptation problem to a problem of semi-supervised learning.

2. We propose a data augmentation based regularization technique that constrains a classification function to exhibit cross-domain linear behavior between convex combinations of input images and their resulting predictions.

3. A novel domain adaptation architecture – Domain Adaptive Fusion – is proposed that combines our regularization technique with dominant algorithms from domain adaptation and semi-supervised learning research into a single neural network that can be trained end-to-end. The performance of this model is evaluated against other domain adaptation methods on a series of domain adaptive transfer tasks.

1.2 Thesis Outline

Chapter 2 provides an overview of domain adaptation. We formally define the problems of transfer learning and domain adaptation, as well as the notation used throughout this thesis. We then outline the various settings studied within domain adaption.

Chapter 3 is a literature review of modern algorithms and architectures most relevant to the contributions of this thesis. The first section focuses on research in domain adaptive methods, followed by a section on the most prominent approaches to semi-supervised learning.

Chapter 4 describes our proposed Domain Adaptive Fusion (DAF) architecture.

Chapter 5 details the experimental setup. We explain the datasets used and enumerate the transfer tasks that they contain. Implementation details of the Domain Adaptive Fusion architecture are provided, including the selection of hyperparameters.

Chapter 6 contains the results and analysis of our experiments. In the first section, performance results on transfer tasks from both *Office-31* and *Office-Home* datasets are compared against leading baseline models. The sections that follow provide visualizations and discussion on the prediction error in our experiments in the form of confusion matrices, t-SNE plots to analyze the domain invariance and clustering of extracted features, and 3D plots of prediction vectors generated by the baseline and DAF models to inspect the impact of the contributed regularization technique.

Chapter 7 concludes the thesis with a summary of the contributions of this work, and lists several ideas for future work that can be explored based these contributions.

Chapter 2

DOMAIN ADAPTATION

In the previous chapter, domain adaptation is motivated through example, presenting the ubiquity of the problems that the work of this thesis aims to solve. In this chapter, a formal approach is taken to precisely define domain adaptation in the landscape of transfer learning problems. The first section provides the notation and definitions of key components required to define transfer learning following the notation provided by Pan and Yang (2009) and Venkateswara (2017). In the second section, this notation is then used to illustrate the landscape of various domain adaptation settings, as well as to highlight the relationship between unsupervised domain adaptation (UDA) and semi-supervised learning (SSL).

2.1 Domains and Tasks

Transfer learning problems involve *domains* and *tasks* (Pan and Yang (2009)). A domain $\mathcal{D}$ is defined as $\mathcal{D} = \{\mathcal{X}, \mathcal{P}(\mathcal{X})\}$, where $\mathcal{X}$ is a feature space and $\mathcal{P}(\mathcal{X})$ is the marginal probability distribution that governs that feature space. Two domains are then said to be different if either their feature spaces or their probability distributions (or both) are different. Given a domain $\mathcal{D}$, a task $\mathcal{T}$ under that domain is then defined as $\mathcal{T} = \{\mathcal{Y}, f(.)\}$, where $\mathcal{Y}$ is the label space and $f(.)$ is a function $f : \mathcal{X} \to \mathcal{Y}$. The function f is generally unknown, and in a supervised setting is learned using a set of training data pairs $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$ where $\boldsymbol{x}_i \in \mathcal{X}$ and $\boldsymbol{y}_i \in \mathcal{Y}$. Once learned, the label of a new data point $\boldsymbol{x}$ can be predicted using the value returned by $f(x)$, which can be seen as the posterior probability $p(\boldsymbol{y}|\boldsymbol{x})$. In some scenarios, it may also be useful to define a domain $\mathcal{D}$ as a joint space of the features and the labels and their joint probability distribution such that $\mathcal{D} = \{(\mathcal{X} \times \mathcal{Y}), P(\mathcal{X}, \mathcal{Y})\}$ (Venkateswara (2017)).

Domain adaptation problems generally consist of two domains: a *source* and a *target*. With a slight abuse of notation, a source dataset can be represented as a collection of data points $\mathcal{D}_s = \{(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s)\}_{i=1}^{n_s}$, where $\boldsymbol{x}_i^s \in \mathcal{X}_s$ and $\boldsymbol{y}_i^s \in \mathcal{Y}_s$. Similarly, a target dataset is represented as $\mathcal{D}_t = \{(\boldsymbol{x}_i^t, \boldsymbol{y}_i^t)\}_{i=1}^{n_t}$, where $\boldsymbol{x}_i^t \in \mathcal{X}_t$ and $\boldsymbol{y}_i^t \in \mathcal{Y}_t$.

Definition 1 ***Transfer Learning:*** *(Pan and Yang (2009)) Given a source domain $\mathcal{D}_s$ with task $\mathcal{T}_s$ and a target domain $\mathcal{D}_t$ with task $\mathcal{T}_t$, a transfer learning algorithm is an algorithm that seeks to improve the performance of the target predictive function $f_t(.)$ by utilizing $\mathcal{D}_s$ and $\mathcal{T}_s$, where $\mathcal{D}_s \neq \mathcal{D}_t$ or $\mathcal{T}_s \neq \mathcal{T}_t$.*

2.2 Domain Adaptation

Using this definition of transfer learning, standard domain adaptation can then be seen as the case of transfer learning where the source and target domains are different ($\mathcal{D}_s \neq \mathcal{D}_t$), but they share the same task ($\mathcal{T}_s = \mathcal{T}_t$). The difference between the domains can be modeled as the divergence of their joint probability distributions $\mathcal{P}_s(\mathcal{X}, \mathcal{Y}) \neq \mathcal{P}_t(\mathcal{X}, \mathcal{Y})$, and the key task of domain adaptation is thus to estimate $\hat{\mathcal{P}}_t(\mathcal{X}, \mathcal{Y})$ using the learned distribution $\hat{\mathcal{P}}_s(\mathcal{X}, \mathcal{Y})$.

This setting of domain adaptation implies that not only do the domains share the same label space ($\mathcal{Y}_s = \mathcal{Y}_t$, an assumption more specifically called *closed-set* domain adaptation), but that their posterior probabilities are also similar $\mathcal{P}_s(\mathcal{Y}|\mathcal{X}) = \mathcal{P}_t(\mathcal{Y}|\mathcal{X})$. Keeping in mind that a joint probability distribution $\mathcal{P}(\mathcal{X}, \mathcal{Y}) = P(\mathcal{Y}|\mathcal{X})\mathcal{P}(\mathcal{X})$, it is the *co-variate shift* of the marginal distributions $\mathcal{P}_s(\mathcal{X}) \neq \mathcal{P}_t(\mathcal{X})$ that must be bridged in order to successfully adapt across the domains.

2.2.1 Unsupervised Domain Adaptation

The most popular domain adaptation scenario assumes that the labels for the source domain are available, but the target labels are not, resulting in the datasets $\mathcal{D}_s = \{(\boldsymbol{x}_1^s, \boldsymbol{y}_1^s), \ldots, (\boldsymbol{x}_{n_s}^s, \boldsymbol{y}_{n_s}^s)\}$ and $\mathcal{D}_t = \{\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_{n_t}^t\}$. This scenario receives a lot of attention in computer vision research, as there are many modern services that make it easy to collect unlabeled data from their domains. Unsupervised domain adaptation shares many similarities with traditional semi-supervised learning:

Definition 2 ***Semi-Supervised Learning:*** *(Chapelle* et al. *(2010)) This machine learning paradigm consists of a training set* $\mathbf{X}_l = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$ *and an unlabeled training set* $\mathbf{X}_u = \{\boldsymbol{x}_i\}_{i=1}^n$*. The goal of semi-supervised learning is to use all of the training data available to learn a model that can either predict the labels for the entire feature space (as in inductive learning), or only predict the labels of the unlabeled data* $\mathbf{X}_u$ *(as in* transductive *learning).*

By setting $\mathcal{D}_s = \mathbf{X}_l$ and $\mathcal{D}_t = \mathbf{X}_u$, it may even seem that these two areas of machine learning are identical. However, there is an important distinction to be made. In semi-supervised learning, there is a fundamental assumption that all of the data from both the labeled and unlabeled dataset are drawn from the same probability distribution, and thus the marginal distributions $\mathcal{P}_s(\mathcal{X})$ and $\mathcal{P}_t(\mathcal{X})$ must be the same. By bridging the the differences of these distributions, algorithms developed for semi-supervised learning become natural candidates for solving unsupervised domain adaptation problems.

2.2.2 Semi-Supervised Domain Adaptation

Despite the popularity of unsupervised domain adaptation algorithms, they do not consider the fact that it may sometimes be inexpensive to get labels for a subset of the target dataset, and that even a few labeled samples in the target domain could be leveraged in the learning algorithm to improve a model's performance. Semi-supervised domain adaptation is used to address these problems where there is again a labeled source domain $\mathcal{D}_s = \{(\boldsymbol{x}_1^s, \boldsymbol{y}_1^s), \ldots, (\boldsymbol{x}_{n_s}^s, \boldsymbol{y}_{n_s}^s)\}$, but the target domain $\mathcal{D}_t$ contains both an unlabeled collection of samples $\mathcal{D}_t^u = \{\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_{n_u}^t\}$ and a relatively small labeled collection of samples $\mathcal{D}_t^l = \{(\boldsymbol{x}_{n_u+1}^t, \boldsymbol{y}_{n_u+1}^t), \ldots, (\boldsymbol{x}_{n_t}^t, \boldsymbol{y}_{n_t}^t)\}$, such that $\mathcal{D}_t = \mathcal{D}_t^u \cup \mathcal{D}_t^l$, but there are not enough labeled samples in the target domain to directly estimate $\hat{\mathcal{P}}_t(\mathcal{X}, \mathcal{Y})$. Exploration of this setting can be seen in **??** and **??**.

2.2.3 Partial Domain Adaptation

In partial domain adaptation, proposed by Cao *et al.* (2018), the equal label space assumption is relaxed, allowing the target label space to be a subset of the source label space ($\mathcal{Y}_t \subset \mathcal{Y}_s$). This relaxation introduces new challenges, as standard domain adaptation algorithms that attempt to match the target data to all of the source data (including the data whose labels aren't in the shared label space) are highly susceptible to negative transfer. Partial domain adaptation is generally viewed in an unsupervised domain adaptation setting, where none of the target labels are available, and thus the target label space is unknown. Successful solutions to partial domain adaptation problems must then not only ameliorate the domain gap between the domains, but also account for the new *category gap* between the label spaces.

2.2.4 Open-set and Universal Domain Adaptation

The open-set and universal recognition settings remove the equal label space assumption entirely, such that both the source and target label spaces may include categories not shared by the domains. Like partial domain adaptation, these modifications to the standard domain adaptation setting further increase the challenge of the training task. *Open-set* domain adaptation approaches assume that the shared label space is known, and then seek to train the model to throw away the "unknown" classes while learning to correctly classify data samples in the target domain whose labels are from the shared label space Panareda Busto and Gall (2017). Universal recognition then tackles the domain adaptation problems where the label spaces are unrestricted *and* the shared label space between the domains is unknown You *et al.* (2019).

Chapter 3

RELATED WORK

The problem of domain adaptation is defined by the domain gap that exists between related but distinct domains. In this thesis, a view is adopted that the amelioration of the discrepancy between source and target domains can be equivalently seen as reducing a domain adaptation problem to a semi-supervised learning problem. As such, it is prudent to provide a survey of recent literature from both problem spaces in order to effectively outline the landscape in which the contribution of this work resides. Domain adaptation is ubiquitous in the fields of machine learning; however, we'll focus most of our attention on methods developed for computer vision.

3.1 Feature Reduction

Classification tasks are often accomplished by first embedding high-dimensional inputs into lower dimensional embeddings or *features*, and then training a classifier on those features. Classic techniques for image classification in computer vision relied on designing functions by hand to extract meaningful features, which could then be used as input for shallow learning models such as Support Vector Machines (SVMs) to generate class predictions (Dalal and Triggs (2005)). The advent of deep neural networks spawned what is now the dominant modeling technique for image feature extraction – Convolutional Neural Networks (CNNs) (LeCun *et al.* (1989)). CNNs enable the feature extractor and classifier to be trained end-to-end using the same objective function, and have been shown to automatically learn transformations that generate transferable features.

3.2 Deep Domain Adaptation

Recent algorithms and model architectures designed to resolve the domain gap mostly follow one of two primary methodologies: adversarial training and moment matching.

