Adapting Across Domains by Aligning Representations and Images

Eric Tzeng

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2020-69
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-69.html

May 27, 2020

Adapting Across Domains by Aligning Representations and Images

by

Eric S Tzeng

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Trevor Darrell, Chair
Professor Alexei A. Efros
Professor Bruno Olshausen

Spring 2020

Adapting Across Domains by Aligning Representations and Images

Abstract

Adapting Across Domains by Aligning Representations and Images

by

Eric S Tzeng

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Trevor Darrell, Chair

The advent of deep convolutional networks has powered a new wave of progress in visual recognition. These learned representations vastly outperform hand-engineered features, achieving much higher performance on visual tasks while generalizing better across datasets. However, as general as these models may seem, they still suffer when there is a mismatch between the data they were trained on and the data they are being asked to operate on. Domain adaptation offers a potential solution, allowing us to *adapt* networks from the source domain that they were trained on to new target domains, where labeled data is sparse or entirely absent. However, before the rise of end-to-end learnable representations, visual domain adaptation techniques were largely limited to adapting classifiers trained on top of fixed, hand-designed visual features. In this thesis, we show how visual domain adaptation can be integrated with deep learning to directly learn representations that are resilient to domain shift, thereby enabling models to generalize beyond the source domain.

In Chapter 2 we demonstrate how we can design losses that attempt to measure how different two domains are. We show that by optimizing representations to minimize these losses, we can learn representations that generalize better from source to target.

In Chapters 3 and 4, rather than hand-designing these domain losses, we show that we can train models that attempt to measure domain discrepancy. Since these models are themselves end-to-end learnable, we can backpropagate through them to learn representations that minimize the learned discrepancy. This is similar in concept to Generative Adversarial Networks [Goo+14], and we additionally explore the relationship between the two and how we can use techniques developed for GANs in an adversarial setting as well.

Finally, in Chapters 5 and 6, we show that adaptation need not be limited to intermediate features of a deep network. Adversarial adaptation techniques can also be used to train models that directly alter the pixels of images, transforming them into cross-domain analogues. These transformed images can then be used as a labeled pseudo-target dataset to learn supervised models better suited for the target domain. We show that this technique is complementary to feature-based adaptation, yielding even better performance when the two are combined.

To my parents, Ada and John

Contents

Acknowledgments

My time at Berkeley has been, without a doubt, the most memorable experience of my life. Thank you all to the countless people who made my time here so unforgettable.

Thank you to my advisor, Trevor Darrell, for enabling me to embark on this journey at all by taking me on as a student. You provided no shortage of enthusiasm and fruitful research ideas, while also giving me the freedom to explore the research ideas I was most excited about. Thank you for teaching me so much, and for making my time at Berkeley with you and the group so memorable.

Thank you to Kate Saenko and Judy Hoffman. Kate, thank you for all of your research ideas and paper-writing advice over the years. And Judy, thank you for showing me the ropes when I first joined the group—even if you were still a student at the time, you were like an advisor to me, and you've been hugely influential in the way I conduct research today. Together, the two of you have taught me almost everything I know about domain adaptation, and none of the work in this dissertation would have been possible without you.

Thanks to the professors who served on my committees: Jitendra Malik, for helping with my qualifying examination, and Alexei Efros and Bruno Olshausen, for helping with both my qualifying examination and my dissertation. You have all been enormously patient and helpful during the entire process, and your feedback and encourgement has helped shape my work into what it is today.

Thank you to my fellow friends, colleagues, and collaborators at Berkeley. Being with all of you has been such an enriching experience, from the productive research talks to the silly small talk we've had in the hallways. The memories I formed with all of you will stay with me forever. In particular, I'd like to thank those who were an especially important part of my PhD experience, whether in collaboration or in companionship: Evan Shelhamer, Lisa Anne Hendricks, Jeff Donahue, Jon Long, Alex Lee, Seth Park, Kaylee Burns, Samaneh Azadi, Parsa Mahmoudieh, Sayna Ebrahimi, and Devin Guillory.

Thank you to my wonderfully talented team at Pinterest, especially those I have worked with closely: Andrew Zhai, Dmitry Kislyuk, Rex Wu, Josh Beal, David Xue, and Nadia Fawaz. It's been a wonderful experience approaching computer vision from a more practical angle, and working with the team has been an incredibly fulfilling experience.

Thank you to all of my friends outside of school, who have provided valuable perspective while patiently listening to me, as I vented my frustrations and celebrated my successes. Once again, this list feels endless, but I'd like to especially thank Yong Hoon Lee, Flora Ting, Alex Texter, Michelle Bu, Julie De Lorenzo, Christina Morris, and Glenn Battishill.

Thank you to my family, for their enduring love and patience during this entire experience. It's easy to spend a lot of time thanking all of the people I interacted with at Berkeley, but I only made it here because of their love and support as I was growing up.

Finally, no matter how long I spend on this acknowledgments section, there simply isn't enough space to list out the countless people who have helped me make it this far. Even if I did not mention you by name, please know that if I've ever talked to you or confided in you about the ups and downs of graduate school, I'm thinking of you as I write this.

Chapter 1

Introduction

In recent years, we have seen great strides in visual recognition on a variety of tasks ranging from classification to object detection to segmentation. At the heart of these advances is the paradigm of supervised learning, where we collect a labeled training dataset, then train models to reproduce the labels when given the corresponding input. Supervised learning relies heavily on the assumption that we can collect this training dataset so that it closely matches the test data distribution. If we train a model that is able to accurately reproduce the labels on the training set, then because of this assumption, it should perform well on the test data as well.

Domain adaptation seeks to complicate this problem by introducing an explicit assumption that the test data *does not* come from the same distribution as the training data. We commonly refer to the training data distribution as the *source domain*, the test data distribution as the *target domain*, and the differences between them as *domain shift*. Generally, the source domain is large and well-labeled, and thus supervised learning can produce models that perform well on source data. However, because of domain shift, models that are trained exclusively on source data do not generalize well to the target domain. Additionally, the target domain typically either has only a few available labels (*supervised adaptation*), or possibly no labels at all (*unsupervised adaptation*). Thus, it is either difficult or outright impossible to train supervised models exclusively on the target data and have them perform well in the target domain. Domain adaptation can then be succinctly summarized as the problem of how to adapt models trained on source data to the target domain, in the hopes that we can mitigate or undo the effects of domain shift.

While this problem may seem contrived at first glance, in actuality domain adaptation problems are a fundamental part of practical machine learning. In real-world settings, the test data rarely, if ever, exactly matches the data seen during training. One particularly illustrative example is the "Name That Dataset!" game introduced by Torralba and Efros [TE11], which demonstrates that models trained on one dataset generalize poorly to examples from another dataset, even when tested on classes present in both datasets. While many of our datasets attempt to capture the way the world looks, in truth it is impossible to fully capture the space of natural images in a single dataset. Recognizing these shortcomings and explicitly accounting for them during training is a crucial step in ensuring that models can generalize to the real world. Acknowledging the need for adaptation also opens up exciting new possibilities, such as the ability to train models on an infinite amount of synthetic imagery, then adapt these models to work on real imagery without the need for costly dataset labeling.

Previously, visual domain adaptation techniques were built on top of hand-engineered visual features. These features were designed to capture and summarize high-level semantics within images so that standard machine learning techniques could be applied afterwards, and they were typically used as black boxes. As a result, visual domain adaptation methods were generally unable to directly adapt these features, and operated primarily by modifying the classifiers learned on top of them.

However, in 2012, Krizhevsky et al. [KSH12] set a new state of the art on the ImageNet classification challenge [BDFF12] by over 40%, and with it came a powerful new tool for learning representations entirely from scratch. Rather than relying on hand-designed features, for the first time, backpropagation through a convolutional neural network enabled us to learn representations directly on input pixel intensities that were well-suited for the specific supervised tasks at hand. This method would form the basis of the vast majority of computer vision research for the next few years. However, while these learned representations were effective across a wide variety of computer vision tasks, they too suffered from the effects of domain shift, and achieving optimal performance on non-ImageNet datasets typically mandated an additional round of fine-tuning [Don+14].

While fine-tuning can adjust network parameters to better operate on new data, the sheer size of these networks often necessitates large amounts of labeled training data, which can be difficult to procure. Thus, as an alternative, in this thesis we explore the intersection of deep learning and visual domain adaptation. With these powerful tools at our disposal, how can we directly learn representations that are resistant to domain shift, even when labeled target data is scarce or entirely absent?

We begin in Chapter 2 by demonstrating how convolutional networks can be used to directly optimize for representations that adapt better between domains. We propose the concept of a *domain alignment* loss that attempts to measure how large the shift between domains is. For an initial attempt, we simply use the difference between the means of batches and source and target data as our domain alignment loss, but many other alternatives would be valid options. We can then directly optimize deep representations to minimize this loss, thereby bringing source and target representations closer together. In turn, this facilitates the transfer of classifiers trained on source data into the target domain, leading to improved target accuracy during inference. While straightforward, this approach is quite effective, and easily outperforms previous adaptation methods that operated on fixed, hand-designed features.

In Chapter 3, we extend upon this idea of directly optimizing representations against a measure of the domain discrepancy. However, rather than relying on hand-designed functions to estimate the discrepancy, we instead learn this estimator via backpropagation through a convolutional network. We train a domain discriminator that attempts to classify which domain images originate from, while the representation is optimized to maximize the error of the domain discriminator. We also demonstrate that, in settings where target labels are available, we can use the structure inherent within source classifier predictions to transfer the semantic relationships between classes into the source domain, further improving adaptation performance.

Chapter 4 explores the relationship between adversarial adaptation approaches and Generative Adversarial Networks [Goo+14]. We observe that adversarial adaptation is similar in concept to a GAN: adversarial adaptation attempts to map target images into a pre-trained source representation by fooling a discriminator, much like how a GAN attempts to map randomly sampled latent vectors into images that mimic real ones. We take this analogy to its logical conclusion, showing how many of the procedures and techniques intended for training GANs can be incorporated into an adversarial adaptation setting. With this additional insight, our resulting model achieves stronger performance on unsupervised adaptation tasks than previous adversarial methods.

Chapter 5 demonstrates that adversarial adaptation approaches are not limited to operating only on internal representations of convolutional networks. We show that adversarial approaches can be applied at the pixel level as well by training image-to-image generators that map images between domains. By mapping source images into the target domain while preserving their semantic identities, we can effectively generate a “target-like” dataset with labels that can be used to train target models. This technique alone already proves to be surprisingly effective on a variety of classification and segmentation adaptation tasks. However, we additionally demonstrate that pixel-level adaptation is complementary to feature-level adaptation, and performing both in concert leads to even stronger performance than either variant alone.

In Chapter 6 we propose a simpler variant of the previous pixel-adaptation method. By exploiting the rich structure of object detection and semantic segmentation models during training, we are able to more effectively constrain the model so that preserves semantic information during cross-domain generation. In turn, this allows us to shrink the model drastically without sacrificing accuracy on the final task, which leads to an almost four-fold speedup in the runtime of the model. We also empirically validate our method on a car detection task, demonstrating that pixel-adaptation methods are effective in object detection settings as well.

Finally, in Chapter 7 we summarize the findings of this thesis, and discuss possible extensions and future avenues for further research.

Chapter 2

Learning Adaptive Representations via Mean Alignment

In this chapter we introduce a simple method for end-to-end learning of representations that are robust to the negative effects of domain shift. Dataset bias was classically illustrated in computer vision by way of the “name the dataset” game of Torralba and Efros [TE11]. Indeed, this turns out to be formally connected to measures of domain discrepancy [KBDG04; Bor+06]. Optimizing for domain invariance, therefore, can be considered equivalent to the task of learning to predict the class labels while simultaneously finding a representation that makes the domains appear as similar as possible. This principle forms the essence of our proposed approach. We learn deep representations by optimizing over a loss which includes both classification error on the labeled data as well as a domain alignment loss which seeks to make the domains indistinguishable.

We propose a new CNN architecture, outlined in Figure 2.1, which uses an adaptation layer along with a domain alignment loss based on maximum mean discrepancy (MMD) [Bor+06] to automatically learn a representation jointly trained to optimize for classification and domain invariance. We show that our domain alignment metric can be used both to select the dimension of the adaptation layers, choose an effective placement for a new adaptation layer within a pre-trained CNN architecture, and fine-tune the representation.

Our architecture can be used to solve both *supervised adaptation*, when a small amount of target labeled data is available, and *unsupervised adaptation*, when no labeled target training data is available. We provide a comprehensive evaluation on the popular Office benchmark for classification across visually distinct domains [Sae+10]. We demonstrate that by jointly optimizing for domain alignment and classification, we are able to significantly outperform baselines that cannot learn representations end-to-end. In fact, for the case of minor pose, resolution, and lighting changes, our algorithm is able to achieve 96% accuracy on the target

This chapter is based on joint work with Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell [Tze+14].

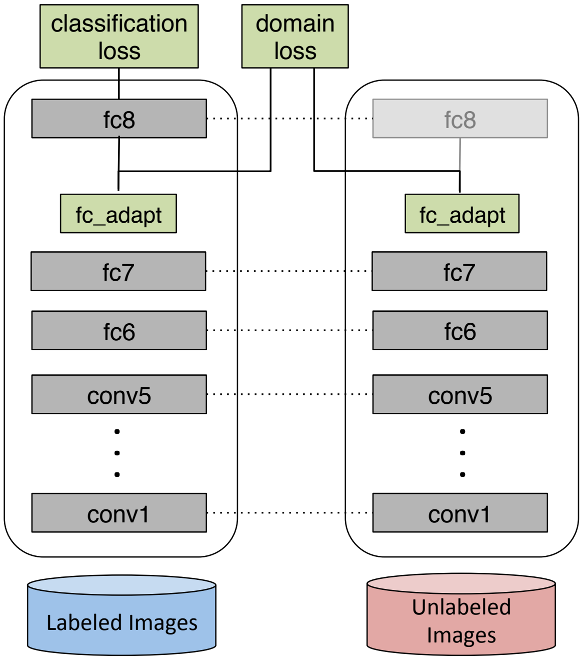


Figure 2.1: Our architecture optimizes a deep CNN for both classification loss as well as domain invariance. The model can be trained for *supervised* adaptation, when there is a small amount of target labels available, or *unsupervised* adaptation, when no target labels are available. We introduce domain invariance through guided selection of the depth and width of the adaptation layer, as well as an additional domain loss term during fine-tuning that directly minimizes the distance between source and target representations.

domain, demonstrating that we have in fact learned a representation that is invariant to these biases.