3.2.1 Adversarial Methods

Generative Adversarial Networks (GANs) Goodfellow *et al.* (2014) first introduced the idea of using a discriminative module to adversarially train a generative network in order to improve its ability to produce realistic, fake data samples. This was accomplished by training a discriminator D to accurately predict the label for real and fake samples, while simultaneously training a generator G to minimize $log(1 - D(G))$ and thus to generate fake samples that would fool D into thinking they're real. Extensive work expounding on the utility and performance of GANs quickly followed (Ganin and Lempitsky (2014); Zhu *et al.* (2017); Kim *et al.* (2017); Hoffman *et al.* (2017); Sankaranarayanan *et al.* (2018)).

Inspired by the discriminative mechanism used for measuring the distribution discrepancy in GANs, Ganin and Lempitsky (2014) introduced a similar mechanism with the goal of minimizing the discrepancy for domain adaptation by training a feature extractor to instead confuse a domain discriminator – making it uncertain about whether samples originated from the source or the target domain. This approach, called the Domain Adversarial Neural Network (DANN), follows directly from the theory that effective domain transfer necessitates that the predictions be made on features that are invariant of the domain from which they originated. Where GANs required a two-step iterative process to first train the discriminator and then the generator, Ganin and Lempitsky (2014) introduced the gradient reversal layer (GRL), enabling the network to be trained end-to-end by simply reversing the gradient of the discriminator during backpropagation to train the feature extractor. The simplicity and effectiveness of the GRL for domain adaptation has resulted in extensive utilization of this technique in recent domain adaptation literature, including: class-level predictions using multiple domain discriminators conditioned on the softmax predictions of the classifier Chen *et al.* (2017); combinations of global feature domain discriminators augmented with domain-specific loss functions for learning semantic details of the domains Tsai *et al.* (2018); Chen *et al.* (2018); leveraging multiple local domain discriminators and a global discriminator as attention mechanisms for fine-grained transfer Wang *et al.* (2019b); and using the output of domain discriminators as sample-level weighting mechanisms in various domain adaptation settings Zhang *et al.* (2018); Cao *et al.* (2019); You *et al.* (2019).

CycleGANs, introduced in Zhu *et al.* (2017), enable unpaired image-to-image translation by combining the domain alignment method from the standard GAN architecture with a *cycle-consistency* loss at the pixel-level. CycleGANs train two GANs in parallel – one to translate the input from source to target, and the other to translate from target to source. By passing an input through both generative networks, a cycle-consistency loss is calculated by measuring the error between the original input and the generated one.

Following the success of the CycleGAN architecture, Hoffman *et al.* (2017) developed CyCADA in order to constrain the input mapping to retain vital semantic information within the image that may otherwise be lost. In doing so, they introduced the power of cycle-consistency to the world of unsupervised domain adaptation.

3.2.2 Statistical Moment Matching

Another method of aligning the domain distributions is by explicitly matching statistical measures of deep feature representations between the domains. The Maximum Mean Discrepancy (MMD) is a standard metric used to estimate the distance between two distributions. The Domain Adaptation Network (DAN) calculates a multi-kernel variant of this metric, MK-MMD, for the final layers of the network, and minimizes these discrepancies alongside the standard classification loss in order to directly align the domains. Since its application in DAN, MK-MMD and a series of other distribution divergence measures have have been widely adopted and adapted in domain adaptation research. Shen *et al.* (2017) uses a neural network to estimate the Wasserstein distance as an objective measure between the source and target domains. Venkateswara *et al.* (2017a) minimizes the MK-MMD while training the network to learn hash values for each object category in an unsupervised domain adaptation setting. Long *et al.* (2017) uses the joint probability distribution across the final layers of the Joint Adaptation Network (JAN) in order to train using a joint maximum mean discrepancy (JMMD) criterion.

3.3 Semi-Supervised Learning

The semi-supervised learning (SSL) paradigm considers scenarios where the labeled data available is insufficient to train a strong model, and seeks to improve the performance of the model by including unlabeled data in the training process. In order to utilize unlabeled data, SSL algorithms hinge on the following assumptions:

- **Smoothness assumption Chapelle *et al.* (2010):** If two data points are close to each other, their respective labels should also be close to each other.

- **Low-density separation assumption Chapelle and Zien (2005):** The decision boundaries between classes should occur in areas of low density in the feature space. This can be equivalently formulated as the cluster assumption, which states that two points that fall within the same cluster are likely to belong to the same class.

- **Manifold assumption Belkin and Niyogi (2004); Chapelle *et al.* (2010):** High-dimensional data lies on a low dimensional manifold.

For deep learning models, these assumptions are exploited by encoding them into functions and then appending those to the objective function being minimized to train the neural network. In this section, we discuss the three most dominant SSL objectives being deployed in modern models.

3.3.1 Entropy Minimization

Many shallow and deep learning methods make use of the cluster assumption by minimizing the intracluster distance while also maximizing the intercluster distance, resulting in decision boundaries that pass through low-density regions of the space. The entropy minimization principle Grandvalet and Bengio (2005) is used to coax deep neural networks into producing such clusters by noting that the classification of unlabeled samples should be confident, and confident predictions lead to lower entropy on the prediction vector. Minimizing the entropy of the unlabeled data predictions therefore encourages low-density separation of the of the feature embeddings.

Given the entropy function

$$\mathcal{H}(\mathbf{X};\theta) = -\mathbb{E}_{\mathbf{X}} log \left[P(\mathbf{Y}|\mathbf{X};\theta) \right],$$

entropy minimization can be implemented explicitly into an SSL loss function as

$$\mathcal{L}_{EM} = \mathcal{L}_l - \lambda \mathcal{H}(\mathbf{X}_u;\theta),$$

where $\mathcal{L}_l$ is the standard cross-entropy loss for supervised training on the labeled data (discussed further in Section 4.0.3), $\mathbf{X}_u$ is the unlabeled target dataset, and λ is the hyperparameter used to control the influence of the unlabeled data on the overall training process Grandvalet and Bengio (2005). Note that by maximizing $\mathcal{L}_{EM}$, the objective function simultaneously maximizes the cross-entropy loss while minimizing the empirical entropy. The value given to λ is important – if λ is set too high, the supervised learning signal will be overpowered; if set too low, the model will not be able to learn from the unlabeled data. This hyperparameter is often implemented using a deterministic annealing scheme to slowly increase the influence of the entropy signal over time.

Pseudo-labelling (Lee (2013)) can be seen as an equivalent, implicit implementation of entropy minimization that enables supervised training on unlabeled data. Pseudo-labels are generated by simply treating the highest probability class prediction as the true class, that is,

$$\boldsymbol{y}_i = \begin{cases} 1 & \text{if } i = argmax \left[p(\boldsymbol{y}_i | \boldsymbol{x}) \right] \\ 0 & \text{otherwise} \end{cases},$$

and then training using cross-entropy loss on both the labeled and pseudo-labeled data.

Similarly, a sharpening function can be applied to the prediction vectors of unlabeled data to generate *soft* pseudo-labels with lower entropy (Berthelot *et al.* (2019)). Supervised training with these sharpened predictions also results in an implicit minimization of entropy.

3.3.2 Consistency Regularization

Consistency-based learning methods exploit the smoothness assumption of SSL by ensuring that an input sample is consistently mapped to the same point in the feature space or label space. In the π-model (Laine and Aila (2016)), each input is passed through the model twice with different dropout initializations, and the mean squared difference of the predictions is penalized. In the same work, Laine and Aila (2016) note that the π-model could just as effectively be implemented by instead generating predictions on the inputs without backpropogation to generate pseudo-labels, and then passing augmentations of the input using a different dropout initialization to train the network using these predictions with the pseudo-labels as the targets for the unsupervised loss component. As such, the update of the π-model is based on a single initialization of the network, is inherently noisy. To resolve this issue, *temporal ensembling* is proposed. In this method, the exponential moving average (EMA) of the network prediction p_i is maintained for each sample $\boldsymbol{x}_i$.

$$\bar{\boldsymbol{p}}_i = \alpha \bar{\boldsymbol{p}}_i + (1 - \alpha)\boldsymbol{p}_i,$$

where α determines how much weight is applied to previous predictions.

Where temporal ensembling asserts consistency on the *predictions* of a network, the mean-teacher model (Tarvainen and Valpola (2017)) asserts consistency on the *weights* of a network by materializing a "teacher" network whose weights are the exponential moving average (EMA) of the "student's". In Li *et al.* (2019), certainty-driven consistency loss (CCL) is proposed for mean-teacher models to either filter or weight the impact of consistency training at the instance-level by measuring the predictive variance of each sample with different augmentations. For the Domain Adaptive Fusion network, multiple augmentations of each unlabeled sample are evaluated and then assigned the averaged, sharpened prediction for that sample as a label to encourage consistent predictions.

Other consistency regularization methods have been developed that generate new training data to span the space between the inputs of the original dataset. In Mixup (Zhang *et al.* (2017)), a data augmentation-based regularization technique was proposed to compel models to learn a linear continuity between convex combinations of the input features and their corresponding classification labels in a supervised learning setting. This is accomplished by sampling a mixing coefficient λ_i from the Beta distribution

$$\lambda_i \sim \beta(\alpha, \alpha), \forall (\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathbf{X}$$

and a random data point $(\boldsymbol{x}_j, \boldsymbol{y}_j) \in \mathbf{X}$ to generate an augmented data point $(\boldsymbol{x}_{i,a}, \boldsymbol{y}_{i,a})$ where

$$\boldsymbol{x}_{i,a} = \lambda \boldsymbol{x}_i + (1 - \lambda)\boldsymbol{x}_j$$
$$\boldsymbol{y}_{i,a} = \lambda \boldsymbol{y}_i + (1 - \lambda)\boldsymbol{y}_j.$$

In CutMix (Yun *et al.* (2019)), the spirit of the Mixup augmentation strategy is applied to the regional dropout technique of Cutout. Where Cutout creates augmented inputs by removing a region of the input image, CutMix then fills the removed region with a patch from another image. As the augmented image may then contain a mixture of two different classes, they also generate a new label for the augmented data with a proportional mixture of the labels corresponding to the original inputs.

The usefulness of Mixup regularization has since been studied in the realm of semi-supervised learning problems, including in the recently developed MixMatch algorithm from Berthelot *et al.* (2019), which uses a combination of Mixup, label sharpening, and entropy minimization principles to produce a holistic objective function for utilizing unlabeled data. MixMatch uses the sharpened predictions of the target samples as pseudolabels, and then applies the mixup algorithm to both the source and target samples, resulting in two augmented datasets that each contain mixtures from both domains. The outstanding results of MixMatch inspired our interest in exploring the combined efficacy of these techniques when applied to various settings of domain adaptation, where discrepancies between the marginal distributions of the labeled and unlabeled datasets introduce challenges not faced by the preceding studies.

Concurrent to the development of MixMatch, Verma *et al.* (2019) applied Mixup to another SSL technique called Interpolation Consistency Training (ICT). ICT differs from MixMatch in a few small but meaningful ways: It doesn't use sharpening for entropy minimization (and instead uses standard pseudo-labels for the target data), it only applies mixup to the target data, and it adopts the mean teacher approach of maintaining a second network for classification whose weights are the exponential moving average of the weights of the primary network being trained.

3.3.3 Standard Regularization

Machine learning algorithms generally seek to learn a generalized function from the dataset they're trained on. Regularization penalties are often imposed in order to avoid overfitting to the data and thus to improve the generalizability of the learned model. The expansive number of weights in large neural networks make them particularly prone to memorization of the data when effective regularization is not employed. A simple method of moderating the complexity of a model is by penalizing the magnitude of the weights using *L2 regularization* (Krogh and Hertz (1992); Ng (2004)).

Chapter 4

DEEP DOMAIN FUSION

This chapter introduces Domain Adaptive Fusion (DAF), a deep neural network which performs domain alignment and domain fusion towards unsupervised domain adaptation. The following sections outline the different components of the DAF network, and motivate their utility in the training process.

In unsupervised domain adaptation we have labeled data from the source domain; $\mathcal{D}_s = \{\boldsymbol{x}_i^s, y_i^s\}_{i=1}^{n_s}$, and unlabeled data from the target domain; $\mathcal{D}_t = \{\boldsymbol{x}_i^t\}_{i=1}^{n_t}$. The data points $\boldsymbol{x}_i^*$ belong to an input space denoted by X and the labels belong to a discrete space $y_i^* \in Y := \{1, \ldots, \mathcal{C}\}$. The goal is to determine the unknown target data labels given the constraint that the source and target data joint distributions are different, i.e., $\mathcal{P}_s(X,Y) \neq \mathcal{P}_t(X,Y)$. The DAF network has parameters $\theta := \{\theta_G, \theta_D, \theta_C\}$, where θ_G are the parameters for the base feature extractor component G, θ_D are the parameters for the domain alignment component D and θ_C are the parameters for the classifier C. The different components of the DAF and the gradient paths are illustrated in Figure 4.1. When training the DAF, we deploy mini-batches of size $2B$ with B samples $\mathcal{X}_s = \{\boldsymbol{x}_i^s\}_{i=1}^B$ and $\mathcal{Y}_s = \{y_i^s\}_{i=1}^B$ from source and B samples $\mathcal{X}_t = \{\boldsymbol{x}_i^t\}_{i=1}^B$ from the target. We describe the model in terms of mini-batches and note that it can be extended to the entire dataset.

4.0.1 Domain Alignment

In order to reduce the domain adaptation problem to a semi-supervised one, we align the features of the source and target. For G to output domain-aligned features we adopt the domain confusion model from DaNN Ganin *et al.* (2016), to train an

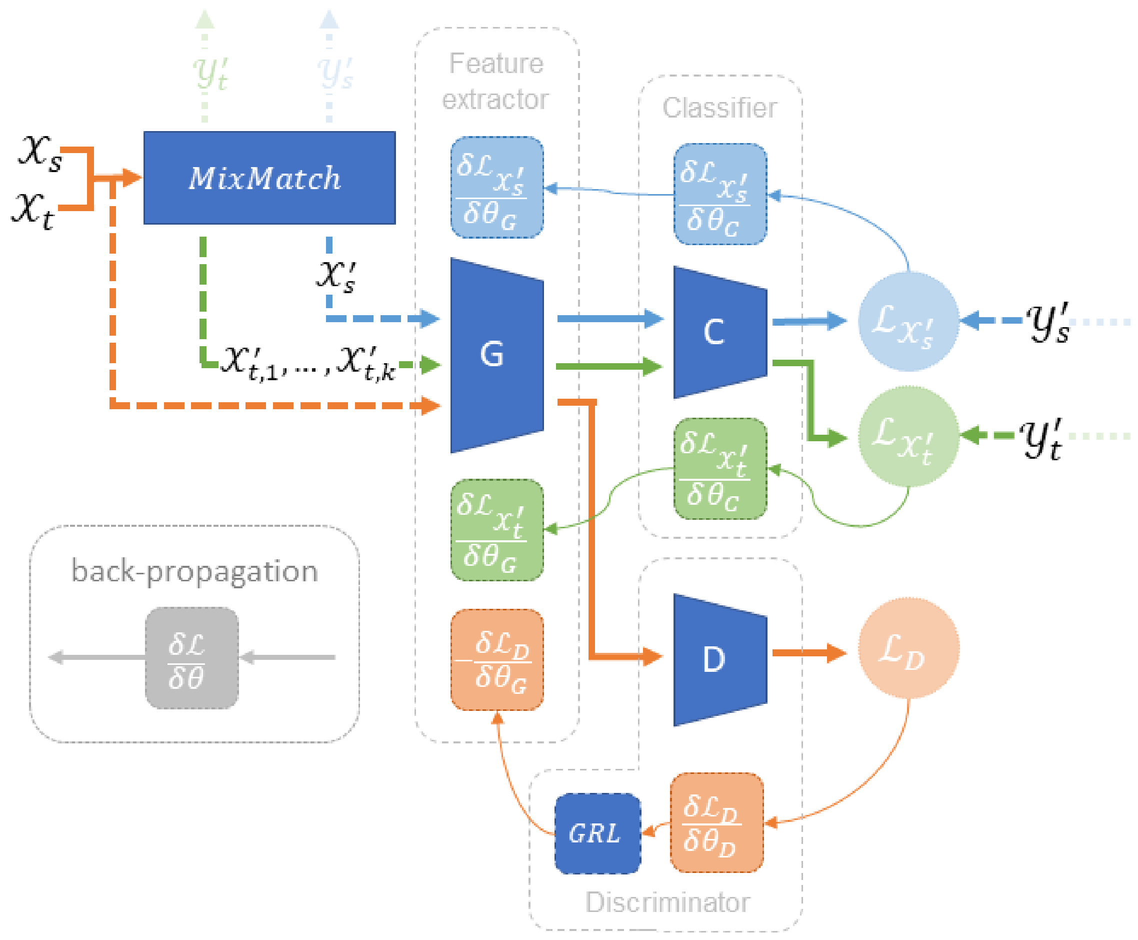

Feature extractor
Classifier
MixMatch
$\mathcal{X}_s$
$\mathcal{X}_t$
$\mathcal{Y}'_t$
$\mathcal{Y}'_s$
$\mathcal{X}'_s$
$\mathcal{X}'_{t,1}, \dots, \mathcal{X}'_{t,k}$
G
C
D
$\frac{\delta \mathcal{L}_{\mathcal{X}'_s}}{\delta \theta_G}$
$\frac{\delta \mathcal{L}_{\mathcal{X}'_s}}{\delta \theta_C}$
$\frac{\delta \mathcal{L}_{\mathcal{X}'_t}}{\delta \theta_C}$
$\frac{\delta \mathcal{L}_{\mathcal{X}'_t}}{\delta \theta_G}$
$-\frac{\delta \mathcal{L}_D}{\delta \theta_G}$
$\frac{\delta \mathcal{L}_D}{\delta \theta_D}$
$\mathcal{L}_{\mathcal{X}'_s}$
$\mathcal{L}_{\mathcal{X}'_t}$
$\mathcal{L}_D$
back-propagation
$\frac{\delta \mathcal{L}}{\delta \theta}$
GRL
Discriminator

Figure 4.1: The Domain Adaptive Fusion architecture. $\mathcal{X}_s$ and $\mathcal{X}_t$ represent the mini-batches from the source and target domains. $\mathcal{X}'_s$ and $\{\mathcal{X}'_{t,1}, \dots, \mathcal{X}'_{t,k}\}$ represent the augmented batches generated by the *MixMatch* algorithm. The neural network modules for the feature extractor, classifier, and domain discriminator are represented by G, C, and D, and GRL represents the gradient reversal layer used for domain adversarial training during backpropogation. The supervised classification task for the augmented source data is shown in light blue, where $\mathcal{L}_{\mathcal{X}'_f}$ corresponds to the cross-entropy loss objective. The semi-supervised task for the k-augmented target data is show in green, where $\mathcal{L}_{\mathcal{X}'_t}$ represents the consistency regularization objective using the multi-class brier score. The task of reducing the feature distribution discrepancy is shown in orange, where $\mathcal{L}_D$ represents the domain adversarial loss objective.

auxiliary network D to align the features output from G. If $d \in \{1, 0\}$ are the domain labels where $d = 1$ for source samples and $d = 0$ for target samples, the discriminator network D tries to minimize,

$$\mathcal{L}_D = -\frac{1}{2B} \sum_{x \in \{\mathcal{X}_s \cup \mathcal{X}_t\}} d\log[D(G(\boldsymbol{x}))] + (1-d)(1 - \log[D(G(\boldsymbol{x}))]), \tag{4.1}$$

where $D(G(\boldsymbol{x}))$, is the output probability from a sigmoid activation. The discriminator is trained through back propagation to minimize $\mathcal{L}_D$, i.e., distinguish between the source and target samples. On the other hand, a gradient reversal (*GRL* in Figure 4.1) is applied to modify the parameters of G in an adversarial manner in order to align the source and target features and make them indistinguishable to the discriminator. This involves reversing the gradient $-\frac{\partial \mathcal{L}_D}{\partial \theta_G}$ during back propagation over the parameters in G. The domain alignment component ensures that the source and target features output from $G(.)$ have no little to no domain discrepancy.

4.0.2 Domain Fusion

With domain alignment in place, the G network plays the role of a feature extractor that aligns the source and target data features. This reduces the domain adaptation problem to a semi-supervised learning problem with the source data being treated as the labeled set and the target data becoming the unlabeled set. In the following we outline the steps to implement domain fusion.

Data Augmentation

As is common with semi-supervised learning procedures, we estimate artificial labels for the target data using consistency regularization techniques Miyato *et al.* (2015); Tarvainen and Valpola (2017); Zhou *et al.* (2004). We augment the training data with multiple stochastic transformations of the input $\boldsymbol{x}$ to yield different versions of the input that have the same label. Data augmentation is performed on the input vectors for both the source and target batches using an $\texttt{Augment}(\boldsymbol{x})$ function, which performs random flips and crops on the input image $\boldsymbol{x}$. The source inputs are augmented once, and the target inputs are augmented K times to produce K different augmentations of the target batch:

$$\hat{\mathcal{X}}_s = \{\texttt{Augment}(\boldsymbol{x}_i^s)\}_{i=1}^{B} \qquad \hat{\mathcal{X}}_t = \{\texttt{Augment}(\boldsymbol{x}_i^t)_k\}_{i=1,k=1}^{B,K} \tag{4.2}$$

Soft Pseudo-labeling

We perform consistency regularization on the unlabeled data by ensuring that the same pseudo-label is assigned to each of the K augmented versions of an input image $\boldsymbol{x}$. These pseudo-labels are generated by first predicting a soft label $y_{i,k}^t$ for each $\boldsymbol{x}_{i,k}^t \in \hat{\mathcal{X}}_t$, with,

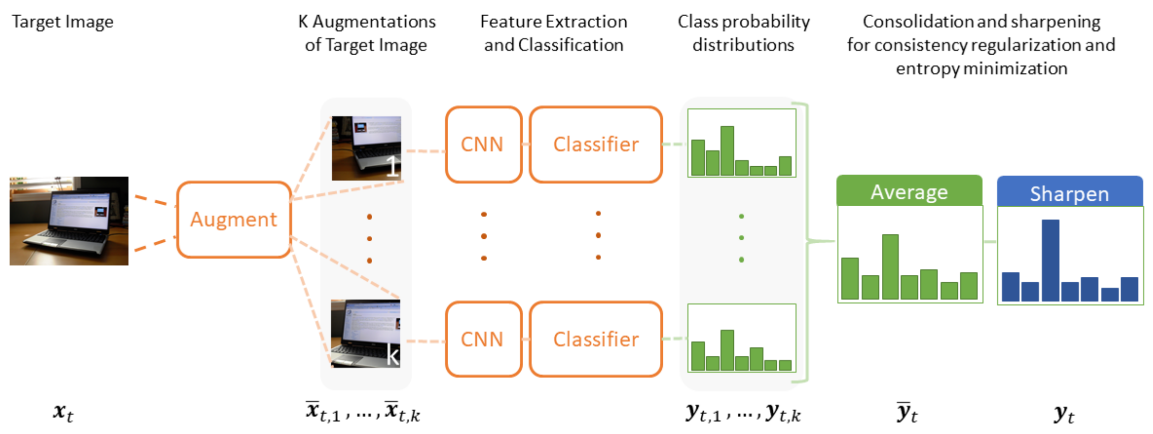
$$\boldsymbol{y}_{i,k}^t = C(G(\boldsymbol{x}_{i,k}^t)) \quad \forall i \in \{1, \ldots, B\}, k \in \{1, \ldots, K\}, \tag{4.3}$$

where $C(.)$ is the classifier network and $C(G(\boldsymbol{x}))$ gives the softmax output from the classifier network - a probability vector $\boldsymbol{y}_{i,k}^t = [p_{i,k}^{1,t}, \ldots, p_{i,k}^{\mathcal{C},t}]^\top$ over $\mathcal{C}$ classes, where $p_{i,k}^{c,t}$ is the probability $p(y_{i,k}^t = c|\boldsymbol{x}_{i,k}^t)$. To arrive at a consistent prediction for the unlabeled data, we average over the K predictions and estimate a common label for each of the augmented input images $\{\boldsymbol{x}_{i,1}^t, \ldots, \boldsymbol{x}_{i,K}^t\}$:

$$\boldsymbol{y}_i^t = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{y}_{i,k}^t \quad \forall i \in \{1, \ldots, B\}. \tag{4.4}$$