2.1 Related work

The concept of visual dataset bias was popularized in [TE11]. There have been many approaches proposed in recent years to solve the visual domain adaptation problem. All recognize that there is a shift in the distribution of the source and target data representations. In fact, the size of a domain shift is often measured by the distance between the source and target subspace representations [Bor+06; Fer+13; KBDG04; MMR09; Pan+09]. A large number of methods have sought to overcome this difference by learning a feature space transformation to align the source and target representations [Sae+10; KSD11; Fer+13; Gon+12]. For the *supervised* adaptation scenario, when a limited amount of labeled data is available in the target domain, some approaches have been proposed to learn a target classifier regularized against the source classifier [YYH07; AZ11; AB10]. Others have sought to both learn a feature transformation and regularize a target classifier simultaneously [Hof+13; DXT12].

Recently, supervised convolutional neural network (CNN) based feature representations have been shown to be extremely effective for a variety of visual recognition tasks [KSH12; Don+14; Gir+13; Ser+13]. In particular, using deep representations dramatically reduce the effect of resolution and lighting on domain shifts [Don+14; Hof+14a].

Parallel CNN architectures such as Siamese networks have been shown to be effective for learning invariant representations [Bro+93; CHL05]. However, training these networks requires labels for each training instance, so it is unclear how to extend these methods to unsupervised settings.

Multimodal deep learning architectures have also been explored to learn representations that are invariant to different input modalities [Ngi+11]. However, this method operated primarily in a generative context and therefore did not leverage the full representational power of supervised CNN representations.

Training a joint source and target CNN architecture was proposed by [CBG13], but was limited to two layers and so was significantly outperformed by the methods which used a deeper architecture [KSH12], pre-trained on a large auxiliary data source (ex: ImageNet [BDFF12]).

[GKZ14] proposed pre-training with a denoising auto encoder, then training a two-layer network simultaneously with the MMD domain alignment loss. This effectively learns a domain invariant representation, but again, because the learned network is relatively shallow, it lacks the strong semantic representation that is learned by directly optimizing a classification objective with a supervised deep CNN.

2.2 Training CNN-based domain invariant representations

We introduce a new convolutional neural network (CNN) architecture which we use to learn a visual representation that is both domain invariant and which offers strong semantic separation. It has been shown that a pre-trained CNN can be adapted for a new task through fine-tuning [Gir+13; Ser+13; Hof+14b]. However, in the domain adaptation scenario there is little, or no, labeled training data in the target domain so we can not directly fine-tune for the categories of interest, C in the target domain, T. Instead, we will use data from a related, but distinct source domain, S, where more labeled data is available from the corresponding categories, C.

Directly training a classifier using only the source data often leads to overfitting to the source distribution, causing reduced performance at test time when recognizing in the target domain. Our intuition is that if we can learn a representation that minimizes the distance between the source and target distributions, then we can train a classifier on the source labeled data and directly apply it to the target domain with minimal loss in accuracy.

To minimize this distance, we consider the standard distribution distance metric, Maximum Mean Discrepancy (MMD) [Bor+06]. This distance is computed with respect to a

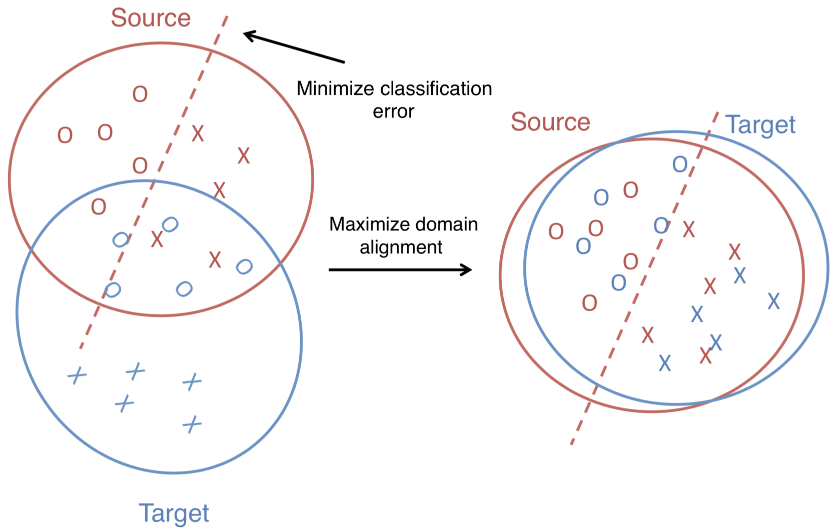


Figure 2.2: For biased datasets (left), classifiers learned in a source domain do not necessarily transfer well to target domains. By optimizing an objective that simultaneously minimizes classification error and maximizes domain alignment (right), we can learn representations that are discriminative and domain invariant.

particular representation, $\phi(\cdot)$. In our case, we define a representation, $\phi(\cdot)$, which operates on source data points, $x_s \in X_S$, and target data points, $x_t \in X_T$. Then an empirical approximation to this distance is computed as

$$\mathrm{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\|. \tag{2.1}$$

As Figure 2.2 shows, not only do we want to minimize the distance between domains (or maximize the domain alignment), but we want a representation which is conducive to training strong classifiers. Such a representation would enable us to learn strong classifiers that readily transfer across domains. One approach to meeting both these criteria is to minimize the loss

$$\mathcal{L} = \mathcal{L}_C(X_L, y) + \lambda \mathrm{MMD}^2(X_S, X_T) \tag{2.2}$$

where $\mathcal{L}_C(X_L, y)$ denotes classification loss on the available labeled data, X_L, and the ground truth labels, y, and $\mathrm{MMD}(X_S, X_T)$ denotes the distance between the source data, X_S, and the target data, X_T. The hyperparameter λ determines how strongly we would like to confuse the domains.

One approach to minimizing this loss is to take a fixed CNN, which is already a strong classification representation, and use MMD to decide which layer to use activations from to minimize the domain distribution distance. We can then use this representation to train another classifier for the classes we are interested in recognizing. This can be viewed as coordinate descent on Eqn. 2.2: we take a network that was trained to minimize $\mathcal{L}_C$, select the representation that minimizes MMD, then use that representation to again minimize $\mathcal{L}_C$.

However, this approach is limited in that it cannot directly adapt the representation—instead, it is constrained to selecting from a set of fixed representations. Thus, we propose creating a network to directly optimize the classification and domain alignment objectives, outlined in Figure 2.1.

We begin with the Krizhevsky architecture [KSH12], which has five convolutional and pooling layers and three fully connected layers with dimensions $\{4096, 4096, |C|\}$. We additionally add a lower dimensional, "bottleneck," adaptation layer. Our intuition is that a lower dimensional layer can be used to regularize the training of the source classifier and prevent overfitting to the particular nuances of the source distribution. We place the domain distance loss on top of the "bottleneck" layer to directly regularize the representation to be invariant to the source and target domains.

There are two model selection choices that must be made to add our adaptation layer and the domain distance loss. We must choose where in the network to place the adaptation layer and we must choose the dimension of the layer. We use the MMD metric to make both of these decisions. First, as previously discussed, for our initial fixed representation we find the layer which minimizes the empirical MMD distance between all available source and target data, in our experiments this corresponded to placing the layer after the fully connected layer, $fc7$.

Next, we must determine the dimension for our adaptation layer. We solve this problem with a grid search, where we fine-tune multiple networks using various dimensions and compute the MMD in the new lower dimension representation, finally choosing the dimension which minimizes the source and target distance.

Both the selection of which layer's representation to use ("depth") and how large the adaptation layer should be ("width") are guided by MMD, and thus can be seen as descent steps on our overall objective.

Our architecture (see Figure 2.1) consists of a source and target CNN, with shared weights. Only the labeled examples are used to compute the classification loss, while all data is used from both domains to compute the domain alignment loss. The network is jointly trained on all available source and target data.

The objective outlined in Eqn. 2.2 is easily represented by this convolutional neural network where MMD is computed over minibatches of source and target data. We simply use a fork at the top of the network, after the adaptation layer. One branch uses the labeled data and trains a classifier, and the other branch uses all the data and computes MMD between source and target.

After fine-tuning this architecture, owing to the two terms in the joint loss, the adaptation layer learns a representation that can effectively discriminate between the classes in question due to the classification loss term, while still remaining invariant to domain shift due the MMD term. We expect that such a representation will thus enable increased adaptation performance.

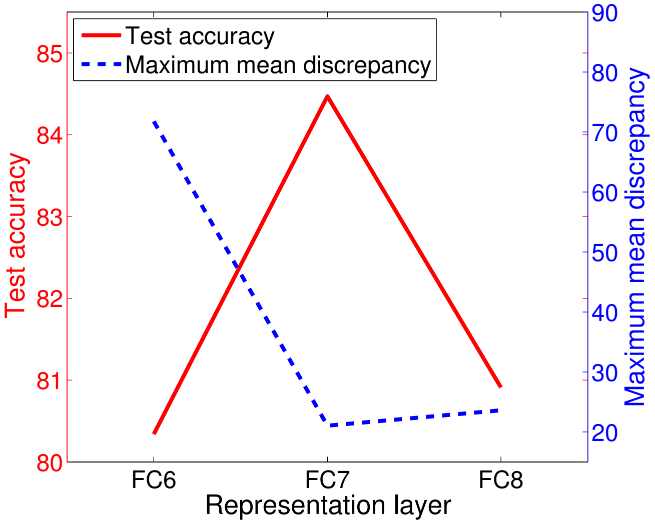


Figure 2.3: Maximum mean discrepancy and test accuracy for different choices of representation layer. We observe that MMD between source and target and accuracy on the target domain test set seem inversely related, indicating that MMD can be used to help select a layer for adaptation.

2.3 Evaluation

We evaluate our adaptation algorithm on a standard domain adaptation dataset with small-scale source domains. We show that our algorithm is effectively able to adapt a deep CNN representation to a target domain with limited or no target labeled data.

The Office [Sae+10] dataset is a collection of images from three distinct domains: Amazon, DSLR, and Webcam. The 31 categories in the dataset consist of objects commonly encountered in office settings, such as keyboards, file cabinets, and laptops. The largest domain has 2817 labeled images.

We evaluate our method across 5 random train/test splits for each of the 3 transfer tasks commonly used for evaluation (Amazon→Webcam, DSLR→Webcam, and Webcam→DSLR) and report averages and standard errors for each setting. We compare in both supervised and unsupervised scenarios against the numbers reported by six recently published methods.

We follow the standard training protocol for this dataset of using 20 source examples per category for the Amazon source domain and 8 images per category for Webcam or DSLR as the source domains [Sae+10; Gon+12]. For the supervised adaptation setting we assume 3 labeled target examples per category.

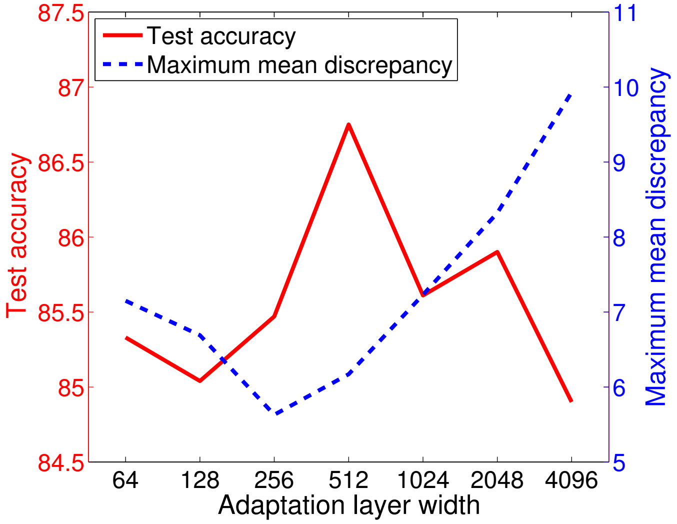


Figure 2.4: Maximum mean discrepancy and test accuracy for different values of adaptation layer dimensionality. We observe that MMD between source and target and accuracy on the target domain test set seem inversely related, indicating that MMD can be used to help select a dimensionality to use.

2.3.1 Evaluating adaptation layer placement

We begin with an evaluation of our representation selection strategy. Using a pre-trained convolutional neural network, we extract features from source and target data using the representations at each fully connected layer. We can then compute the MMD between source and target at each layer. Since a lower MMD indicates that the representation is more domain invariant, we expect the representation with the lowest MMD to achieve the highest performance after adaptation.

To test this hypothesis, for one of the Amazon→Webcam splits we apply a simple domain adaptation baseline introduced by Daumé III [Dau07] to compute test accuracy for the target domain. Figure 2.3 shows a comparison of MMD and adaptation performance across different choices of bridge layers. We see that MMD correctly ranks the representations, singling out $fc7$ as the best performing layer and $fc6$ as the worst. Therefore, we add our adaptation layer after $fc7$ for the remaining experiments.

2.3.2 Choosing the adaptation layer dimension

Before we can learn a new representation via our proposed fine-tuning method, we must determine how wide this representation should be. Again, we use MMD as the deciding metric.

In order to determine what dimensionality our learned adaptation layer should have, we

	$A \to W$	$D \to W$	$W \to D$	Average
GFK(PLS,PCA) [Gon+12]	46.4 ± 0.5	61.3 ± 0.4	66.3 ± 0.4	53.0
SA [Fer+13]	45.0	64.8	69.9	59.9
DA-NBNN [TC13]	52.8 ± 3.7	76.6 ± 1.7	76.2 ± 2.5	68.5
DLID [CBG13]	51.9	78.2	89.9	73.3
DeCAF$_6$ S+T [Don+14]	80.7 ± 2.3	94.8 ± 1.2	–	–
DaNN [GKZ14]	53.6 ± 0.2	71.2 ± 0.0	83.5 ± 0.0	69.4
Ours	**84.1 ± 0.6**	**95.4 ± 0.4**	**96.3 ± 0.3**	**91.9**

Table 2.1: Multi-class accuracy evaluation on the standard supervised adaptation setting with the *Office* dataset. We evaluate on all 31 categories using the standard experimental protocol from [Sae+10]. Here, we compare against six state-of-the-art domain adaptation methods.

	$A \to W$	$D \to W$	$W \to D$	Average
GFK(PLS,PCA) [Gon+12]	15.0 ± 0.4	44.6 ± 0.3	49.7 ± 0.5	36.4
SA [Fer+13]	15.3	50.1	56.9	40.8
DA-NBNN [TC13]	23.3 ± 2.7	67.2 ± 1.9	67.4 ± 3.0	52.6
DLID [CBG13]	26.1	68.9	84.9	60.0
DeCAF$_6$ S [Don+14]	52.2 ± 1.7	91.5 ± 1.5	–	–
DaNN [GKZ14]	35.0 ± 0.2	70.5 ± 0.0	74.3 ± 0.0	59.9
Ours	**59.4 ± 0.8**	**92.5 ± 0.3**	**91.7 ± 0.8**	**81.2**

Table 2.2: Multi-class accuracy evaluation on the standard unsupervised adaptation setting with the *Office* dataset. We evaluate on all 31 categories using the standard experimental protocol from [Gon+12]. Here, we compare against six state-of-the-art domain adaptation methods.

train a variety of networks with different widths on the Amazon→Webcam task, as this is the most challenging of the three. In particular, we try different widths varying from 64 to 4096, stepping by a power of two each time. Once the networks are trained, we then compute MMD between source and target for each of the learned representations. Our method then selects the dimensionality that minimizes the MMD between the source and target data.

To verify that MMD makes the right choice, again we compare MMD with performance on a test set. Figure 2.4 shows that we select 256 dimensions for the adaptation layer, and although this setting is not the one that maximizes test performance, it appears to be a reasonable choice. In particular, using MMD avoids choosing either extreme, near which performance suffers. It is worth noting that the plot has quite a few irregularities—perhaps finer sampling would allow for a more accurate choice.

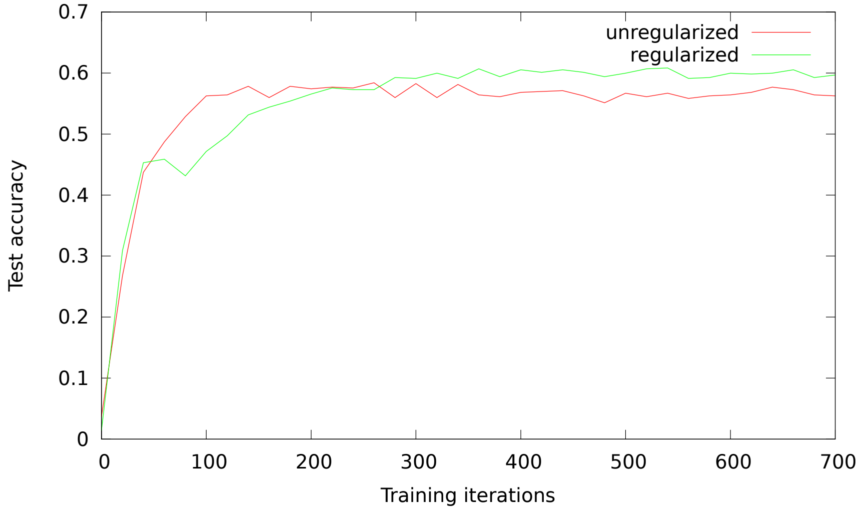


Figure 2.5: A plot of the test accuracy on an unsupervised Amazon→Webcam split during the first 700 iterations of fine-tuning for both regularized and unregularized methods. Although initially the unregularized training achieves better performance, it overfits to the source data. In contrast, using regularization prevents overfitting, so although initial learning is slower we ultimately see better final performance.

2.3.3 Fine-tuning with domain alignment regularization

Once we have settled on our choice of adaptation layer dimensionality, we can begin fine-tuning using the joint loss described in Section 2.2. However, we need to set the regularization hyperparameter λ. Setting λ too low will cause the MMD regularizer have no effect on the learned representation, but setting λ too high will regularize too heavily and learn a degenerate representation in which all points are too close together. We set the regularization hyperparameter to $\lambda = 0.25$, which makes the objective primarily weighted towards classification, but with enough regularization to avoid overfitting.

We use the same fine-tuning architecture for both unsupervised and supervised. However, in the supervised setting, the classifier is trained on data from both domains, whereas in the unsupervised setting, due to the lack of labeled training data, the classifier sees only source data. In both settings, the MMD regularizer sees all of the data, since it does not require labels.

Finally, because the adaptation layer and classifier are being trained from scratch, we set their learning rates to be 10 times higher than the lower layers of the network that were copied from the pre-trained model. Fine-tuning then proceeds via standard backpropagation optimization.

The supervised adaptation setting results are shown in Table 2.1 and the unsupervised adaptation results are shown in Table 2.2. We notice that our algorithm dramatically outperforms all of the competing methods. The distinct improvement of our method demonstrates that the adaptation layer learned via MMD regularized fine-tuning is able to succesfully transfer to a new target domain.

In order to determine how MMD regularization affects learning, we also compare the learning curves with and without regularization on the Amazon→Webcam transfer task in Figure 2.5. We see that, although the unregularized version is initially faster to train, it quickly begins overfitting, and test accuracy suffers. In contrast, using MMD regularization prevents the network from overfitting to the source data, and although training takes longer, the regularization results in a higher final test accuracy.

To further demonstrate the domain invariance of our learned representation, we plot in Figure 2.6 a t-SNE embedding of Amazon and Webcam images using our learned representation and compare it to an embedding created with $fc7$ in the pretrained model. Examining the embeddings, we see that our learned representation exhibits tighter class clustering while mixing the domains within each cluster. While there is weak clustering in the $fc7$ embedding, we find that most tight clusters consist of data points from one domain or the other, but rarely both.

2.4 Conclusion

In this chapter, we presented an objective function for learning domain invariant representations for classification. This objective makes use of an additional domain alignment term to ensure that domains are indistinguishable in the learned representation. We then presented a variety of ways to optimize this objective, ranging from simple representation selection from a fixed pool to a full convolutional architecture that directly minimizes the objective via backpropagation.

Our full method, which uses MMD both to select the depth and width of the architecture while using it as a regularizer during fine-tuning, achieves strong performance on the standard visual domain adaptation benchmark, beating previous methods by a considerable margin.

These experiments show that incorporating a domain alignment term into the discriminative representation learning process is an effective way to ensure that the learned representation is both useful for classification and invariant to domain shifts.

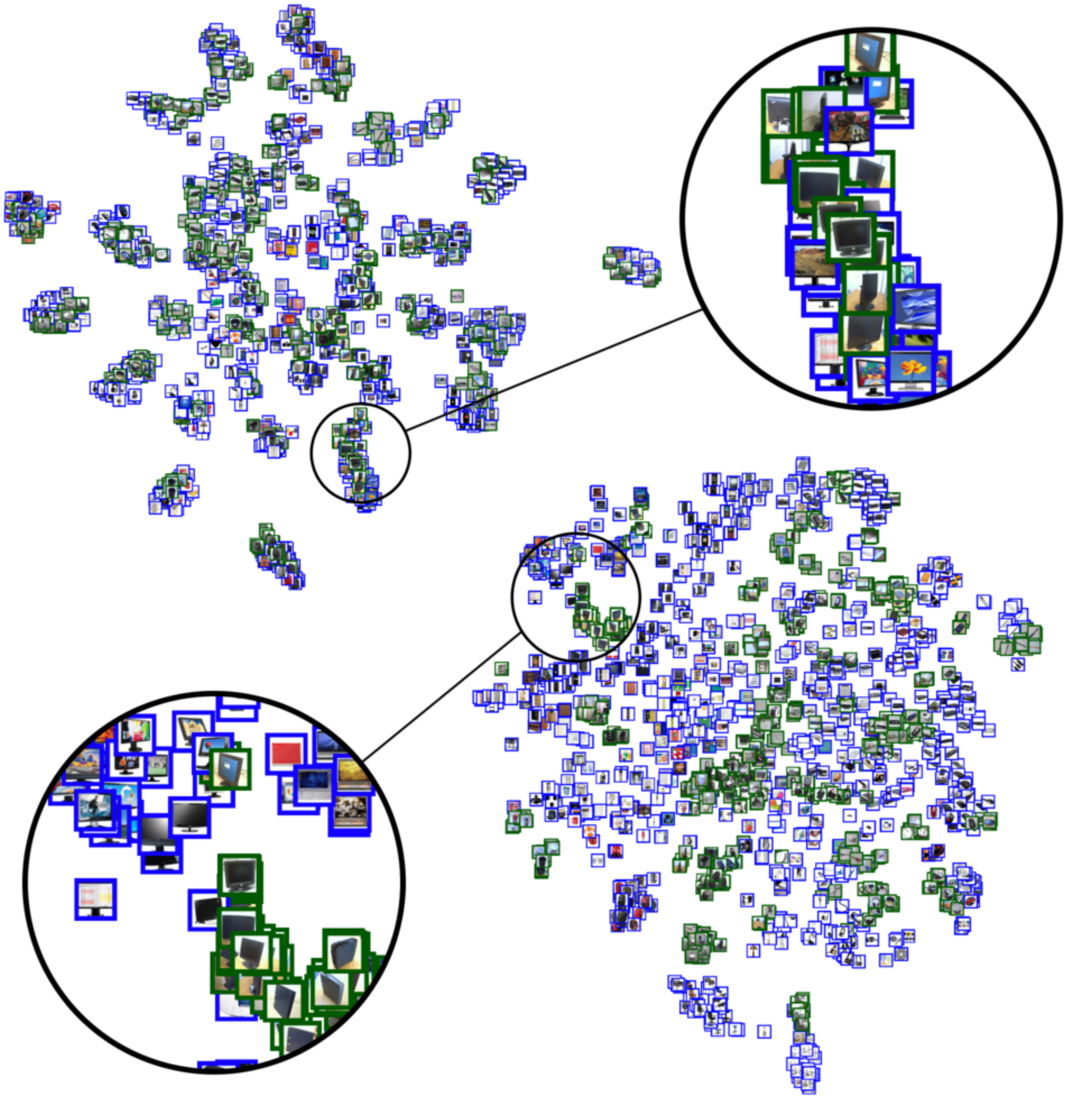


Figure 2.6: t-SNE embeddings of Amazon (blue) and Webcam (green) images using our supervised 256-dimensional representation learned with MMD regularization (top left) and the original $fc7$ representation from the pre-trained model (bottom right). Observe that the clusters formed by our representation separate classes while mixing domains much more effectively than the original representation that was not trained for domain invariance. For example, in $fc7$-space the Amazon monitors and Webcam monitors are separated into distinct clusters, whereas with our learned representation all monitors irrespective of domain are mixed into the same cluster.

Chapter 3

Optimizing Against Learned Measures of Domain Discrepancy

In this chapter, we introduce a domain adaptation approach that simultaneously learns a measure of the domain discrepancy while optimizing against it, in contrast to previously discussed methods which relied on fixed measures such as maximum mean discrepancy. Our approach performs transfer learning both across domains and across tasks (see Figure 3.1). Intuitively, domain transfer is accomplished by making the marginal feature distributions of source and target as similar to each other as possible. Task transfer is enabled by transferring empirical category correlations learned on the source to the target domain. This helps to preserve relationships between categories, e.g., *bottle* is similar to *mug* but different from *keyboard*. Previous work proposed techniques for domain transfer with CNN models [GL15; Lon+15] but did not utilize the learned source semantic structure for task transfer.

To enable domain transfer, we use the unlabeled target data to compute an estimated marginal distribution over the new environment and explicitly optimize a feature representation that minimizes the distance between the source and target domain distributions. We learn deep representations by optimizing over a loss which includes both classification error on the labeled data as well as a *domain confusion* loss which seeks to make the domains indistinguishable.

However, while maximizing domain confusion pulls the marginal distributions of the domains together, it does not necessarily align the classes in the target with those in the source. Thus, we also explicitly transfer the similarity structure amongst categories from the source to the target and further optimize our representation to produce the same structure in the target domain using the few target labeled examples as reference points. We are inspired by prior work on distilling deep models [BC14; HVD14] and extend the ideas presented in these works to a domain adaptation setting. We first compute the average output probability distribution, or “soft label,” over the source training examples in each category. Then, for

This chapter is based on joint work with Judy Hoffman, Trevor Darrell, and Kate Saenko [Tze+15].

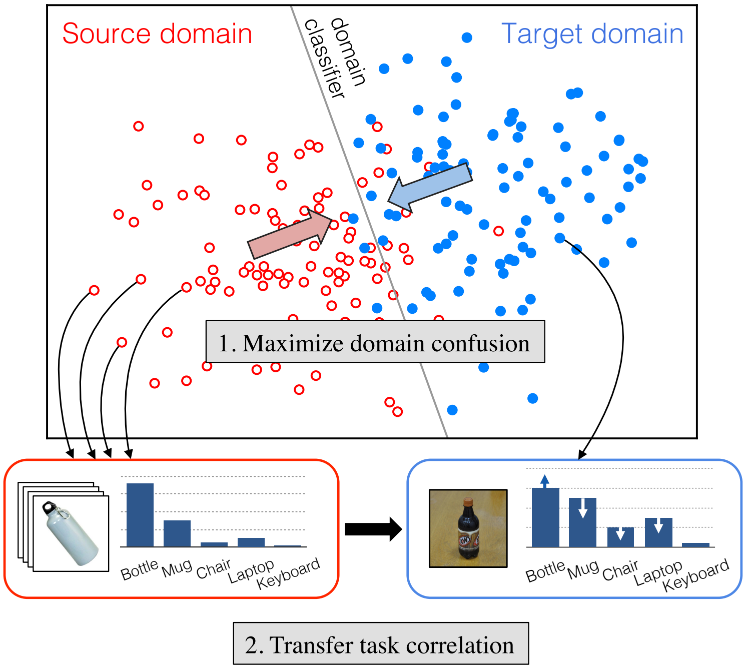


Figure 3.1: We transfer discriminative category information from a source domain to a target domain via two methods. First, we maximize domain confusion by making the marginal distributions of the two domains as similar as possible. Second, we transfer correlations between classes learned on the source examples directly to the target examples, thereby preserving the relationships between classes.

each target labeled example, we directly optimize our model to match the distribution over classes to the soft label. In this way we are able to perform task adaptation by transferring information to categories with no explicit labels in the target domain.

We solve the two problems jointly using a new CNN architecture, outlined in Figure 3.2. We combine domain confusion and softmax cross-entropy losses to train the network with the target data. Our architecture can be used to solve *supervised adaptation*, when a small amount of target labeled data is available from each category, and *semi-supervised adaptation*, when a small amount of target labeled data is available from a subset of the categories. We provide a comprehensive evaluation on the popular Office benchmark [Sae+10] and the recently introduced cross-dataset collection [TTC14] for classification across visually distinct domains. We demonstrate that by jointly optimizing for domain confusion and matching soft labels, we are able to outperform the current state-of-the-art visual domain adaptation results.

3.1 Related work

There have been many approaches proposed in recent years to solve the visual domain adaptation problem, which is also commonly framed as the visual dataset bias problem [TE11]. All recognize that there is a shift in the distribution of the source and target data representations. In fact, the size of a domain shift is often measured by the distance between the source and target subspace representations [Bor+06; Fer+13; KBDG04; MMR09; Pan+09]. A large number of methods have sought to overcome this difference by learning a feature space transformation to align the source and target representations [Sae+10; KSD11; Fer+13; Gon+12]. For the *supervised* adaptation scenario, when a limited amount of labeled data is available in the target domain, some approaches have been proposed to learn a target classifier regularized against the source classifier [YYH07; AZ11; AB10]. Others have sought to both learn a feature transformation and regularize a target classifier simultaneously [Hof+13; DXT12].

Recently, supervised CNN based feature representations have been shown to be extremely effective for a variety of visual recognition tasks [KSH12; Don+14; Gir+13; Ser+13]. In particular, using deep representations dramatically reduces the effect of resolution and lighting on domain shifts [Don+14; Hof+14a]. Parallel CNN architectures such as Siamese networks have been shown to be effective for learning invariant representations [Bro+93; CHL05]. However, training these networks requires labels for each training instance, so it is unclear how to extend these methods to unsupervised or semi-supervised settings. Multimodal deep learning architectures have also been explored to learn representations that are invariant to different input modalities [Ngi+11]. However, this method operated primarily in a generative context and therefore did not leverage the full representational power of supervised CNN representations.