Following the approach proposed in Berthelot *et al.* (2019), we encourage the low-density separation of class assignments to target data samples by implicitly exploiting the minimum entropy criterion using a sharpening function on $\boldsymbol{y}_i^t$. Specifically,

$$\hat{\boldsymbol{y}}_{i,k}^t = \frac{(\boldsymbol{y}_i^t)^{1/T}}{\sum_{c=1}^{\mathcal{C}} (y_i^{c,t})^{1/T}} \quad \forall i \in \{1, \ldots, B\}, k \in \{1, \ldots, K\}, \tag{4.5}$$



Target Image
K Augmentations of Target Image
Feature Extraction and Classification
Class probability distributions
Consolidation and sharpening for consistency regularization and entropy minimization
Augment
CNN
Classifier
CNN
Classifier
1
K
Average
Sharpen
x_t
$\bar{x}_{t,1}, \dots, \bar{x}_{t,k}$
$y_{t,1}, \dots, y_{t,k}$
$\bar{y}_t$
y_t

Figure 4.2: Illustration of the data augmentation process applied to images in the target domain.

where the hyperparameter T controls the *temperature* of the distribution Goodfellow *et al.* (2016). As $T \rightarrow 0$, $\hat{\boldsymbol{y}}_{i,k}^{t}$ approaches the Dirac-delta function, which will produce one-hot labels. The pseudo labels $\hat{\boldsymbol{y}}_{i,k}^{t}$ are then assigned to their corresponding input vectors in $\hat{\mathcal{X}}_t$, with all the K augmentations of $\boldsymbol{x}_{i,k}^{t}$ for $k \in \{1, \dots, K\}$ assigned the same pseudo label $\hat{\boldsymbol{y}}_{i,k}^{t}$ for $k \in \{1, \dots, K\}$. The data augmentation for the source results in the modified datasets, $\hat{\mathcal{X}}_s$ (Equation 4.2) with one-hot labels $\hat{\mathcal{Y}}_s$ where $\hat{\mathcal{Y}}_s$ is one-hot vector representation of source data labels $\mathcal{Y}_s = \{y_i^s\}_{i=1}^{B}$. Likewise, the data augmentation followed by pseudo labeling for the target dataset yields $\hat{\mathcal{X}}_t$ (Equation 4.2) and the corresponding labels $\hat{\mathcal{Y}}_s$ where $\hat{\mathcal{Y}}_s = \{\hat{\boldsymbol{y}}_{i,k}^{t}\} \forall i \in \{1, \dots, B\}, k \in \{1, \dots, K\}$. The data augmentation and pseudo label generation is depicted in Figure 4.2.

Data Fusion

The DAF network is trained on data generated from the fusion of source and target samples. Our hypothesis is that a model with linear behavior across domains will be an effective classifier for data from both the domains. Once the domains are aligned,

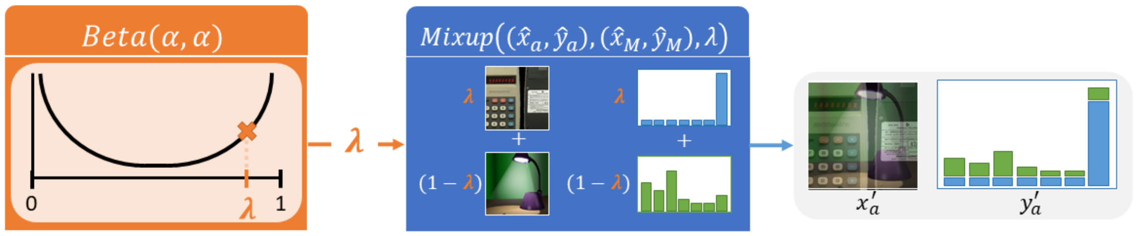


Figure 4.3: Illustration of the mixup function, where $(\hat{\mathbf{x}}_a, \hat{\mathbf{y}}_a) \in \hat{\mathcal{X}}_s || \hat{\mathcal{X}}_t$, and $(\hat{\mathbf{x}}_M, \hat{\mathbf{y}_M}) \in \hat{\mathcal{X}}_M$. $(\mathbf{x}'_a, \mathbf{y}'_a)$ represents an augmented, mixed-up, labeled datum that will be used during the training phase of Domain Adaptive Fusion.

we train the DAF with a convex combination of data from both the domains along with a convex combination of their corresponding labels. We employ the *MixUp* procedure Zhang *et al.* (2017) to enforce a linear behavior between data from the two domains. We constrain the DAF model output for a convex combination of inputs to be similar to the convex combination of the DAF model outputs over the individual inputs. We create a unified set of augmented data samples, $\hat{\mathcal{X}}_m = \{\hat{\mathcal{X}}_s \cup \hat{\mathcal{X}}_t\}$ by concatenating and shuffling the augmented source and target datasets. Likewise, we create the unified label set $\hat{\mathcal{Y}}_m = \{\hat{\mathcal{Y}}_s \cup \hat{\mathcal{Y}}_t\}$ all the while maintaining the order between data in $\hat{\mathcal{X}}_m$ and their labels in $\hat{\mathcal{Y}}_m$. The fusion dataset is created using the *MixUp* procedure,

$$\mathcal{X}'_s = \texttt{MixUp}(\hat{\mathcal{X}}_s, \{\hat{\mathcal{X}}_{m,i}\}_{i=1}^{B}, \alpha) \qquad \mathcal{Y}'_s = \texttt{MixUp}(\hat{\mathcal{Y}}_s, \{\hat{\mathcal{Y}}_{m,i}\}_{i=1}^{B}, \alpha) \tag{4.6}$$

$$\mathcal{X}'_t = \texttt{MixUp}(\hat{\mathcal{X}}_t, \{\hat{\mathcal{X}}_{m,i}\}_{i=B+1}^{|\hat{\mathcal{X}}_m|}, \alpha) \qquad \mathcal{Y}'_t = \texttt{MixUp}(\hat{\mathcal{Y}}_t, \{\hat{\mathcal{Y}}_{m,i}\}_{i=B+1}^{|\hat{\mathcal{Y}}_m|}, \alpha). \tag{4.7}$$

The $\texttt{Mixup}(X_1, X_2, \alpha)$, takes two equal sized sets as input along with hyperparameter α. It then performs a linear combination of elements from X_1 and X_2 to create a fused dataset of the same size as X_1. Mixup samples a mixing value λ from the U-shaped $Beta(\alpha, \alpha)$ distribution, where $0 < \alpha < 1$. As alpha approaches 0, $Beta(\alpha, \alpha)$ approaches the Bernoulli distribution. We illustrate `MixUp()` with an example. Let $\boldsymbol{x}_1^i \in X_1$ and $\boldsymbol{x}_2^i \in X_2$ be the i^{th} elements of X_1 and X_2. Let Y_1 and Y_2 be the labels corresponding to X_1 and X_2. If $\boldsymbol{y}_1^i \in Y_1$ and $\boldsymbol{y}_2^i \in Y_2$ are the i^{th} elements of Y_1 and Y_2. Then, the fusion of $\texttt{MixUp}(\{\boldsymbol{x}_1^i\}, \{\boldsymbol{x}_2^i\}, alpha)$ and $\texttt{MixUp}(\{\boldsymbol{y}_1^i\}, \{\boldsymbol{y}_2^i\}, alpha)$, would yield,

$$\boldsymbol{x}_i' = \lambda \boldsymbol{x}_1^i + (1-\lambda)\boldsymbol{x}_2^i \tag{4.8}$$

$$\boldsymbol{y}_i' = \lambda \boldsymbol{y}_1^i + (1-\lambda)\boldsymbol{y}_2^i \tag{4.9}$$

In practice we set $\lambda = \lambda_{max} = \max(\lambda, (1-\lambda))$. This is in order to ensure that the majority of the mixing weight is given to the original sample of the batch being mixed (samples from the first argument of the mixup function). The data fusion procedure is illustrated in Figure 4.3. The data fusion component creates fused samples from domain-aligned source and target samples and trains the DAF model to predict their fused labels. We consider the DAF model to be robust to domain shift because it is trained with fused samples from the source and the target. In practice, the data augmentation procedure is treated as auxiliary to the training process. This is accomplished by detaching the augmented batches $\mathcal{X}_s'$ and $\mathcal{X}_t'$ from the network to prevent the flow of the gradient through the augmentation steps during backpropagation.

4.0.3 Objective Function

The DAF network is guided by the following objectives functions. The labeled data ($\mathcal{X}_s'$ and $\mathcal{Y}_s'$) has more confident labels since it is created using the ground truth source labels. The labeled data is used to minimize the cross-entropy objective,

$$\mathcal{L}_s = \frac{1}{|\mathcal{X}_s'|} \sum_{\boldsymbol{x}_i'^s \in \mathcal{X}_s', \boldsymbol{y}_i'^s \in \mathcal{Y}_s'} \mathrm{KL}(\boldsymbol{y}_i'^s || C(G(\boldsymbol{x}_i'^s))), \tag{4.10}$$

where KL stands for Kullback-Leibler divergence which estimates the cross-entropy between labels $\boldsymbol{y}_i'^s$ and DAF prediction $C(G(\boldsymbol{x}_i'^s))$. The pseudo-labeled data (${\mathcal{X}'}_t$ and ${\mathcal{Y}'}_t$) has artificial labels. In view of the less confident labels for the target, we apply the Brier score Berthelot *et al.* (2019), which is less sensitive to outliers and is bounded. This is a standard loss function for unlabeled data in semi-supervised learning literature Laine and Aila (2016). The objective function for the unlabeled data is given by,

$$\mathcal{L}_t = \frac{1}{|\mathcal{X}'_t|} \sum_{\boldsymbol{x}'^t_{i,k} \in \mathcal{X}'_t, \boldsymbol{y}'^t_{i,k} \in \mathcal{Y}'_t} \left|\left| \boldsymbol{y}'^s_{i,k} - C(G(\boldsymbol{x}'^s_{i,k})) \right|\right|_2^2, \tag{4.11}$$

Finally, to discourage DAF from overfitting to the training data, the $L2$ regularization loss is applied across the layer of the network's parameters,

$$\mathcal{L}_2 = \sum_{\theta_i \in \theta} \|\theta_i\|_2^2. \tag{4.12}$$

DAF Objective Function

The objective for the DAF model is estimated from Equations (4.1), (4.10), (4.11) and (4.12). In each iteration the DAF objective is determined by a two-player, minimax game,

$$(\theta_G, \theta_C) = \underset{\theta_G, \theta_C}{\arg\min} \left[\mathcal{L}_s - \lambda \mathcal{L}_D + \gamma \mathcal{L}_t + \eta \mathcal{L}_2\right] \tag{4.13}$$

$$(\theta_D) = \underset{\theta_D}{\arg\min} \left[\lambda \mathcal{L}_D + \eta \mathcal{L}_2\right] \tag{4.14}$$

where, λ, γ and η are hyperparameters that control the importance of the corresponding terms in the DAF objective.

Chapter 5

EXPERIMENTAL SETUP

The Domain Adaptive Fusion (DAF) network was evaluated on a collection of transfer tasks contained within several domain adaptation datasets. We start this chapter by first providing a description of each of these datasets and identifying some of the challenges posed by their transfer tasks. We then provide implementation details regarding our network and training parameters used during experimentation.

5.1 Datasets

Office-31 The *Office-31* dataset Saenko *et al.* (2010) is a de-facto standard in computer vision for benchmarking domain adaptation techniques. It consists of three domains – Amazon (**A**), Webcam (**W**), and DSLR (**D**) – with 31 categories of images in each domain, and 4,652 images in total. Images in the Amazon dataset were collected from `amazon.com`, while the images from Webcam and DSLR are taken with a webcam and digital SLR camera, respectively.

Historically, the $\mathbf{W} \to \mathbf{A}$ and $\mathbf{D} \to \mathbf{A}$ transfer tasks have proven to be particularly trying for the adaptive networks that have attempted to bridge the gap between these domains. It can immediately be noted that where the Webcam and DSLR datasets have 500 and 800 images, respectively, the Amazon dataset has over 2800 images. While this fact alone is likely to contribute the challenge of learning to adapt and generalize from Webcam and DSLR, further analysis in Section 6.2.1 reveals additional challenges of these tasks.