Training a joint source and target CNN architecture was proposed by [CBG13], but was limited to two layers and so was significantly outperformed by the methods which used a deeper architecture [KSH12], pre-trained on a large auxiliary data source (ex: ImageNet [BDFF12]). [GKZ14] proposed pre-training with a denoising auto-encoder, then training a two-layer network simultaneously with the MMD domain confusion loss. This effectively learns a domain invariant representation, but again, because the learned network is relatively shallow, it lacks the strong semantic representation that is learned by directly optimizing a classification objective with a supervised deep CNN.

Using classifier output distributions instead of category labels during training has been explored in the context of model compression or distillation [BC14; HVD14]. However, we are the first to apply this technique in a domain adaptation setting in order to transfer class correlations between domains.

Other works have cotemporaneously explored the idea of directly optimizing a representation for domain invariance [GL15; Lon+15]. However, they either use weaker measures of domain invariance or make use of optimization methods that are less robust than our proposed method, and they do not attempt to solve the task transfer problem in the semi-supervised setting.

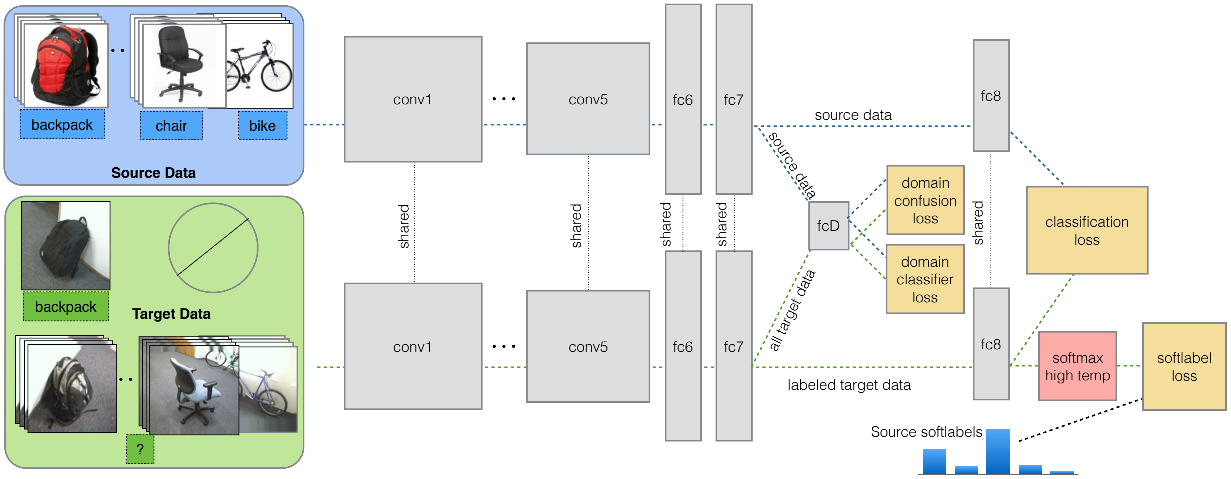


Figure 3.2: Our overall CNN architecture for domain and task transfer. We use a domain confusion loss over all source and target (both labeled and unlabeled) data to learn a domain invariant representation. We simultaneously transfer the learned source semantic structure to the target domain by optimizing the network to produce activation distributions that match those learned for source data in the source only CNN. *Best viewed in color.*

3.2 Joint CNN architecture for domain and task transfer

We first give an overview of our convolutional network (CNN) architecture, depicted in Figure 3.2, that learns a representation which both aligns visual domains and transfers the semantic structure from a well labeled source domain to the sparsely labeled target domain. We assume access to a limited amount of labeled target data, potentially from only a subset of the categories of interest. With limited labels on a subset of the categories, the traditional domain transfer approach of fine-tuning on the available target data [Gir+13; Ser+13; Hof+14b] is not effective. Instead, since the source labeled data shares the label space of our target domain, we use the source data to guide training of the corresponding classifiers.

Our method takes as input the labeled source data $\{x_S, y_S\}$ (blue box Figure 3.2) and the target data $\{x_T, y_T\}$ (green box Figure 3.2), where the labels y_T are only provided for a subset of the target examples. Our goal is to produce a category classifier θ_C that operates on an image feature representation $f(x; \theta_{\text{repr}})$ parameterized by representation parameters θ_{repr} and can correctly classify target examples at test time.

For a setting with K categories, let our desired classification objective be defined as the standard softmax loss

$$\mathcal{L}_C(x, y; \theta_{\text{repr}}, \theta_C) = -\sum_k \mathbb{1}[y = k] \log p_k \tag{3.1}$$

where p is the softmax of the classifier activations, $p = \text{softmax}(\theta_C^T f(x; \theta_{\text{repr}}))$.

We could use the available source labeled data to train our representation and classifier parameters according to Equation equation 3.1, but this often leads to overfitting to the source distribution, causing reduced performance at test time when recognizing in the target domain. However, we note that if the source and target domains are very similar then the classifier trained on the source will perform well on the target. In fact, it is sufficient for the source and target data to be similar under the learned representation, θ_{repr}.

Inspired by the "name the dataset" game of Torralba and Efros [TE11], we can directly train a domain classifier θ_D to identify whether a training example originates from the source or target domain given its feature representation. Intuitively, if our choice of representation suffers from domain shift, then they will lie in distinct parts of the feature space, and a classifier will be able to easily separate the domains. We use this notion to add a new *domain confusion* loss $\mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}})$ to our objective and directly optimize our representation so as to minimize the discrepancy between the source and target distributions. This loss is described in more detail in Section 3.2.1.

Domain confusion can be applied to learn a representation that aligns source and target data without any target labeled data. However, we also presume a handful of sparse labels in the target domain, y_T. In this setting, a simple approach is to incorporate the target labeled data along with the source labeled data into the classification objective of Equation equation 3.1[1]. However, fine-tuning with hard category labels limits the impact of a single training example, making it hard for the network to learn to generalize from the limited labeled data. Additionally, fine-tuning with hard labels is ineffective when labeled data is available for only a subset of the categories.

For our approach, we draw inspiration from recent network distillation works [BC14; HVD14], which demonstrate that a large network can be "distilled" into a simpler model by replacing the hard labels with the softmax activations from the original large model. This modification proves to be critical, as the distribution holds key information about the relationships between categories and imposes additional structure during the training process. In essence, because each training example is paired with an output distribution, it provides valuable information about not only the category it belongs to, but also each other category the classifier is trained to recognize.

Thus, we propose using the labeled target data to optimize the network parameters through a *soft label* loss, $\mathcal{L}_{\text{soft}}(x_T, y_T; \theta_{\text{repr}}, \theta_C)$. This loss will train the network parameters to produce a "soft label" activation that matches the average output distribution of source examples on a network trained to classify source data. This loss is described in more detail in Section 3.2.2. By training the network to match the expected source output distributions on target data, we transfer the learned inter-class correlations from the source domain to examples in the target domain. This directly transfers useful information from source to target, such as the fact that *bookshelves* appear more similar to *filing cabinets* than to *bicycles*.

[1] We present this approach as one of our baselines.

Our full method then minimizes the joint loss function

$$\begin{aligned}\mathcal{L}(x_S, y_S, x_T, y_T, \theta_D;&\theta_{\text{repr}}, \theta_C) = \\ &\mathcal{L}_C(x_S, y_S, x_T, y_T; \theta_{\text{repr}}, \theta_C) \\ &+ \lambda \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}}) \\ &+ \nu \mathcal{L}_{\text{soft}}(x_T, y_T; \theta_{\text{repr}}, \theta_C).\end{aligned} \tag{3.2}$$

where the hyperparameters λ and ν determine how strongly domain confusion and soft labels influence the optimization.

Our ideas of domain confusion and soft label loss for task transfer are generic and can be applied to any CNN classification architecture. For our experiments and for the detailed discussion in this chapter we modify the standard Krizhevsky architecture [KSH12], which has five convolutional layers (conv1–conv5) and three fully connected layers (fc6–fc8). The representation parameter θ_{repr} corresponds to layers 1–7 of the network, and the classification parameter θ_C corresponds to layer 8. For the remainder of this section, we provide further details on our novel loss definitions and the implementation of our model.

3.2.1 Aligning domains via domain confusion

In this section we describe in detail our proposed *domain confusion* loss objective. Recall that we introduce the domain confusion loss as a means to learn a representation that is domain invariant, and thus will allow us to better utilize a classifier trained using the labeled source data. We consider a representation to be domain invariant if a classifier trained using that representation can not distinguish examples from the two domains.

To this end, we add an additional domain classification layer, denoted as fcD in Figure 3.2, with parameters θ_D. This layer simply performs binary classification using the domain corresponding to an image as its label. For a particular feature representation, θ_{repr}, we evaluate its domain invariance by learning the best domain classifier on the representation. This can be learned by optimizing the following objective, where y_D denotes the domain that the example is drawn from:

$$\mathcal{L}_D(x_S, x_T, \theta_{\text{repr}}; \theta_D) = -\sum_d \mathbb{1}[y_D = d] \log q_d \tag{3.3}$$

with q corresponding to the softmax of the domain classifier activation: $q = \text{softmax}(\theta_D^T f(x; \theta_{\text{repr}}))$.

For a particular domain classifier, θ_D, we can now introduce our loss which seeks to “maximally confuse” the two domains by computing the cross entropy between the output predicted domain labels and a uniform distribution over domain labels:

$$\mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}}) = -\sum_d \frac{1}{D} \log q_d. \tag{3.4}$$

This domain confusion loss seeks to learn domain invariance by finding a representation in which the best domain classifier performs poorly.

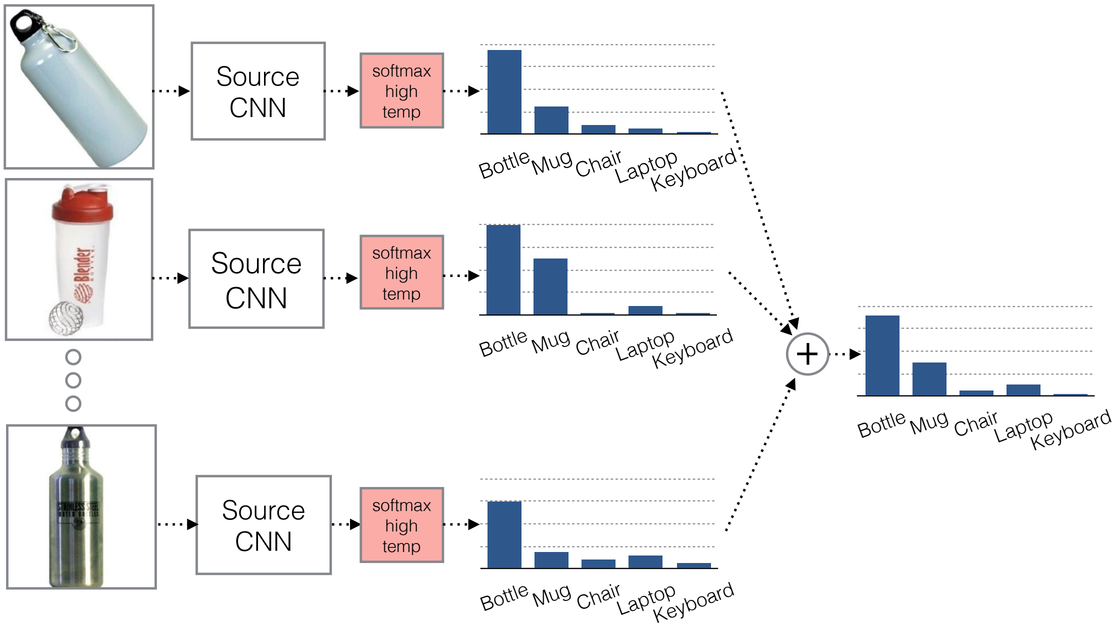


Figure 3.3: Soft label distributions are learned by averaging the per-category activations of source training examples using the source model. An example, with 5 categories, depicted here to demonstrate the final soft activation for the bottle category will be primarily dominated by bottle and mug with very little mass on chair, laptop, and keyboard.

Ideally, we want to simultaneously minimize Equations equation 3.3 and equation 3.4 for the representation and the domain classifier parameters. However, the two losses stand in direct opposition to one another: learning a fully domain invariant representation means the domain classifier must do poorly, and learning an effective domain classifier means that the representation is not domain invariant. Rather than globally optimizing θ_D and θ_{repr}, we instead perform iterative updates for the following two objectives given the fixed parameters from the previous iteration:

$$\min_{\theta_D} \mathcal{L}_D(x_S, x_T, \theta_{\text{repr}}; \theta_D) \tag{3.5}$$

$$\min_{\theta_{\text{repr}}} \mathcal{L}_{\text{conf}}(x_S, x_T, \theta_D; \theta_{\text{repr}}). \tag{3.6}$$

These losses are readily implemented in standard deep learning frameworks, and after setting learning rates properly so that Equation equation 3.5 only updates θ_D and Equation equation 3.6 only updates θ_{repr}, the updates can be performed via standard backpropagation. Together, these updates ensure that we learn a representation that is domain invariant.

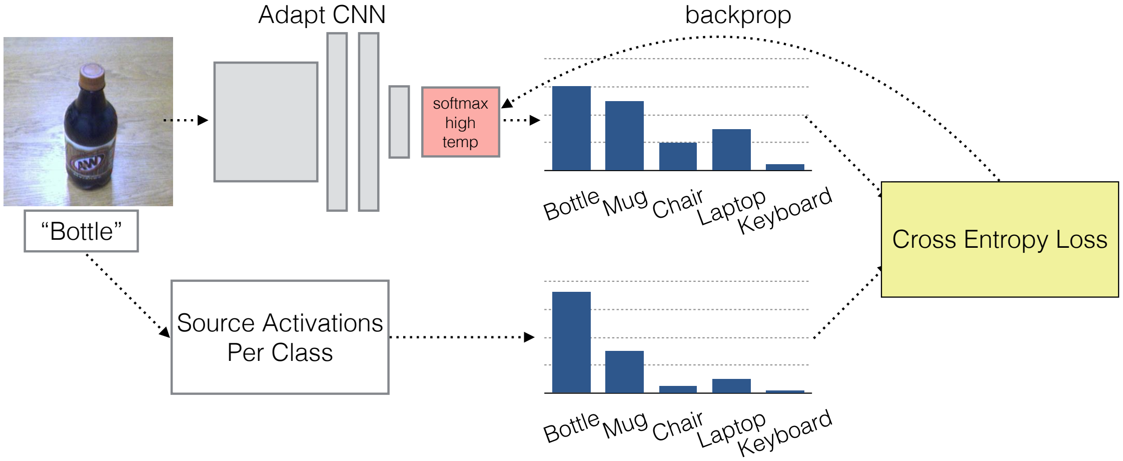


Figure 3.4: Depiction of the use of source per-category soft activations with the cross entropy loss function over the current target activations.

3.2.2 Aligning source and target classes via soft labels

While training the network to confuse the domains acts to align their marginal distributions, there are no guarantees about the alignment of classes between each domain. To ensure that the relationships between classes are preserved across source and target, we fine-tune the network against "soft labels" rather than the image category hard label.

We define a soft label for category k as the average over the softmax of all activations of source examples in category k, depicted graphically in Figure 3.3, and denote this average as $l^{(k)}$. Note that, since the source network was trained purely to optimize a classification objective, a simple softmax over each z_S^i will hide much of the useful information by producing a very peaked distribution. Instead, we use a softmax with a high temperature τ so that the related classes have enough probability mass to have an effect during fine-tuning. With our computed per-category soft labels we can now define our soft label loss:

$$\mathcal{L}_{\text{soft}}(x_T, y_T; \theta_{\text{repr}}, \theta_C) = -\sum_i l_i^{(y_T)} \log p_i \tag{3.7}$$

where p denotes the soft activation of the target image, $p = \text{softmax}(\theta_C^T f(x_T; \theta_{\text{repr}})/\tau)$. The loss above corresponds to the cross-entropy loss between the soft activation of a particular target image and the soft label corresponding to the category of that image, as shown in Figure 3.4.

To see why this will help, consider the soft label for a particular category, such as *bottle*. The soft label $l^{(\text{bottle})}$ is a K-dimensional vector, where each dimension indicates the similarity of bottles to each of the K categories. In this example, the bottle soft label will have a higher weight on *mug* than on *keyboard*, since bottles and mugs are more visually similar. Thus, soft label training with this particular soft label directly enforces the relationship that bottles and mugs should be closer in feature space than bottles and keyboards.

	$A \to W$	$A \to D$	$W \to A$	$W \to D$	$D \to A$	$D \to W$	Average
DLID [CBG13]	51.9	–	–	89.9	–	78.2	–
DeCAF$_6$ S+T [Don+14]	80.7 ± 2.3	–	–	–	–	94.8 ± 1.2	–
DaNN [GKZ14]	53.6 ± 0.2	–	–	83.5 ± 0.0	–	71.2 ± 0.0	–
Source CNN	56.5 ± 0.3	64.6 ± 0.4	42.7 ± 0.1	93.6 ± 0.2	47.6 ± 0.1	92.4 ± 0.3	66.22
Target CNN	80.5 ± 0.5	81.8 ± 1.0	59.9 ± 0.3	81.8 ± 1.0	59.9 ± 0.3	80.5 ± 0.5	74.05
Source+Target CNN	82.5 ± 0.9	85.2 ± 1.1	**65.2 ± 0.7**	96.3 ± 0.5	65.8 ± 0.5	93.9 ± 0.5	81.50
Ours: dom confusion only	**82.8 ± 0.9**	85.9 ± 1.1	64.9 ± 0.5	97.5 ± 0.2	**66.2 ± 0.4**	95.6 ± 0.4	82.13
Ours: soft labels only	82.7 ± 0.7	84.9 ± 1.2	**65.2 ± 0.6**	**98.3 ± 0.3**	66.0 ± 0.5	**95.9 ± 0.6**	82.17
Ours: dom confusion+soft labels	82.7 ± 0.8	**86.1 ± 1.2**	65.0 ± 0.5	97.6 ± 0.2	**66.2 ± 0.3**	95.7 ± 0.5	**82.22**

Table 3.1: Multi-class accuracy evaluation on the standard supervised adaptation setting with the *Office* dataset. We evaluate on all 31 categories using the standard experimental protocol from [Sae+10]. Here, we compare against three state-of-the-art domain adaptation methods as well as a CNN trained using only source data, only target data, or both source and target data together.

One important benefit of using this soft label loss is that we ensure that the parameters for categories without any labeled target data are still updated to output non-zero probabilities. We explore this benefit in Section 3.3, where we train a network using labels from a subset of the target categories and find significant performance improvement even when evaluating only on the unlabeled categories.

3.3 Evaluation

To analyze the effectiveness of our method, we evaluate it on the Office dataset, a standard benchmark dataset for visual domain adaptation, and on a new large-scale cross-dataset domain adaptation challenge.

3.3.1 Adaptation on the Office dataset

The Office dataset is a collection of images from three distinct domains, Amazon, DSLR, and Webcam, the largest of which has 2817 labeled images [Sae+10]. The 31 categories in the dataset consist of objects commonly encountered in office settings, such as keyboards, file cabinets, and laptops.

We evaluate our method in two different settings:

- **Supervised adaptation** Labeled training data for all categories is available in source and sparsely in target.
- **Semi-supervised adaptation (task adaptation)** Labeled training data is available in source and sparsely for a subset of the target categories.

For all experiments we initialize the parameters of conv1–fc7 using the released CaffeNet [Jia+14] weights. We then further fine-tune the network using the source labeled data

	$A \rightarrow W$	$A \rightarrow D$	$W \rightarrow A$	$W \rightarrow D$	$D \rightarrow A$	$D \rightarrow W$	Average
MMDT [Hof+13]	–	44.6 ± 0.3	–	58.3 ± 0.5	–	–	–
Source CNN	54.2 ± 0.6	63.2 ± 0.4	34.7 ± 0.1	94.5 ± 0.2	36.4 ± 0.1	89.3 ± 0.5	62.0
Ours: dom confusion only	55.2 ± 0.6	63.7 ± 0.9	**41.1 ± 0.0**	96.5 ± 0.1	41.2 ± 0.1	**91.3 ± 0.4**	64.8
Ours: soft labels only	56.8 ± 0.4	65.2 ± 0.9	38.8 ± 0.4	96.5 ± 0.2	41.7 ± 0.3	89.6 ± 0.1	64.8
Ours: dom confusion+soft labels	**59.3 ± 0.6**	**68.0 ± 0.5**	40.5 ± 0.2	**97.5 ± 0.1**	**43.1 ± 0.2**	90.0 ± 0.2	**66.4**

Table 3.2: Multi-class accuracy evaluation on the standard semi-supervised adaptation setting with the *Office* dataset. We evaluate on 16 held-out categories for which we have no access to target labeled data. We show results on these unsupervised categories for the source only model, our model trained using only soft labels for the 15 auxiliary categories, and finally using domain confusion together with soft labels on the 15 auxiliary categories.

in order to produce the soft label distributions and use the learned source CNN weights as the initial parameters for training our method. All implementations are produced using the open source Caffe [Jia+14] framework, and the network definition files and cross entropy loss layer needed for training will be released upon acceptance. We optimize the network using a learning rate of 0.001 and set the hyper-parameters to $\lambda = 0.01$ (confusion) and $\nu = 0.1$ (soft).

For each of the six domain shifts, we evaluate across five train/test splits, which are generated by sampling examples from the full set of images per domain. In the source domain, we follow the standard protocol for this dataset and generate splits by sampling 20 examples per category for the Amazon domain, and 8 examples per category for the DSLR and Webcam domains.

We first present results for the supervised setting, where 3 labeled examples are provided for each category in the target domain. We report accuracies on the remaining unlabeled images, following the standard protocol introduced with the dataset [Sae+10]. In addition to a variety of baselines, we report numbers for both soft label fine-tuning alone as well as soft labels with domain confusion in Table 3.1. Because the Office dataset is imbalanced, we report multi-class accuracies, which are obtained by computing per-class accuracies independently, then averaging over all 31 categories.

We see that fine-tuning with soft labels or domain confusion provides a consistent improvement over hard label training in 5 of 6 shifts. Combining soft labels with domain confusion produces marginally higher performance on average. This result follows the intuitive notion that when enough target labeled examples are present, directly optimizing for the joint source and target classification objective (Source+Target CNN) is a strong baseline and so using either of our new losses adds enough regularization to improve performance.

Next, we experiment with the semi-supervised adaptation setting. We consider the case in which training data and labels are available for some, but not all of the categories in the target domain. We are interested in seeing whether we can transfer information learned from the labeled classes to the unlabeled classes.

To do this, we consider having 10 target labeled examples per category from only 15 of the

31 total categories, following the standard protocol introduced with the *Office* dataset [Sae+10]. We then evaluate our classification performance on the remaining 16 categories for which no data was available at training time.

In Table 3.2 we present multi-class accuracies over the 16 held-out categories and compare our method to a previous domain adaptation method [Hof+13] as well as a source-only trained CNN. Note that, since the performance here is computed over only a subset of the categories in the dataset, the numbers in this table should not be directly compared to the supervised setting in Table 3.1.

We find that all variations of our method (only soft label loss, only domain confusion, and both together) outperform the baselines. Contrary to the fully supervised case, here we note that both domain confusion and soft labels contribute significantly to the overall performance improvement of our method. This stems from the fact that we are now evaluating on categories which lack labeled target data, and thus the network can not implicitly enforce domain invariance through the classification objective alone. Separately, the fact that we get improvement from the soft label training on related tasks indicates that information is being effectively transferred between tasks.

In Figure 3.5, we show examples for the Amazon→Webcam shift where our method correctly classifies images from held out object categories and the baseline does not. We find that our method is able to consistently overcome error cases, such as the notebooks that were previously confused with letter trays, or the black mugs that were confused with black computer mice.

3.3.2 Adaptation between diverse domains

For an evaluation with larger, more distinct domains, we test on the recent testbed for cross-dataset analysis [TTC14], which collects images from classes shared in common among computer vision datasets. We use the dense version of this testbed, which consists of 40 categories shared between the ImageNet, Caltech-256, SUN, and Bing datasets, and evaluate specifically with ImageNet as source and Caltech-256 as target.

We follow the protocol outlined in [TTC14] and generate 5 splits by selecting 5534 images from ImageNet and 4366 images from Caltech-256 across the 40 shared categories. Each split is then equally divided into a train and test set. However, since we are most interested in evaluating in the setting with limited target data, we further subsample the target training set into smaller sets with only 1, 3, and 5 labeled examples per category.

Results from this evaluation are shown in Figure 3.6. We compare our method to both CNNs fine-tuned using only source data using source and target labeled data. Contrary to the previous supervised adaptation experiment, our method significantly outperforms both baselines. We see that our full architecture, combining domain confusion with the soft label loss, performs the best overall and is able to operate in the regime of no labeled examples in the target (corresponding to the red line at point 0 on the x-axis). We find that the most benefit of our method arises when there are few labeled training examples per category in the target domain. As we increase the number of labeled examples in the target, the standard

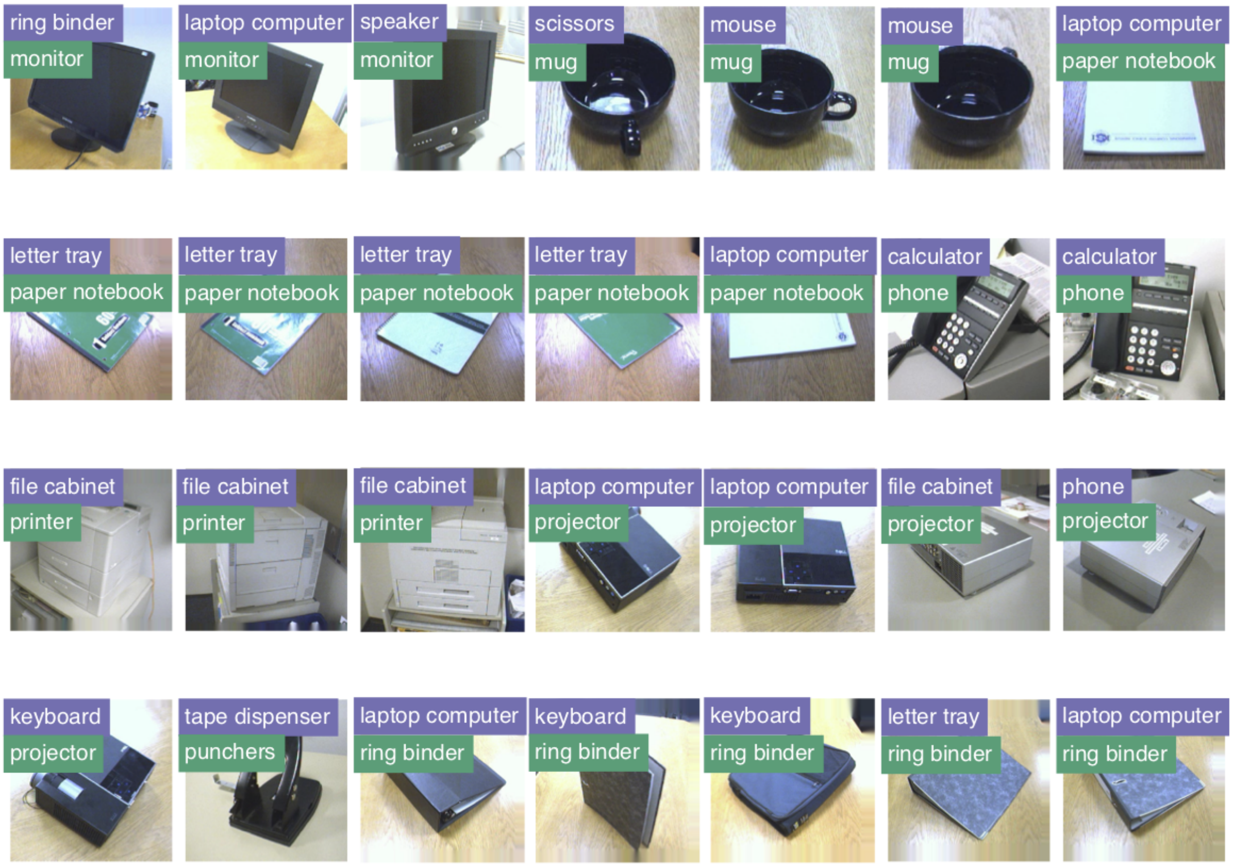


Figure 3.5: Examples from the Amazon→Webcam shift in the semi-supervised adaptation setting, where our method (the bottom turquoise label) correctly classifies images while the baseline (the top purple label) does not.

fine-tuning strategy begins to approach the performance of the adaptation approach. This indicates that direct joint source and target fine-tuning is a viable adaptation approach when you have a reasonable number of training examples per category. In comparison, fine-tuning on the target examples alone yields accuracies of 36.6 ± 0.6, 60.9 ± 0.5, and 67.7 ± 0.5 for the cases of 1, 3, and 5 labeled examples per category, respectively. All of these numbers underperform the source only model, indicating that adaptation is crucial in the setting of limited training data.

Finally, we note that our results are significantly higher than the 24.8% result reported in [TTC14], despite the use of much less training data. This difference is explained by their use of SURF BoW features, indicating that CNN features are a much stronger feature for use in adaptation tasks.