Office-Home The *Office-Home* dataset Venkateswara *et al.* (2017b) consists of approximately 15,500 images of common household and office objects. With 65 cate-

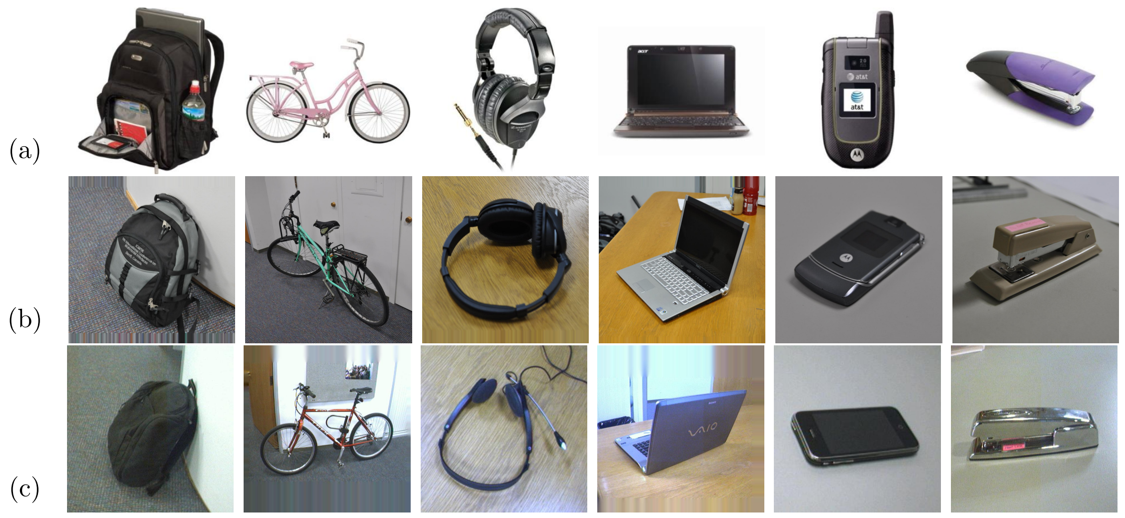


Figure 5.1: Images sampled from the Office-31 dataset. (a) Amazon (b) DSLR (c) Webcam

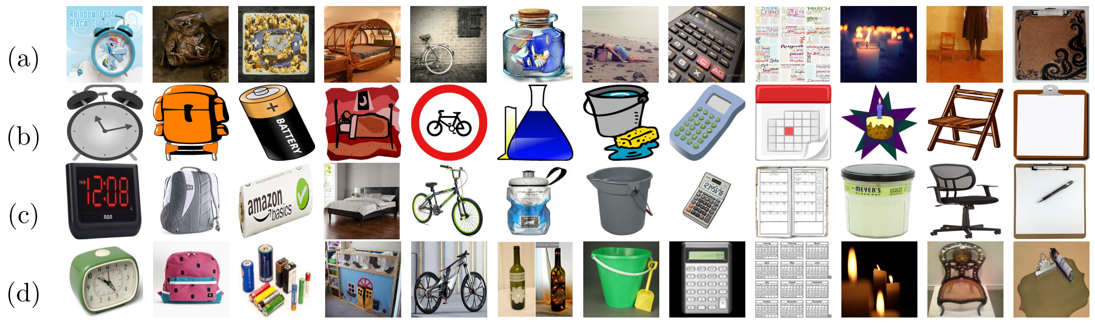


Figure 5.2: Images sampled from the Office-Home dataset

gories and four unique domains, this dataset constitutes a more challenging set of domain adaptation tasks as compared to *Office-31*. The domains include Art (**Ar**), Clipart (**Cl**), Product (**Pr**), and Real World (**Rw**).

5.2 Implementation

The network architecture and training procedures of DAF were implemented in PyTorch. The feature extractor is comprised of a Resnet50 model He *et al.* (2016) with weights pre-trained on the ImageNet dataset Deng *et al.* (2009), which is fine-tuned during the training process. The adversarial domain discriminator and classification modules are each connected to the final convolutional layer of the ResNet50 model via a shared bottleneck layer to reduce the dimensions of their input features. The domain adversarial discriminator is implemented using a gradient reversal layer with a linear ramp-up coefficient calculated for the first 10,000 iterations of the training process. Our DAF model was built on top of the `CDAN codebase`[1] for loading image datasets and training parameters, and utilizes network class definitions provided by the `easydl`[2] deep learning utilities library.

An Adam optimizer Kingma and Ba (2014) is used for weight updates, where η is provided as the weight decay parameter for implementing the L2 regularization. All experiments were ran with batch size of 16 on a single Tesla V100 or Titan X using the following hyperparameter values: $\alpha = 0.75, \gamma = 10, \lambda = 1.5, \eta = 0.04, T = 0.5$. The learning rate was initialized at 0.001 for the classifier, domain discriminator, and bottleneck layer, and to 0.0001 for the ResNet50 model. It's important to note that the temperate hyperparameter T of the sharpening function directly affects the initial entropy value calculated on the target dataset. Reducing the value of this hyperparameter will reduce the entropy of the pseudo-labels, and may require adjusting the value of γ to prevent divergence while training.

During testing, ten different crops of each input image are generated and passed through the network. Samples are then classified as the most frequently predicted class of the ten augmentations. Some of the images of the office-home dataset have resolutions of over 2000x1000 pixels, so to speed up the training process all images in the were first rescaled to 256x256 pixels as a pre-processing step. In future work, generating a pre-processed collection of seeded augmentations saved directly as tensors would allow for more expedient training and tuning of the hyperparameters.

[1]https://github.com/thuml/CDAN

[2]https://github.com/thuml/easydl

Chapter 6

RESULTS AND ANALYSIS

The results of DAF on the domain adaptive computer vision classification tasks for the **Office-31** and **Office-home** datasets are reported in Table 6.1 and Table 6.2, respectively. The average reported performance values for each of the baseline models are compared against a single complete training run of DAF for each experimental setting.

6.1 Results

Method	**A→W**	**D→W**	**W→D**	**A→D**	**D→A**	**W→A**	Avg
ResNet He *et al.* (2016)	68.4	96.7	99.3	68.9	62.5	60.7	76.08
TCA Pan and Yang (2009)	72.7	96.7	99.6	74.1	61.7	60.9	77.62
GFK Gong *et al.* (2012)	72.8	95.0	98.2	74.5	63.4	61.0	77.48
DAN Long *et al.* (2015)	80.5	97.1	99.6	78.6	63.6	62.8	80.37
RTN Long *et al.* (2016)	84.5	96.8	99.4	77.5	66.2	64.8	81.53
DANN Ganin *et al.* (2016)	82.0	96.9	99.1	79.7	68.2	67.4	82.22
ADDA Tzeng *et al.* (2017)	86.2	96.2	98.4	77.8	69.5	68.9	82.83
JAN Long *et al.* (2017)	85.4	97.4	99.8	84.7	68.6	70.0	84.32
MADA Pei *et al.* (2018)	90.0	97.4	99.6	87.8	70.3	66.4	85.25
SimNet Pinheiro (2018)	88.6	98.2	99.7	85.3	**73.4**	**71.6**	86.13
GTA Sankaranarayanan *et al.* (2018)	89.5	97.9	99.8	87.7	72.8	71.4	**86.52**
CGAA Wang and Wang (2018)	75.2	95.7	99.6	72.3	57.2	57.5	76.25
DAF	**92.33**	**99.25**	**100.0**	**88.35**	68.12	70.22	86.38

Table 6.1: Classification accuracy on transfer tasks from *Office-31* dataset

On the *Office-31* dataset, DAF outperformed all baselines on the $\mathbf{A} \rightarrow \mathbf{W}, \mathbf{D} \rightarrow$

$\mathbf{W}, \mathbf{A} \to \mathbf{D}$, and $\mathbf{W} \to \mathbf{D}$ adaptive transfer tasks. Compared to DANN, DAF obtained a 10.3% performance improvement on $\mathbf{A} \to \mathbf{W}$ and an average 4.2% increase across all Office-31 experiments, indicating that the learning signal provided by the joint semi-supervised learning techniques on unlabeled target data significantly improves the model's effectiveness on the target domain. On the more challenging

Method	**Ar→Cl**	**Ar→Pr**	**Ar→Rw**	**Cl→Ar**	**Cl→Pr**	**Cl→Rw**	**Pr→Ar**	**Pr→Cl**	**Pr→Rw**	**Rw→Ar**	**Rw→Cl**	**Rw→Pr**	Avg
ResNet He *et al.* (2016)	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.13
DAN Long *et al.* (2015)	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.28
DANN Ganin *et al.* (2016)	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.5	76.8	57.60
JAN Long *et al.* (2017)	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.31
CGAA Wang and Wang (2018)	43.4	57.1	67.6	49.9	57.7	58.3	51.7	43.5	66.2	59.9	51.7	74.9	56.83
CDAN Long *et al.* (2018)	49.0	69.3	**74.5**	54.4	66.0	**68.4**	55.6	48.3	75.9	**68.4**	55.4	80.5	63.81
EasyTL Wang *et al.* (2019a)	**52.8**	**72.1**	75.9	55.0	65.9	67.6	54.5	46.9	74.7	63.8	52.3	78.0	63.30
DAF	48.8	66.1	73.5	**57.9**	**68.9**	67.9	**55.7**	**49.5**	**79.9**	68.3	**58.8**	**82.2**	**64.79**

Table 6.2: Classification accuracy on transfer tasks from *Office-Home* dataset

dataset of *Office-Home*, the performance of DAF exceeds all domain adaptation baseline methods on most of the transfer tasks.

6.2 Analysis

6.2.1 Class Relationships

To help visualize what classes are responsible for the prediction error in our network, confusion matrices were generated for the $\mathbf{A} \to \mathbf{W}$ task (see Figures 6.1, 6.2, 6.3). In a confusion matrix $\mathcal{C}$, each row i corresponds to the known correct class of the target data, and each column j corresponds to the predicted class. Each cell $\mathcal{C}_{ij}$ is shaded based on the number of samples from class i that were predicted as class j. The number of samples in each class aren't balanced, so we normalize the prediction counts across each row to get a clear view of the prediction distribution for each class.

The normalized confusion matrices for ResNet-50, DANN, and DAF are displayed in Figures 6.1a, 6.2a, and 6.3a, respectively. Simple visual inspection reveals that the

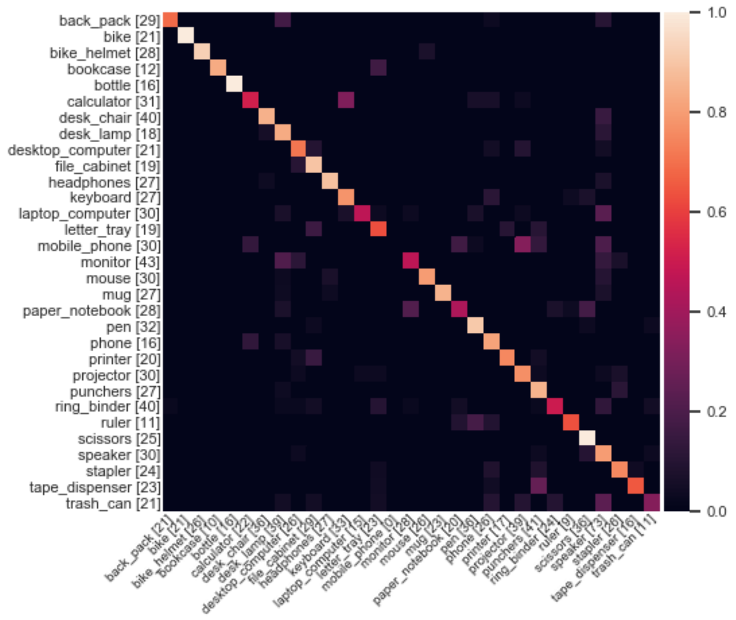
DAF model significantly outperforms the base ResNet-50 model, and it appears that DAF also met or exceeded the percentage of correctly classified samples for every class as compared to the DANN network. This is a strong indicator that the holistic combination of semi-supervised learning components employed in the DAF network was successful in utilizing the unlabeled target data towards the alignment of the source and target domain.

In Figures 6.1b, 6.2b, and 6.3b, the values on the diagonals are masked out (set to 0) prior to normalization in order to highlight the misclassified samples of each model. It is immediately apparent that the *classification entropy* is significantly reduced in our model – that is, the number of incorrect classes predicted for each class set of target data is reduced for almost every set.