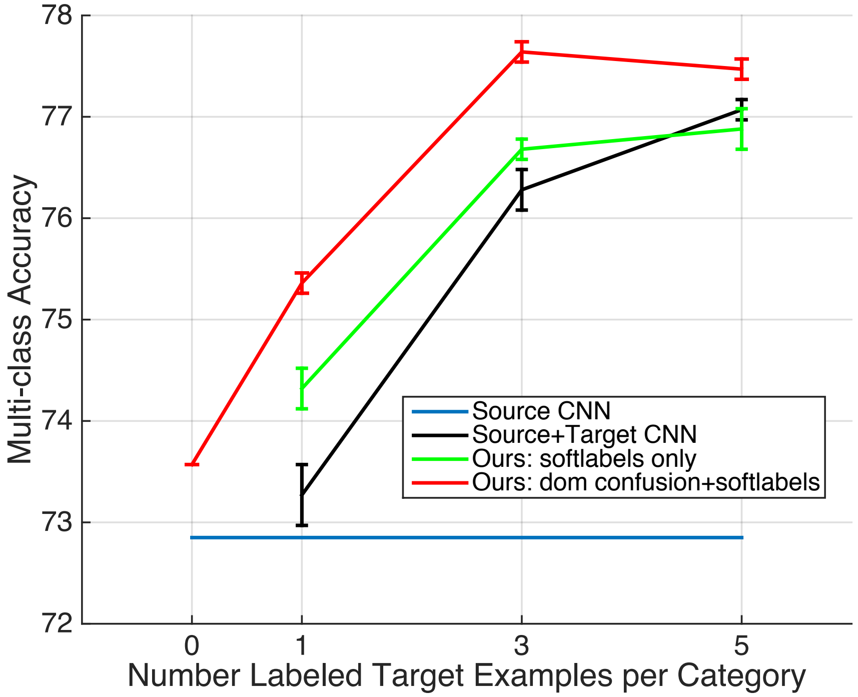


Figure 3.6: ImageNet→Caltech supervised adaptation from the Cross-dataset [TTC14] testbed with varying numbers of labeled target examples per category. We find that our method using soft label loss (with and without domain confusion) outperforms the baselines of training on source data alone or using a standard fine-tuning strategy to train with the source and target data. *Best viewed in color.*

3.4 Analysis

Our experimental results demonstrate that our method improves classification performance in a variety of domain adaptation settings. We now perform additional analysis on our method by confirming our claims that it exhibits domain invariance and transfers information across tasks.

3.4.1 Domain confusion enforces domain invariance

We begin by evaluating the effectiveness of domain confusion at learning a domain invariant representation. As previously explained, we consider a representation to be domain invariant if an optimal classifier has difficulty predicting which domain an image originates from. Thus, for our representation learned with a domain confusion loss, we expect a trained domain classifier to perform poorly.

We train two support vector machines (SVMs) to classify images into domains: one using the baseline CaffeNet fc7 representation, and the other using our fc7 learned with domain

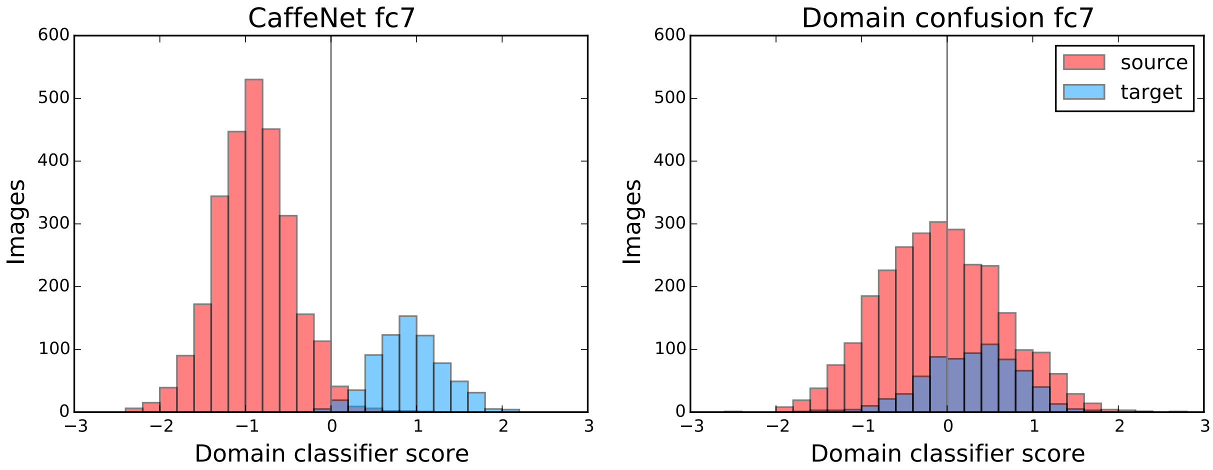


Figure 3.7: We compare the baseline CaffeNet representation to our representation learned with domain confusion by training a support vector machine to predict the domains of Amazon and Webcam images. For each representation, we plot a histogram of the classifier decision scores of the test images. In the baseline representation, the classifier is able to separate the two domains with 99% accuracy. In contrast, the representation learned with domain confusion is domain invariant, and the classifier can do no better than 56%.

confusion. These SVMs are trained using 160 images, 80 from Amazon and 80 from Webcam, then tested on the remaining images from those domains. We plot the classifier scores for each test image in Figure 3.7. It is obvious that the domain confusion representation is domain invariant, making it much harder to separate the two domains—the test accuracy on the domain confusion representation is only 56%, not much better than random. In contrast, on the baseline CaffeNet representation, the domain classifier achieves 99% test accuracy.

3.4.2 Soft labels for task transfer

We now examine the effect of soft labels in transferring information between categories. We consider the Amazon→Webcam shift from the semi-supervised adaptation experiment in the previous section. Recall that in this setting, we have access to target labeled data for only half of our categories. We use soft label information from the source domain to provide information about the held-out categories which lack labeled target examples. Figure 3.8 examines one target example from the held-out category *monitor*. No labeled target monitors were available during training; however, as shown in the upper right corner of Figure 3.8, the soft labels for *laptop computer* was present during training and assigns a relatively high weight to the *monitor* class. Soft label fine-tuning thus allows us to exploit the fact that these categories are similar. We see that the baseline model misclassifies this image as a *ring binder*, while our soft label model correctly assigns the *monitor* label.

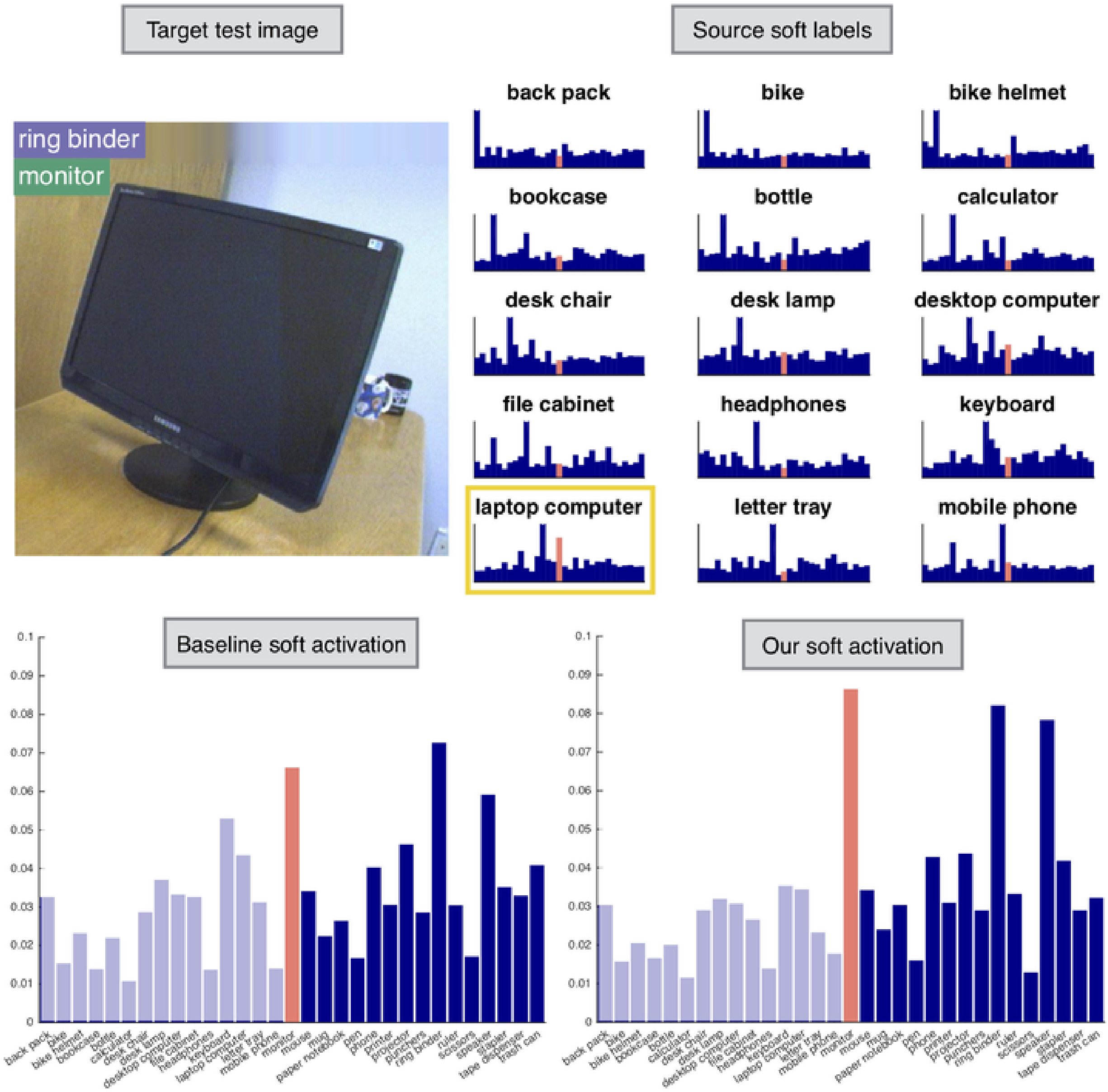


Figure 3.8: Our method (bottom turquoise label) correctly predicts the category of this image, whereas the baseline (top purple label) does not. The source per-category soft labels for the 15 categories with labeled target data are shown in the upper right corner, where the x-axis of the plot represents the 31 categories and the y-axis is the output probability. We highlight the index corresponding to the *monitor* category in red. As no labeled target data is available for the correct category, *monitor*, we find that in our method the related category of *laptop computer* (outlined with yellow box) transfers information to the monitor category. As a result, after training, our method places the highest weight on the correct category. Probability score per category for the baseline and our method are shown in the bottom left and right, respectively, training categories are opaque and correct test category is shown in red.

3.5 Conclusion

In this chapter, we have presented a CNN architecture that effectively adapts to a new domain with limited or no labeled data per target category. We accomplish this through a novel CNN architecture which simultaneously optimizes for domain invariance, to facilitate domain transfer, while transferring task information between domains in the form of a cross entropy soft label loss. We demonstrate the ability of our architecture to improve adaptation performance in the *supervised* and *semi-supervised* settings by experimenting with two standard domain adaptation benchmark datasets. In the semi-supervised adaptation setting, we see an average relative improvement of 13% over the baselines on the four most challenging shifts in the Office dataset. Overall, our method can be easily implemented as an alternative fine-tuning strategy when limited or no labeled data is available per category in the target domain.

Chapter 4

Adversarial Discriminative Domain Adaptation

Many domain adaptation methods attempt to mitigate the harmful effects of domain shift by learning deep neural transformations that map both domains into a common feature space. Adversarial adaptation methods have become an increasingly popular incarnation of this type of approach which seeks to minimize an approximate domain discrepancy distance through an adversarial objective with respect to a domain discriminator. These methods are closely related to generative adversarial learning [Goo+14], which pits two networks against each other—a generator and a discriminator. The generator is trained to produce images in a way that confuses the discriminator, which in turn tries to distinguish them from real image examples. In domain adaptation, this principle has been employed to ensure that the network cannot distinguish between the distributions of its training and test domain examples [GL15; Tze+15; LT16]. However, each algorithm makes different design choices such as whether to use a generator, which loss function to employ, or whether to share weights across domains. For example, [GL15; Tze+15] share weights and learn a symmetric mapping of both source and target images to the shared feature space, while [LT16] decouple some layers thus learning a partially asymmetric mapping.

In this chapter, we propose a novel unified framework for adversarial domain adaptation, allowing us to effectively examine the different factors of variation between the existing approaches and clearly view the similarities they each share. Our framework unifies design choices such as weight-sharing, base models, and adversarial losses and subsumes previous work, while also facilitating the design of novel instantiations that improve upon existing ones.

In particular, we observe that generative modeling of input image distributions is not necessary, as the ultimate task is to learn a discriminative representation. On the other hand, asymmetric mappings can better model the difference in low level features than symmetric

This chapter is based on joint work with Judy Hoffman, Kate Saenko, and Trevor Darrell [Tze+17].

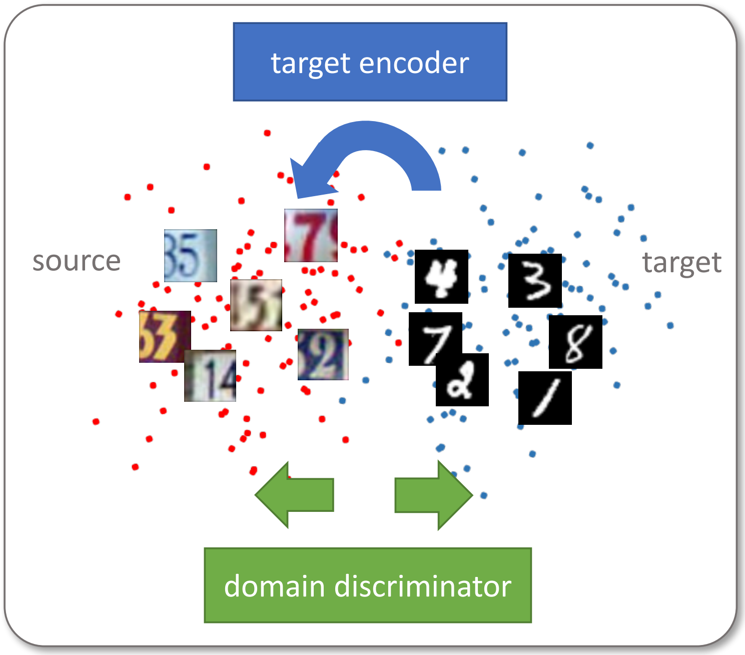


Figure 4.1: We propose an improved unsupervised domain adaptation method that combines adversarial learning with discriminative feature learning. Specifically, we learn a discriminative mapping of target images to the source feature space (target encoder) by fooling a domain discriminator that tries to distinguish the encoded target images from source examples.

ones. We therefore propose a previously unexplored unsupervised adversarial adaptation method, Adversarial Discriminative Domain Adaptation (ADDA), illustrated in Figure 4.1. ADDA first learns a discriminative representation using the labels in the source domain and then a separate encoding that maps the target data to the same space using an asymmetric mapping learned through a domain-adversarial loss. Our approach is simple yet surprisingly powerful and achieves strong visual adaptation results on the MNIST, USPS, and SVHN digits datasets. We also test its potential to bridge the gap between even more difficult cross-modality shifts, without requiring instance constraints, by transferring object classifiers from RGB color images to depth observations. Finally, we evaluate on the standard Office adaptation dataset, and show that ADDA achieves strong improvements over competing methods, especially on the most challenging domain shift.

4.1 Related work

There has been extensive prior work on domain transfer learning, see e.g., [Gre+09]. Recent work has focused on transferring deep neural network representations from a labeled source datasets to a target domain where labeled data is sparse or non-existent. In the case of unlabeled target domains (the focus of this chapter) the main strategy has been to guide feature learning by minimizing the difference between the source and target feature distributions [GL15; Tze+15; Tze+14; Lon+15; SS16; Ghi+16; LT16; Sen+16].

Several methods have used the Maximum Mean Discrepancy (MMD) [Gre+09] loss for this purpose. MMD computes the norm of the difference between two domain means. The DDC method [Tze+14] used MMD in addition to the regular classification loss on the source to learn a representation that is both discriminative and domain invariant. The Deep Adaptation Network (DAN) [Lon+15] applied MMD to layers embedded in a reproducing kernel Hilbert space, effectively matching higher order statistics of the two distributions. In contrast, the deep Correlation Alignment (CORAL) [SS16] method proposed to match the mean and covariance of the two distributions.

Other methods have chosen an adversarial loss to minimize domain shift, learning a representation that is simultaneously discriminative of source labels while not being able to distinguish between domains. [Tze+15] proposed adding a domain classifier (a single fully connected layer) that predicts the binary domain label of the inputs and designed a *domain confusion* loss to encourage its prediction to be as close as possible to a uniform distribution over binary labels. The gradient reversal algorithm (ReverseGrad) proposed in [GL15] also treats domain invariance as a binary classification problem, but directly maximizes the loss of the domain classifier by reversing its gradients. DRCN [Ghi+16] takes a similar approach but also learns to reconstruct target domain images. Domain separation networks [Bou+16] enforce these adversarial losses to minimize domain shift in a shared feature space, but achieve impressive results by augmenting their model with private feature spaces per-domain, an additional dissimilarity loss between the shared and private spaces, and a reconstruction loss.

In related work, adversarial learning has been explored for generative tasks. The Generative Adversarial Network (GAN) method [Goo+14] is a generative deep model that pits two networks against one another: a generative model G that captures the data distribution and a discriminative model D that distinguishes between samples drawn from G and images drawn from the training data by predicting a binary label. The networks are trained jointly using backprop on the label prediction loss in a mini-max fashion: simultaneously update G to minimize the loss while also updating D to maximize the loss (fooling the discriminator). The advantage of GAN over other generative methods is that there is no need for complex sampling or inference during training; the downside is that it may be difficult to train. GANs have been applied to generate natural images of objects, such as digits and faces, and have been extended in several ways. The BiGAN approach [DKD17] extends GANs to also learn the inverse mapping from the image data back into the latent space, and shows that this can learn features useful for image classification tasks. The conditional generative adversarial net (CGAN) [MO14] is an extension of the GAN where both networks G and D receive an additional vector of information as input. This might contain, say, information about the class of the training example. The authors apply CGAN to generate a (possibly multi-modal) distribution of tag-vectors conditional on image features. GANs have also been explicitly applied to domain transfer tasks, such as domain transfer networks [TPW17], which seek to directly map source images into target images.

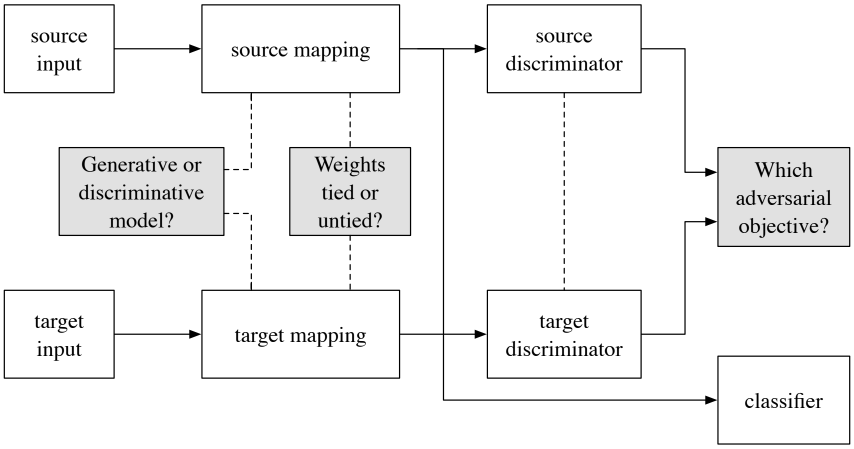


Figure 4.2: Our generalized architecture for adversarial domain adaptation. Existing adversarial adaptation methods can be viewed as instantiations of our framework with different choices regarding their properties.

Recently the CoGAN [LT16] approach applied GANs to the domain transfer problem by training two GANs to generate the source and target images respectively. The approach achieves a domain invariant feature space by tying the high-level layer parameters of the two GANs, and shows that the same noise input can generate a corresponding pair of images from the two distributions. Domain adaptation was performed by training a classifier on the discriminator output and applied to shifts between the MNIST and USPS digit datasets. However, this approach relies on the generators finding a mapping from the shared high-level layer feature space to full images in both domains. This can work well for say digits which can be difficult in the case of more distinct domains. In this chapter, we observe that modeling the image distributions is not strictly necessary to achieve domain adaptation, as long as the latent feature space is domain invariant, and propose a discriminative approach.

4.2 Generalized adversarial adaptation

We present a general framework for adversarial unsupervised adaptation methods. In unsupervised adaptation, we assume access to source images $\mathbf{X}_s$ and labels Y_s drawn from a source domain distribution $p_s(x, y)$, as well as target images $\mathbf{X}_t$ drawn from a target distribution $p_t(x, y)$, where there are no label observations. Our goal is to learn a target representation, M_t and classifier C_t that can correctly classify target images into one of K categories at test time, despite the lack of in domain annotations. Since direct supervised learning on the target is not possible, domain adaptation instead learns a source representation mapping, M_s, along with a source classifier, C_s, and then learns to adapt that model for use in the target domain.

In adversarial adaptive methods, the main goal is to regularize the learning of the source and target mappings, M_s and M_t, so as to minimize the distance between the empirical source

Method	**Base model**	**Weight sharing**	**Adversarial loss**
Gradient reversal [Gan+16]	discriminative	shared	minimax
Domain confusion [Tze+15]	discriminative	shared	confusion
CoGAN [LT16]	generative	unshared	GAN
ADDA (Ours)	discriminative	unshared	GAN

Table 4.1: Overview of adversarial domain adaption methods and their various properties. Viewing methods under a unified framework enables us to easily propose a new adaptation method, adversarial discriminative domain adaptation (ADDA).

and target mapping distributions: $M_s(X_s)$ and $M_t(X_t)$. If this is the case then the source classification model, C_s, can be directly applied to the target representations, elimating the need to learn a separate target classifier and instead setting, $C = C_s = C_t$.

The source classification model is then trained using the standard supervised loss

$$\min_{M_s,C} \mathcal{L}_{\text{cls}}(\mathbf{X}_s, Y_t) = \mathbb{E}_{(\mathbf{x}_s,y_s)\sim(\mathbf{X}_s,Y_t)} \; -\sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log C(M_s(\mathbf{x}_s)). \tag{4.1}$$

We are now able to describe our full general framework view of adversarial adaptation approaches. We note that all approaches minimize source and target representation distances through alternating minimization between two functions. First a domain discriminator, D, which classifies whether a data point is drawn from the source or the target domain. Thus, D is optimized according to a standard supervised loss, $\mathcal{L}_{\text{adv}_D}(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t)$ where the labels indicate the origin domain, defined as

$$\mathcal{L}_{\text{adv}_D}(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t) = -\mathbb{E}_{\mathbf{x}_s\sim\mathbf{X}_s}[\log D(M_s(\mathbf{x}_s))] - \mathbb{E}_{\mathbf{x}_t\sim\mathbf{X}_t}[\log(1 - D(M_t(\mathbf{x}_t)))]. \tag{4.2}$$

Second, the source and target mappings are optimized according to a constrained adversarial objective, whose particular instantiation may vary across methods. Thus, we can derive a generic formulation for domain adversarial techniques below:

$$\begin{aligned} &\min_{D} \;\; \mathcal{L}_{\text{adv}_D}(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t) \\ &\min_{M_s,M_t} \mathcal{L}_{\text{adv}_M}(\mathbf{X}_s, \mathbf{X}_t, D) \\ &\;\text{s.t.} \quad \psi(M_s, M_t) \end{aligned} \tag{4.3}$$

In the next sections, we demonstrate the value of our framework by positioning recent domain adversarial approaches within our framework. We describe the potential mapping structure, mapping optimization constraints ($\psi(M_s, M_t)$) choices and finally choices of adversarial mapping loss, $\mathcal{L}_{\text{adv}_M}$.

4.2.1 Source and target mappings

In the case of learning a source mapping M_s alone it is clear that supervised training through a latent space discriminative loss using the known labels Y_s results in the best representation for final source recognition. However, given that our target domain is unlabeled, it remains an open question how best to minimize the distance between the source and target mappings. Thus the first choice to be made is in the particular parameterization of these mappings.

Because unsupervised domain adaptation generally considers target discriminative tasks such as classification, previous adaptation methods have generally relied on adapting discriminative models between domains [Tze+15; Gan+16]. With a discriminative base model, input images are mapped into a feature space that is useful for a discriminative task such as image classification. For example, in the case of digit classification this may be the standard LeNet model. However, Liu and Tuzel achieve state of the art results on unsupervised MNIST-USPS using two generative adversarial networks [LT16]. These generative models use random noise as input to generate samples in image space—generally, an intermediate feature of an adversarial discriminator is then used as a feature for training a task-specific classifier.

Once the mapping parameterization is determined for the source, we must decide how to parametrize the target mapping M_t. In general, the target mapping almost always matches the source in terms of the specific functional layer (architecture), but different methods have proposed various regularization techniques. All methods initialize the target mapping parameters with the source, but different methods choose different constraints between the source and target mappings, $\psi(M_s, M_t)$. The goal is to make sure that the target mapping is set so as to minimize the distance between the source and target domains under their respective mappings, while crucially also maintaining a target mapping that is category discriminative.

Consider a layered representations where each layer parameters are denoted as, M_s^ℓ or M_t^ℓ, for a given set of equivalent layers, $\{\ell_1, \dots, \ell_n\}$. Then the space of constraints explored in the literature can be described through layerwise equality constraints as follows:

$$\psi(M_s, M_t) \triangleq \{\psi_{\ell_i}(M_s^{\ell_i}, M_t^{\ell_i})\}_{i \in \{1 \dots n\}} \tag{4.4}$$

where each individual layer can be constrained independently. A very common form of constraint is source and target layerwise equality:

$$\psi_{\ell_i}(M_s^{\ell_i}, M_t^{\ell_i}) = (M_s^{\ell_i} = M_t^{\ell_i}). \tag{4.5}$$

It is also common to leave layers unconstrained. These equality constraints can easily be imposed within a convolutional network framework through weight sharing.

For many prior adversarial adaptation methods [Gan+16; Tze+15], all layers are constrained, thus enforcing exact source and target mapping consistency. Learning a symmetric transformation reduces the number of parameters in the model and ensures that the mapping used for the target is discriminative at least when applied to the source domain. However, this may make the optimization poorly conditioned, since the same network must handle images from two separate domains.

An alternative approach is instead to learn an asymmetric transformation with only a subset of the layers constrained, thus enforcing partial alignment. Rozantsev et al. [RSF18] showed that partially shared weights can lead to effective adaptation in both supervised and unsupervised settings. As a result, some recent methods have favored untying weights (fully or partially) between the two domains, allowing models to learn parameters for each domain individually.

4.2.2 Adversarial losses

Once we have decided on a parametrization of M_t, we employ an adversarial loss to learn the actual mapping. There are various different possible choices of adversarial loss functions, each of which have their own unique use cases. All adversarial losses train the adversarial discriminator using a standard classification loss, $\mathcal{L}_{\text{adv}_D}$, previously stated in Equation 4.2. However, they differ in the loss used to train the mapping, $\mathcal{L}_{\text{adv}_M}$.

The gradient reversal layer of [Gan+16] optimizes the mapping to maximize the discriminator loss directly:

$$\mathcal{L}_{\text{adv}_M} = -\mathcal{L}_{\text{adv}_D}. \tag{4.6}$$

This optimization corresponds to the true minimax objective for generative adversarial networks. However, this objective can be problematic, since early on during training the discriminator converges quickly, causing the gradient to vanish.

When training GANs, rather than directly using the minimax loss, it is typical to train the generator with the standard loss function with inverted labels [Goo+14]. This splits the optimization into two independent objectives, one for the generator and one for the discriminator, where $\mathcal{L}_{\text{adv}_D}$ remains unchanged, but $\mathcal{L}_{\text{adv}_M}$ becomes:

$$\mathcal{L}_{\text{adv}_M}(\mathbf{X}_s, \mathbf{X}_t, D) = -\mathbb{E}_{\mathbf{x}_t \sim \mathbf{X}_t}[\log D(M_t(\mathbf{x}_t))]. \tag{4.7}$$

This objective has the same fixed-point properties as the minimax loss but provides stronger gradients to the target mapping. We refer to this modified loss function as the “GAN loss function” for the remainder of this chapter.

Note that, in this setting, we use independent mappings for source and target and learn only M_t adversarially. This mimics the GAN setting, where the real image distribution remains fixed, and the generating distribution is learned to match it.