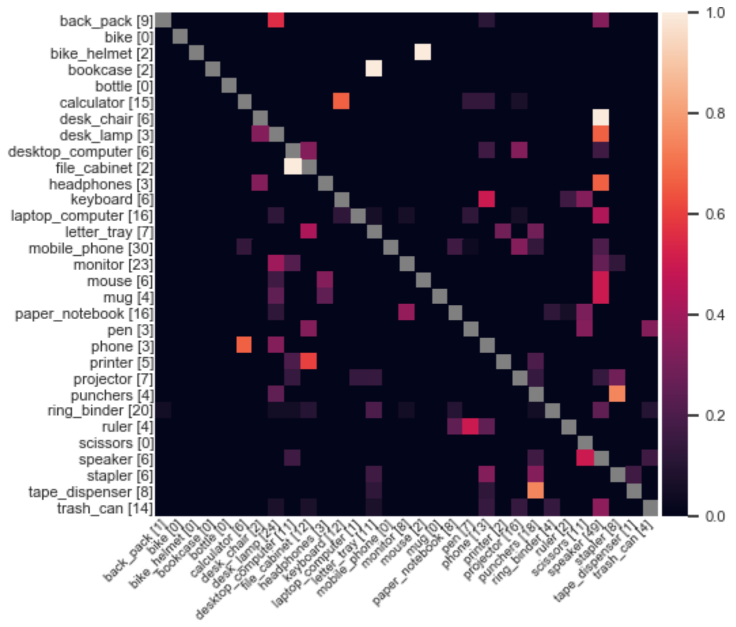
The masked confusion matrix for our DAF model has seven instances of only a single predicted class being responsible for all of the misclassifications in that row. To shed some light on the reason for these errors, we compare a few of the mislabeled samples with samples from their predicted classes (Figure 6.4).

To better understand the underwhelming performance of the DAF model on the $\mathbf{D} \rightarrow \mathbf{A}$ transfer task, another masked t-SNE plot is generated (see Figure **??**). Our network seems to predict the `mobile_phone` class with high precision but very low recall (there are 100 mobile phone samples in the target Amazon dataset). By the same token, the `speaker` class is also incorrectly predicted with very high frequency, and is contributing to significant percentage of the prediction error seen in our results. Both of these classes also happen to be the most frequently occurring classes in the source dataset, with up to four times more samples than other classes. This strongly indicates that the $\mathbf{D} \rightarrow \mathbf{A}$ transfer task suffers from a class imbalance problem, which DAF does nothing to compensate for. Instance-weighting or oversampling techniques may help ameliorate this issue, and is left for future work.

Confusion Matrices – ResNet-50 (source only)



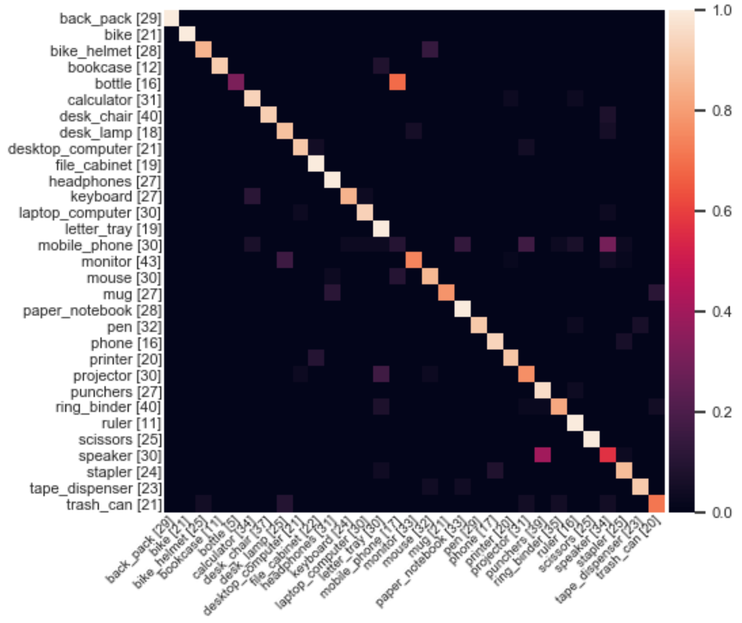
(a) Normalized



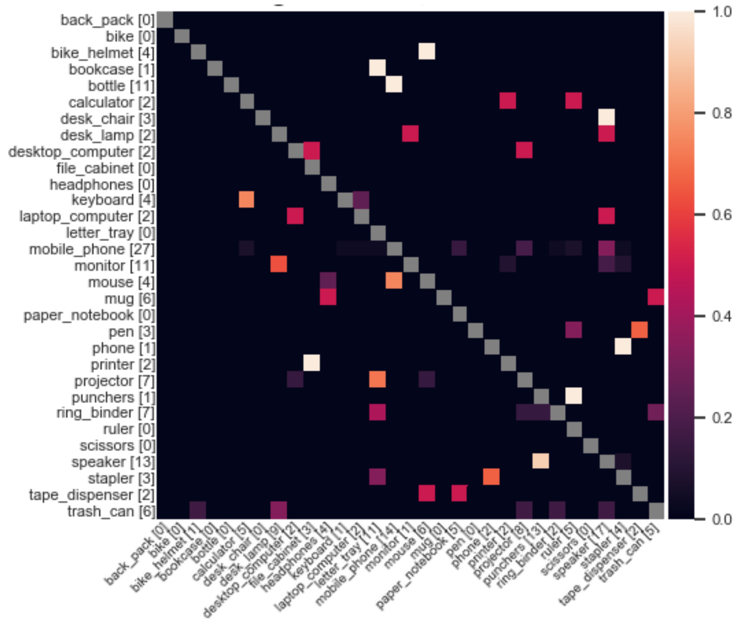
(b) Diagonal masked and normalized

Figure 6.1: Normalized confusion matrices for $\mathbf{A} \rightarrow \mathbf{W}$ transfer task on the ResNet-50 model. In matrix (b), the diagonal entries are masked out to highlight the misclassifications.

Confusion Matrices – Domain Adversarial Neural Network



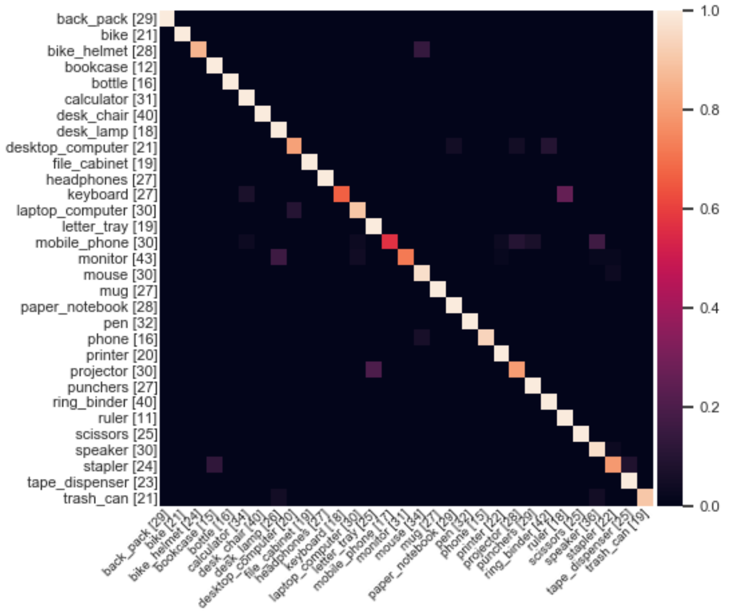
(a) Normalized



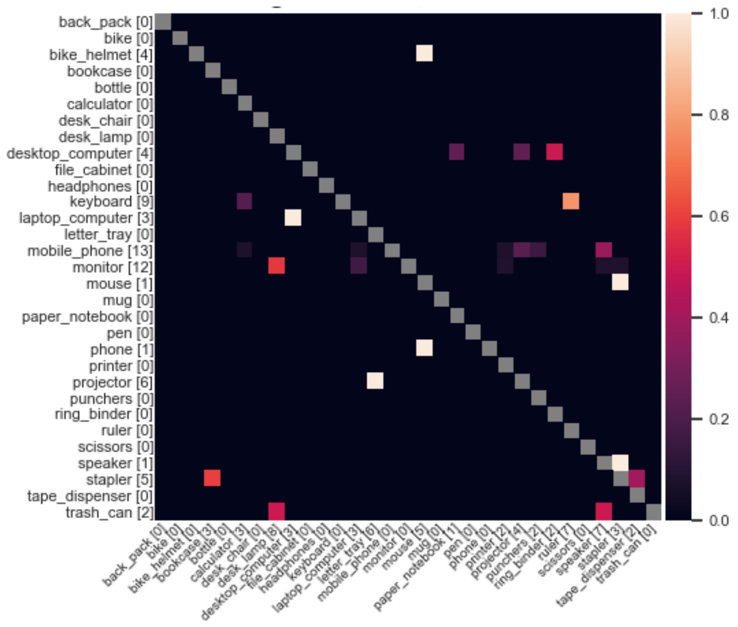
(b) Diagonal masked and normalized

Figure 6.2: Normalized confusion matrices for **A** $\rightarrow$ **W** transfer task on the Domain Adversarial Neural Network model. In matrix (b), the diagonal entries are masked out to highlight the misclassifications.

Confusion Matrices – Domain Adaptive Fusion

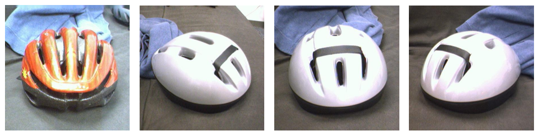


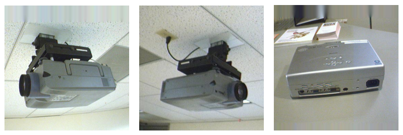
(a) Normalized

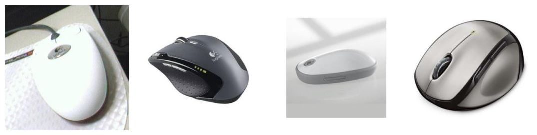


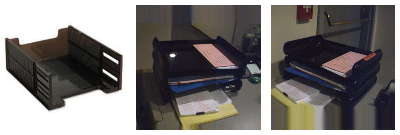
(b) Diagonal masked and normalized

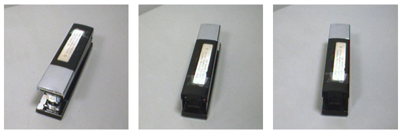
Figure 6.3: Normalized confusion matrices for **A** $\rightarrow$ **W** transfer task on the DAF model. In matrix (b), the diagonal entries are masked out to highlight the misclassifications.


(a) Helmets classified as computer mice


(c) Projectors classified as trays


(b) Actual computer mice


(d) Actual paper trays


(e) Staplers classified as punchers

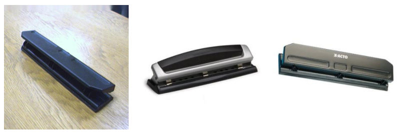
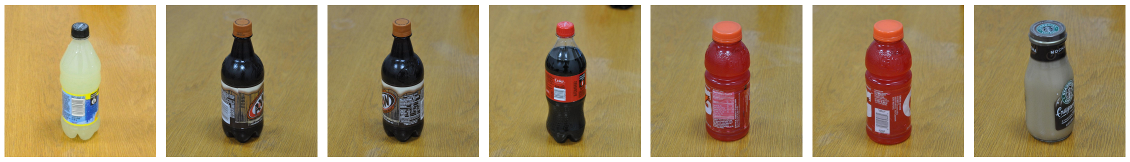

(f) Actual punchers

Figure 6.4: Comparison of incorrectly classified images with samples from their predicted classes on the $\mathbf{A} \rightarrow \mathbf{W}$ transfer task.

Noting that many samples are also incorrectly predicted as the `bottle` class, yet the recall for actual bottles on the target domain is low, we visualize bottle images from both domains for further analysis (see Figure 6.5). It can clearly be seen that the Amazon domain contains samples that are largely mislabeled or loosely labeled as compared to the DSLR domain, which explains the high error rate contributed by these classifications. Inspection of other Amazon classes reveals similar problems with the dataset.


(a) DSLR bottles

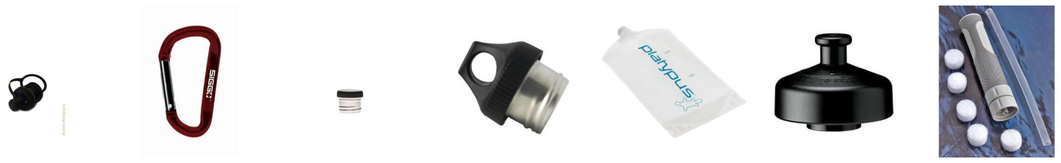

(b) Amazon bottles

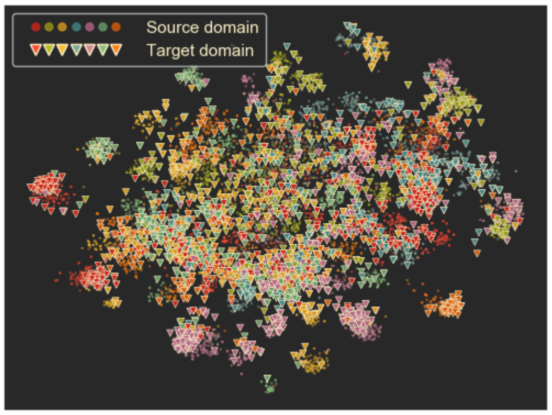
Figure 6.5: Comparison of samples from the `bottle` class between the DSLR and Amazon domains

6.2.2 Feature Clustering

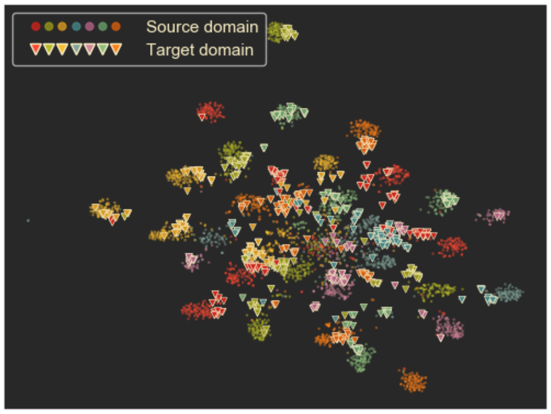
By sharpening the soft pseudo-labels of the target data during training, it was our hope that the learned feature representations for the target data would form distinctive clusters for each class in the dataset. Furthermore, successfully aligning the source and target domains should produce features – and therefore clusters – that are indistinguishable between the domains. We visualize the learned feature space using t-SNE embeddings of the **Cl** $\rightarrow$ **Pr** transfer task (65 classes) and the **A** $\rightarrow$ **W** transfer task (31 classes) for the Resnet-50, DANN, and DAF models (shown in Figure 6.6).

The resulting plots of these t-SNE embeddings perfectly matched our expectations, with DAF producing 31 clearly separated clusters on the **A** $\rightarrow$ **W** task 6.6d, of which the target and source features for each class appear to be primarily nested within the same clusters. The successful clustering of DAF is even more apparent on the **Cl** $\rightarrow$ **Pr** task – while both ResNet-50 (fig. 6.6a) and DANN (fig. 6.6b) struggled to produce coherent clusters, the t-SNE embeddings from the DAF model remain highly

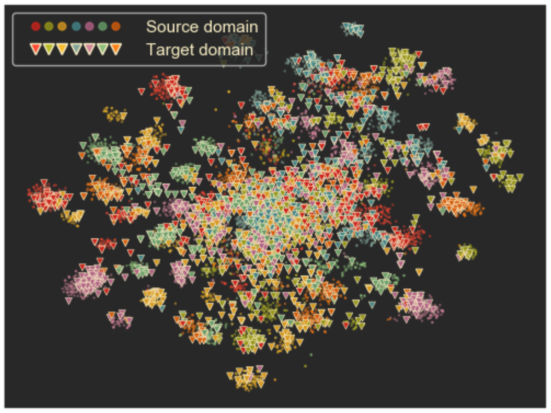
Cl → Pr **A → W**



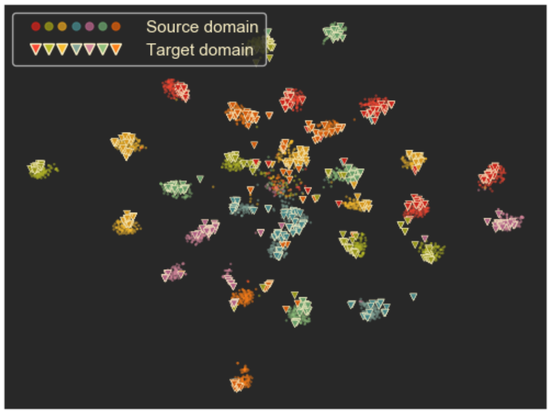
(a) ResNet-50



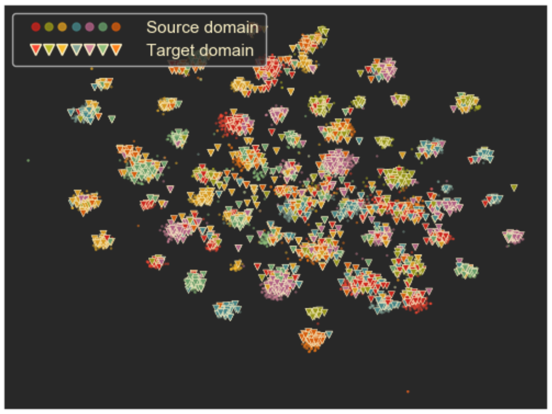
(d) ResNet-50



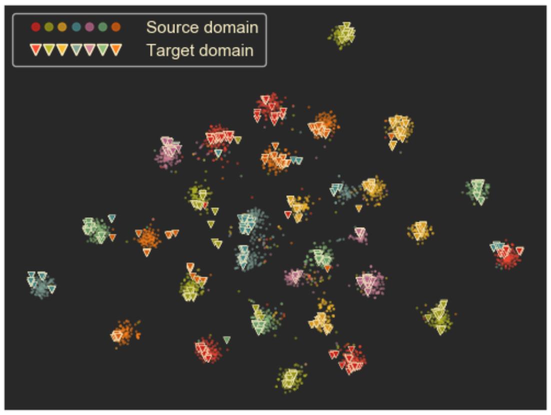
(b) DANN



(e) DANN



(c) DAF



(f) DAF

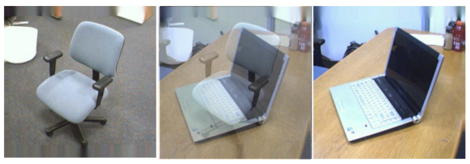
Figure 6.6: t-SNE embeddings for **Cl** $\rightarrow$ **Pr** ((a), (b), (c)) and **A** $\rightarrow$ **W** ((d), (e), (f)) for ResNet-50, DANN, and DAF models.

discriminable (fig. 6.6c).

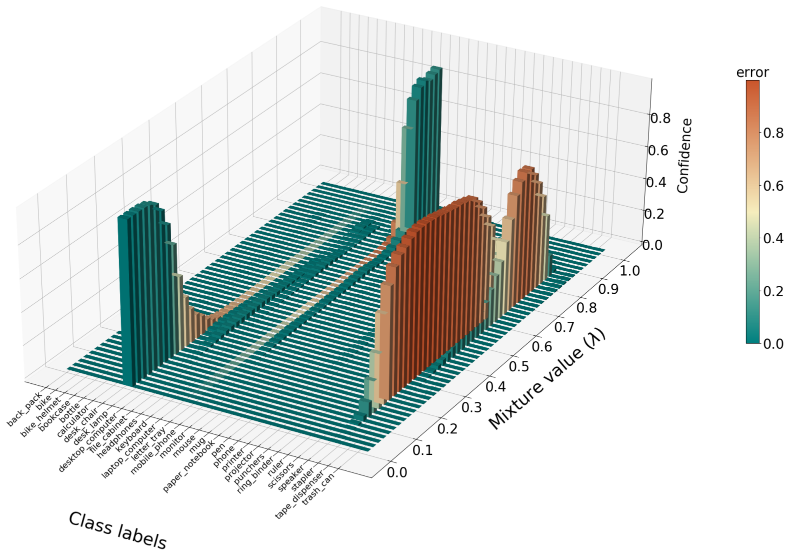
6.2.3 Linear Continuity

One of the key contributions of this work is the fusion of data from separate domains to impose linear continuity between convex combinations of input features with their corresponding predictions. To analyze the impact of this regularization technique, we visualized the prediction vectors of the ResNet-50, DANN, and DAF models by fusing two unseen target samples while varying the mixture value from 0 to 1 (see Figure 6.7).

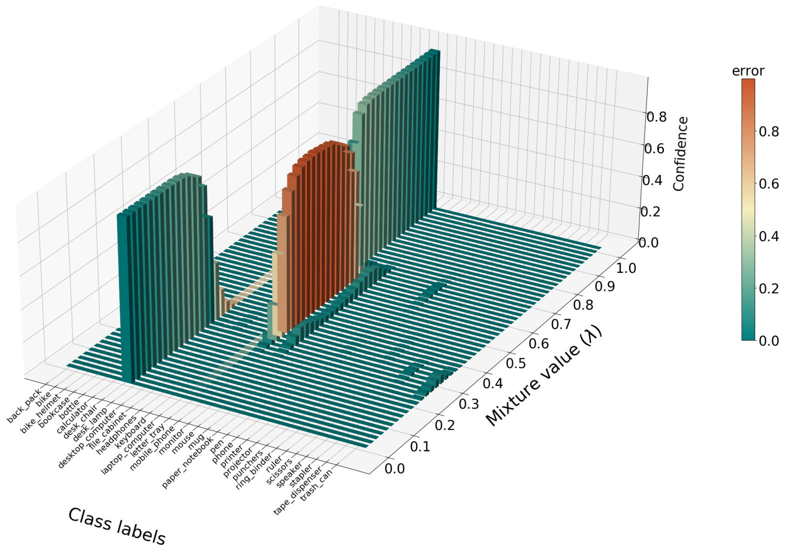
On these samples, all three models correctly classify the unfused input images ($\lambda = 0$ and $\lambda = 1$); however, the prediction error and entropy of both the ResNet-50 model (fig. 6.7b) and the DANN model (fig. 6.7c) increased when confronted with fused samples. On the other hand, the multi-class predictions produced by our model trained using Domain Adaptive Fusion (fig. 6.7d) closely followed the fused ground-truth labels, maintaining correct primary class predictions throughout the varied fusion levels, with approximately equal probabilities of the correct classes predicted on the image produced by an equal fusion of the two input images.



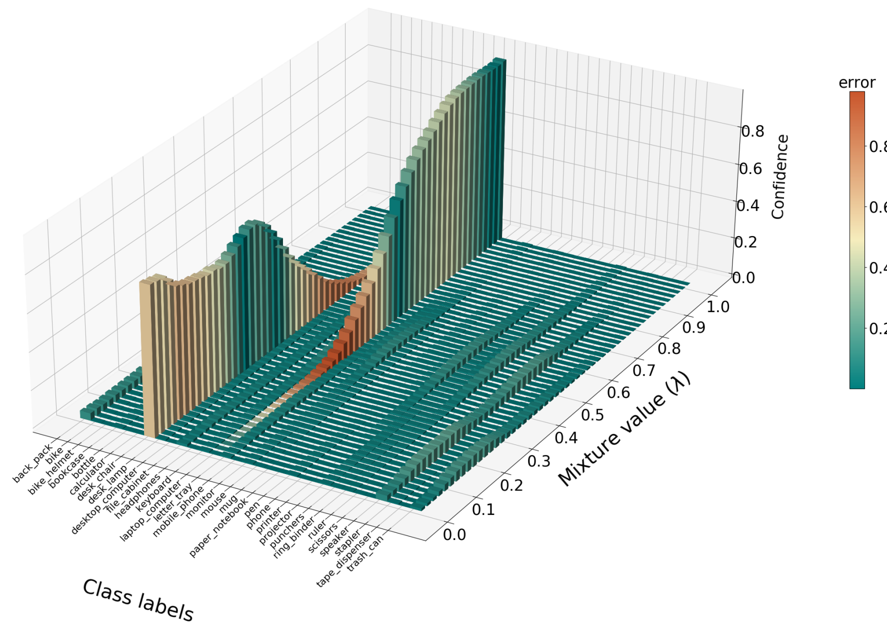
(a) Example of input fusion on chair and laptop samples with $\lambda = 0, 0.5,$ and 1



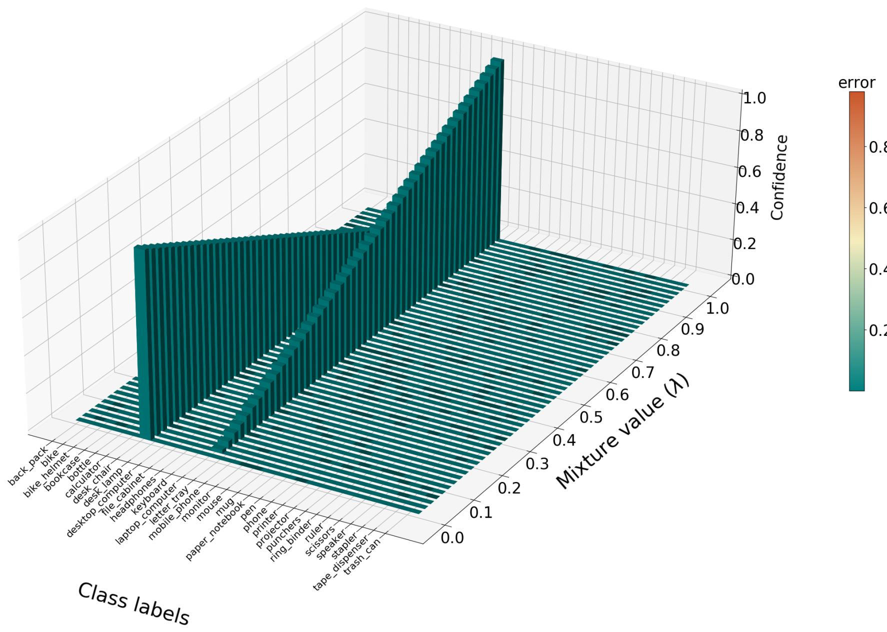
(b) ResNet-50



(c) DANN



(d) DAF



(e) Ground Truth

Figure 6.7: Visualization of the prediction vectors with varying mixture values λ on the (b) ResNet-50, (c) DANN, and (d) DAF models. *(Continued on the following page.)*

Figure 6.7: Subfigure (a) illustrates the fused sample inputs with $\lambda = 0, 0.5,$ and 1, and (e) displays the convex combinations of the ground truth labels, representing the desired outputs in the case of perfect linear behavior of the classifier over fused samples. Each label in the prediction vector is colored based on the absolute error between the class prediction and the ground truth.

Chapter 7

CONCLUSIONS

Using a hybrid approach to fuse domain adaptation with principles of semi-supervised learning, the Domain Adaptive Fusion architecture is able to successfully bridge the source and target domains, producing competitive results on challenging computer vision classification tasks. By encouraging cross-domain linear behavior of the classification function between convex combinations of input images and their predictions, the surface of the function between these samples is effectively smoothed, facilitating the alignment of the domain distributions. Deeper analysis of the DAF model's classification error, clustering, and predictive function supports these claims, and also highlights where mislabeled and imbalanced data in the experimental datasets is likely to contribute to poor performance on a few of the transfer tasks.

The compelling results of the work proposed in this thesis should not be seen as the limit of the fusion approach for domain adaptation. The fusion of deeper feature representations from the network may expand the uses of the DAF architecture beyond the pixel space, and fusions of extracted features with learned categorical prototypes may further enhance the desired regularization characteristics produced by the algorithm. While DAF currently requires the use of the original input images for domain alignment, it may also be possible to the modify adversarial domain discriminator to instead learn the mixture value (λ) in order to directly encourage domain invariance of fused samples. Modifications to α, the hyperparameter that controls the distribution sampled from to generate λ, has also yet to be explored. It is hypothesized that ramping up this value at the beginning of the training process could yield improved training results.

REFERENCES

Belkin, M. and P. Niyogi, “Semi-supervised learning on riemannian manifolds”, Machine learning **56**, 1-3, 209–239 (2004).

Berthelot, D., N. Carlini, I. Goodfellow, N. Papernot, A. Oliver and C. Raffel, “Mixmatch: A holistic approach to semi-supervised learning”, arXiv preprint arXiv:1905.02249 (2019).

Cao, Z., L. Ma, M. Long and J. Wang, “Partial adversarial domain adaptation”, in “Proceedings of the European Conference on Computer Vision (ECCV)”, pp. 135–150 (2018).

Cao, Z., K. You, M. Long, J. Wang and Q. Yang, “Learning to transfer examples for partial domain adaptation”, arXiv preprint arXiv:1903.12230 (2019).

Chapelle, O., B. Schlkopf and A. Zien, *Semi-Supervised Learning* (The MIT Press, 2010), 1st edn.

Chapelle, O. and A. Zien, “Semi-supervised classification by low density separation.”, in “AISTATS”, vol. 2005, pp. 57–64 (Citeseer, 2005).

Chen, C., Q. Dou, H. Chen and P.-A. Heng, “Semantic-aware generative adversarial nets for unsupervised domain adaptation in chest x-ray segmentation”, in “International Workshop on Machine Learning in Medical Imaging”, pp. 143–151 (Springer, 2018).

Chen, Y.-H., W.-Y. Chen, Y.-T. Chen, B.-C. Tsai, Y.-C. Frank Wang and M. Sun, “No more discrimination: Cross city adaptation of road scene segmenters”, in “Proceedings of the IEEE International Conference on Computer Vision”, pp. 1992–2001 (2017).

Dalal, N. and B. Triggs, “Histograms of oriented gradients for human detection”, (2005).

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database”, in “CVPR09”, (2009).

Ganin, Y. and V. Lempitsky, “Unsupervised domain adaptation by backpropagation”, arXiv preprint arXiv:1409.7495 (2014).

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

Gong, B., Y. Shi, F. Sha and K. Grauman, “Geodesic flow kernel for unsupervised domain adaptation”, in “2012 IEEE Conference on Computer Vision and Pattern Recognition”, pp. 2066–2073 (IEEE, 2012).

Goodfellow, I., Y. Bengio and A. Courville, *Deep learning* (MIT press, 2016).

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, “Generative adversarial nets”, in “Advances in neural information processing systems”, pp. 2672–2680 (2014).

Grandvalet, Y. and Y. Bengio, “Semi-supervised learning by entropy minimization”, in “Advances in neural information processing systems”, pp. 529–536 (2005).

He, K., X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition”, in “Proceedings of the IEEE conference on computer vision and pattern recognition”, pp. 770–778 (2016).

Hoffman, J., E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros and T. Darrell, “Cycada: Cycle-consistent adversarial domain adaptation”, arXiv preprint arXiv:1711.03213 (2017).

Kim, T., M. Cha, H. Kim, J. K. Lee and J. Kim, “Learning to discover cross-domain relations with generative adversarial networks”, in “Proceedings of the 34th International Conference on Machine Learning-Volume 70”, pp. 1857–1865 (JMLR. org, 2017).

Kingma, D. P. and J. Ba, “Adam: A method for stochastic optimization”, arXiv preprint arXiv:1412.6980 (2014).

Krogh, A. and J. A. Hertz, “A simple weight decay can improve generalization”, in “Advances in neural information processing systems”, pp. 950–957 (1992).

Laine, S. and T. Aila, “Temporal ensembling for semi-supervised learning”, arXiv preprint arXiv:1610.02242 (2016).

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition”, Neural computation **1**, 4, 541–551 (1989).

Lee, D.-H., “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks”, in “Workshop on Challenges in Representation Learning, ICML”, vol. 3, p. 2 (2013).

Li, Y., L. Liu and R. T. Tan, “Certainty-driven consistency loss for semi-supervised learning”, arXiv preprint arXiv:1901.05657 (2019).

Long, M., Y. Cao, J. Wang and M. I. Jordan, “Learning transferable features with deep adaptation networks”, arXiv preprint arXiv:1502.02791 (2015).

Long, M., Z. Cao, J. Wang and M. I. Jordan, “Conditional adversarial domain adaptation”, in “Advances in Neural Information Processing Systems”, pp. 1640–1650 (2018).

Long, M., H. Zhu, J. Wang and M. I. Jordan, “Unsupervised domain adaptation with residual transfer networks”, in “Advances in Neural Information Processing Systems”, pp. 136–144 (2016).

Long, M., H. Zhu, J. Wang and M. I. Jordan, “Deep transfer learning with joint adaptation networks”, in “Proceedings of the 34th International Conference on Machine Learning-Volume 70”, pp. 2208–2217 (JMLR. org, 2017).

Miyato, T., S.-i. Maeda, M. Koyama, K. Nakae and S. Ishii, “Distributional smoothing with virtual adversarial training”, arXiv preprint arXiv:1507.00677 (2015).

Ng, A. Y., “Feature selection, l 1 vs. l 2 regularization, and rotational invariance”, in “Proceedings of the twenty-first international conference on Machine learning”, p. 78 (ACM, 2004).

Pan, S. J. and Q. Yang, “A survey on transfer learning”, IEEE Transactions on knowledge and data engineering **22**, 10, 1345–1359 (2009).

Panareda Busto, P. and J. Gall, “Open set domain adaptation”, in “Proceedings of the IEEE International Conference on Computer Vision”, pp. 754–763 (2017).

Pei, Z., Z. Cao, M. Long and J. Wang, “Multi-adversarial domain adaptation”, in “Thirty-Second AAAI Conference on Artificial Intelligence”, (2018).

Pinheiro, P. O., “Unsupervised domain adaptation with similarity learning”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 8004–8013 (2018).

Saenko, K., B. Kulis, M. Fritz and T. Darrell, “Adapting visual category models to new domains”, in “European conference on computer vision”, pp. 213–226 (Springer, 2010).

Sankaranarayanan, S., Y. Balaji, C. D. Castillo and R. Chellappa, “Generate to adapt: Aligning domains using generative adversarial networks”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 8503–8512 (2018).

Shen, J., Y. Qu, W. Zhang and Y. Yu, “Wasserstein distance guided representation learning for domain adaptation”, arXiv preprint arXiv:1707.01217 (2017).

Tarvainen, A. and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results”, in “Advances in neural information processing systems”, pp. 1195–1204 (2017).

Tsai, Y.-H., W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang and M. Chandraker, “Learning to adapt structured output space for semantic segmentation”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 7472–7481 (2018).

Tzeng, E., J. Hoffman, K. Saenko and T. Darrell, “Adversarial discriminative domain adaptation”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 7167–7176 (2017).

Venkateswara, H., *Domain Adaptive Computational Models for Computer Vision*, Ph.D. thesis, Arizona State University (2017).

Venkateswara, H., J. Eusebio, S. Chakraborty and S. Panchanathan, “Deep hashing network for unsupervised domain adaptation”, in “The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)”, (2017a).

Venkateswara, H., J. Eusebio, S. Chakraborty and S. Panchanathan, “Deep hashing network for unsupervised domain adaptation”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 5018–5027 (2017b).

Verma, V., A. Lamb, J. Kannala, Y. Bengio and D. Lopez-Paz, “Interpolation consistency training for semi-supervised learning”, arXiv preprint arXiv:1903.03825 (2019).

Wang, J., Y. Chen, H. Yu, M. Huang and Q. Yang, “Easy transfer learning by exploiting intra-domain structures”, in “IEEE International Conference on Multimedia Expo (ICME)”, (2019a).

Wang, X., L. Li, W. Ye, M. Long and J. Wang, “Transferable attention for domain adaptation”, (2019b).

Wang, X. and X. Wang, “Unsupervised domain adaptation with coupled generative adversarial autoencoders”, Applied Sciences **8**, 12, 2529 (2018).

You, K., M. Long, Z. Cao, J. Wang and M. I. Jordan, “Universal domain adaptation”, in “The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)”, (2019).

Yun, S., D. Han, S. J. Oh, S. Chun, J. Choe and Y. Yoo, “Cutmix: Regularization strategy to train strong classifiers with localizable features”, arXiv preprint arXiv:1905.04899 (2019).

Zhang, H., M. Cisse, Y. N. Dauphin and D. Lopez-Paz, “mixup: Beyond empirical risk minimization”, arXiv preprint arXiv:1710.09412 (2017).

Zhang, J., Z. Ding, W. Li and P. Ogunbona, “Importance weighted adversarial nets for partial domain adaptation”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 8156–8164 (2018).

Zhou, D., O. Bousquet, T. N. Lal, J. Weston and B. Schölkopf, “Learning with local and global consistency”, in “Advances in neural information processing systems”, pp. 321–328 (2004).

Zhu, J.-Y., T. Park, P. Isola and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks”, in “Proceedings of the IEEE international conference on computer vision”, pp. 2223–2232 (2017).