The GAN loss function is the standard choice in the setting where the generator is attempting to mimic another unchanging distribution. However, in the setting where both distributions are changing, this objective will lead to oscillation—when the mapping converges to its optimum, the discriminator can simply flip the sign of its prediction in response. Tzeng et al. instead proposed the domain confusion objective, under which the mapping is trained

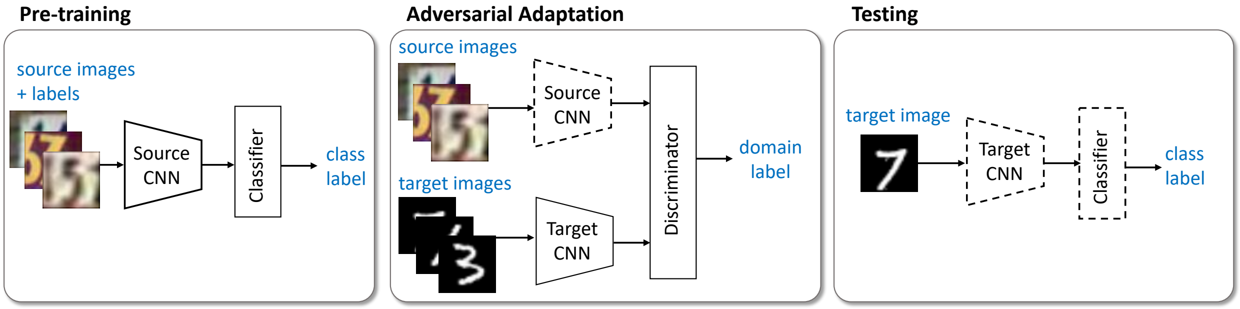


Figure 4.3: An overview of our proposed Adversarial Discriminative Domain Adaptation (ADDA) approach. We first pre-train a source encoder CNN using labeled source image examples. Next, we perform adversarial adaptation by learning a target encoder CNN such that a discriminator that sees encoded source and target examples cannot reliably predict their domain label. During testing, target images are mapped with the target encoder to the shared feature space and classified by the source classifier. Dashed lines indicate fixed network parameters.

using a cross-entropy loss function against a uniform distribution [Tze+15]:

$$\mathcal{L}_{\text{adv}_M}(\mathbf{X}_s, \mathbf{X}_t, D) = -\sum_{d\in\{s,t\}} \mathbb{E}_{\mathbf{x}_d\sim\mathbf{X}_d}\left[\frac{1}{2}\log D(M_d(\mathbf{x}_d)) + \frac{1}{2}\log(1 - D(M_d(\mathbf{x}_d)))\right]. \tag{4.8}$$

This loss ensures that the adversarial discriminator views the two domains identically.

4.3 Adversarial discriminative domain adaptation

The benefit of our generalized framework for domain adversarial methods is that it directly enables the development of novel adaptive methods. In fact, designing a new method has now been simplified to the space of making three design choices: whether to use a generative or discriminative base model, whether to tie or untie the weights, and which adversarial learning objective to use. In light of this view we can summarize our method, adversarial discriminative domain adaptation (ADDA), as well as its connection to prior work, according to our choices (see Table 4.1 “ADDA”). Specifically, we use a discriminative base model, unshared weights, and the standard GAN loss. We illustrate our overall sequential training procedure in Figure 4.3.

First, we choose a discriminative base model, as we hypothesize that much of the parameters required to generate convincing in-domain samples are irrelevant for discriminative adaptation tasks. Most prior adversarial adaptive methods optimize directly in a discriminative space for this reason. One counter-example is CoGANs. However, this method has only shown dominance in settings where the source and target domain are very similar such as

MNIST and USPS, and in our experiments we have had difficulty getting it to converge for larger distribution shifts.

Next, we choose to allow independent source and target mappings by untying the weights. This is a more flexible learing paradigm as it allows more domain specific feature extraction to be learned. However, note that the target domain has no label access, and thus without weight sharing a target model may quickly learn a degenerate solution if we do not take care with proper initialization and training procedures. Therefore, we use the pre-trained source model as an intitialization for the target representation space and fix the source model during adversarial training.

In doing so, we are effectively learning an asymmetric mapping, in which we modify the target model so as to match the source distribution. This is most similar to the original generative adversarial learning setting, where a generated space is updated until it is indistinguishable with a fixed real space. Therefore, we choose the inverted label GAN loss described in the previous section.

Our proposed method, ADDA, thus corresponds to the following unconstrained optimization:

$$\begin{aligned}
&\min_{M_s, C} \; \mathcal{L}_{\text{cls}}(\mathbf{X}_s, Y_s) = -\mathbb{E}_{(\mathbf{x}_s, y_s)\sim(\mathbf{X}_s, Y_s)} \sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log C(M_s(\mathbf{x}_s)) \\
&\min_{D} \; \mathcal{L}_{\text{adv}_D}(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t) = -\mathbb{E}_{\mathbf{x}_s\sim\mathbf{X}_s}[\log D(M_s(\mathbf{x}_s))] - \mathbb{E}_{\mathbf{x}_t\sim\mathbf{X}_t}[\log(1 - D(M_t(\mathbf{x}_t)))] \\
&\min_{M_t} \; \mathcal{L}_{\text{adv}_M}(\mathbf{X}_s, \mathbf{X}_t, D) = -\mathbb{E}_{\mathbf{x}_t\sim\mathbf{X}_t}[\log D(M_t(\mathbf{x}_t))].
\end{aligned} \tag{4.9}$$

We choose to optimize this objective in stages. We begin by optimizing $\mathcal{L}_{\text{cls}}$ over M_s and C by training using the labeled source data, $\mathbf{X}_s$ and Y_s. Because we have opted to leave M_s fixed while learning M_t, we can thus optimize $\mathcal{L}_{\text{adv}_D}$ and $\mathcal{L}_{\text{adv}_M}$ without revisiting the first objective term. A summary of this entire training process is provided in Figure 4.3.

We note that the unified framework presented in the previous section has enabled us to compare prior domain adversarial methods and make informed decisions about the different factors of variation. Through this framework we are able to motivate a novel domain adaptation method, ADDA, and offer insight into our design decisions. In the next section we demonstrate promising results on unsupervised adaptation benchmark tasks, studying adaptation across visual domains and across modalities.

4.4 Experiments

We now evaluate ADDA for unsupervised classification adaptation across three different adaptation settings. We explore three digits datasets of varying difficulty: MNIST [LeC+98], USPS, and SVHN [Net+11]. We additionally evaluate on the NYUD [Sil+12] dataset to study adaptation across modalities. Finally, we evaluate on the standard Office [Sae+10] dataset

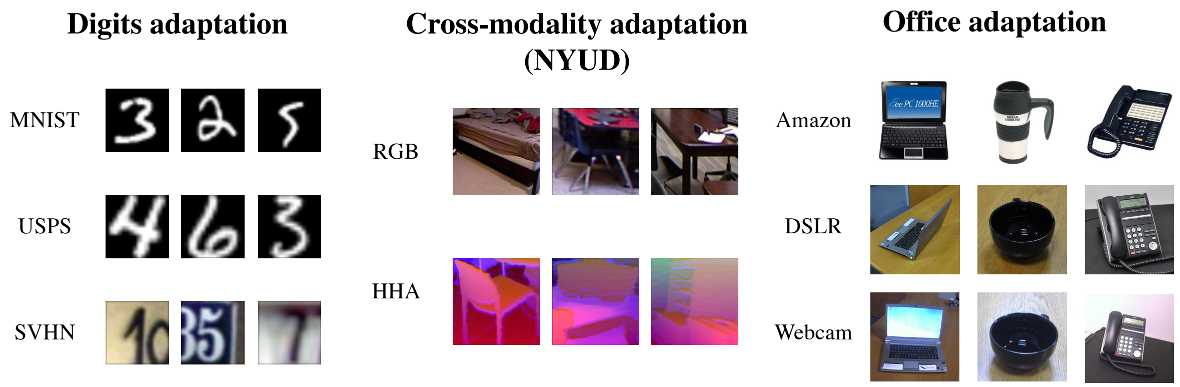


Figure 4.4: We evaluate ADDA on unsupervised adaptation across seven domain shifts in three different settings. The first setting is adaptation between the MNIST, USPS, and SVHN datasets (left). The second setting is a challenging cross-modality adaptation task between RGB and depth modalities from the NYU depth dataset (center). The third setting is adaptation on the standard Office adaptation dataset between the Amazon, DSLR, and Webcam domains (right).

for comparison against previous work. Example images from all experimental datasets are provided in Figure 4.4.

For the case of digit adaptation, we compare against multiple state-of-the-art unsupervised adaptation methods, all based upon domain adversarial learning objectives. In 3 of 4 of our experimental setups, our method outperforms all competing approaches, and in the last domain shift studied, our approach outperforms all but one competing approach. We also validate our model on a real-world modality adaptation task using the NYU depth dataset. Despite a large domain shift between the RGB and depth modalities, ADDA learns a useful depth representation without any labeled depth data and improves over the nonadaptive baseline by over 50% (relative). Finally, on the standard Office dataset, we demonstrate ADDA's effectiveness by showing convincing improvements over competing approaches, especially on the hardest domain shift.

4.4.1 MNIST, USPS, and SVHN digits datasets

We experimentally validate our proposed method in an unsupervised adaptation task between the MNIST [LeC+98], USPS, and SVHN [Net+11] digits datasets, which consist 10 classes of digits. Example images from each dataset are visualized in Figure 4.4 and Table 4.2. For adaptation between MNIST and USPS, we follow the training protocol established in [Lon+13], sampling 2000 images from MNIST and 1800 from USPS. For adaptation between SVHN and MNIST, we use the full training sets for comparison against [Gan+16]. All experiments are performed in the unsupervised settings, where labels in the target domain are withheld, and we consider adaptation in three directions: MNIST→USPS, USPS→MNIST,

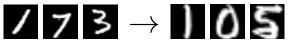
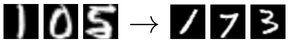
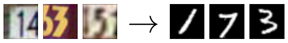
Method	MNIST → USPS 	USPS → MNIST 	SVHN → MNIST 
Source only	0.752 ± 0.016	0.571 ± 0.017	0.601 ± 0.011
Gradient reversal	0.771 ± 0.018	0.730 ± 0.020	0.739 [Gan+16]
Domain confusion	0.791 ± 0.005	0.665 ± 0.033	0.681 ± 0.003
CoGAN	0.912 ± 0.008	0.891 ± 0.008	did not converge
ADDA (Ours)	0.894 ± 0.002	0.901 ± 0.008	0.760 ± 0.018

Table 4.2: Experimental results on unsupervised adaptation among MNIST, USPS, and SVHN.

and SVHN→MNIST.

For these experiments, we use the simple modified LeNet architecture provided in the Caffe source code [LeC+98; Jia+14]. When training with ADDA, our adversarial discriminator consists of 3 fully connected layers: two layers with 500 hidden units followed by the final discriminator output. Each of the 500-unit layers uses a ReLU activation function. Optimization proceeds using the Adam optimizer [KB15] for 10,000 iterations with a learning rate of 0.0002, a β_1 of 0.5, a β_2 of 0.999, and a batch size of 256 images (128 per domain). All training images are converted to greyscale, and rescaled to 28×28 pixels.

Results of our experiment are provided in Table 4.2. On the easier MNIST and USPS shifts ADDA achieves comparable performance to the current state-of-the-art, CoGANs [LT16], despite being a considerably simpler model. This provides compelling evidence that the machinery required to generate images is largely irrelevant to enabling effective adaptation. Additionally, we show convincing results on the challenging SVHN and MNIST task in comparison to other methods, indicating that our method has the potential to generalize to a variety of settings. In contrast, we were unable to get CoGANs to converge on SVHN and MNIST—because the domains are so disparate, we were unable to train coupled generators for them.

4.4.2 Modality adaptation

We use the NYU depth dataset [Sil+12], which contains bounding box annotations for 19 object classes in 1449 images from indoor scenes. The dataset is split into a train (381 images), val (414 images) and test (654) sets. To perform our cross-modality adaptation, we first crop out tight bounding boxes around instances of these 19 classes present in the dataset and evaluate on a 19-way classification task over object crops. In order to ensure that the same instance is not seen in both domains, we use the RGB images from the train split as the source domain and the depth images from the val split as the target domain. This corresponds to 2,186 labeled source images and 2,401 unlabeled target images. Figure 4.4 visualizes samples from each of the two domains.

	bathtub	bed	bookshelf	box	chair	counter	desk	door	dresser	garbage bin	lamp	monitor	night stand	pillow	sink	sofa	table	television	toilet	**overall**
# of instances	19	96	87	210	611	103	122	129	25	55	144	37	51	276	47	129	210	33	17	2401
Source only	0.000	0.010	0.011	0.124	0.188	0.029	0.041	0.047	0.000	0.000	0.069	0.000	0.039	0.587	0.000	0.008	0.010	0.000	0.000	0.139
ADDA (Ours)	0.000	0.146	0.046	0.229	0.344	0.447	0.025	0.023	0.000	0.018	0.292	0.081	0.020	0.297	0.021	0.116	0.143	0.091	0.000	0.211
Train on target	0.105	0.531	0.494	0.295	0.619	0.573	0.057	0.636	0.120	0.291	0.576	0.189	0.235	0.630	0.362	0.248	0.357	0.303	0.647	0.468

Table 4.3: Adaptation results on the NYUD [Sil+12] dataset, using RGB images from the train set as source and depth images from the val set as target domains. We report here per class accuracy due to the large class imbalance in our target set (indicated in # instances). Overall our method improves average per category accuracy from 13.9% to 21.1%.

We consider the task of adaptation between these RGB and HHA encoded depth images [Gup+14], using them as source and target domains respectively. Because the bounding boxes are tight and relatively low resolution, accurate classification is quite difficult, even when evaluating in-domain. In addition, the dataset has very few examples for certain classes, such as `toilet` and `bathtub`, which directly translates to reduced classification performance.

For this experiment, our base architecture is the VGG-16 architecture, initializing from weights pretrained on ImageNet [SZ15]. This network is then fully fine-tuned on the source domain for 20,000 iterations using a batch size of 128. When training with ADDA, the adversarial discriminator consists of three additional fully connected layers: 1024 hidden units, 2048 hidden units, then the adversarial discriminator output. With the exception of the output, these layers use a ReLU activation function. ADDA training then proceeds for another 20,000 iterations, using the same hyperparameters as in the digits experiments.

We find that our method, ADDA, greatly improves classification accuracy for this task. For certain categories, like `counter`, classification accuracy goes from 2.9% under the source only baseline up to 44.7% after adaptation. In general, average accuracy across all classes improves significantly from 13.9% to 21.1%. However, not all classes improve. Three classes have no correctly labeled target images before adaptation, and adaptation is unable to recover performance on these classes. Additionally, the classes of `pillow` and `nightstand` suffer performance loss after adaptation.

4.4.3 Office dataset

Finally, we evaluate our method on the benchmark Office visual domain adaptation dataset [Sae+10]. This dataset consists of 4,110 images spread across 31 classes in 3 domains: *amazon*, *webcam*, and *dslr*. Following previous work [Gan+16], we focus our evaluation on three domain shifts: *amazon* to *webcam* ($A \to W$), *dslr* to *webcam* ($D \to W$), and *webcam* to *dslr* ($W \to D$). We evaluate ADDA fully transductively, where we train on every labeled example in the source domain and every unlabeled example in the target.

Method	$A \to W$	$D \to W$	$W \to D$
Source only (AlexNet)	0.642	0.961	0.978
Source only (ResNet-50)	0.626	0.961	0.986
DDC [Tze+14]	0.618	0.950	0.985
DAN [Lon+15]	0.685	0.960	0.990
DRCN [Ghi+16]	0.687	0.964	0.990
DANN [Gan+16]	0.730	0.964	0.992
ADDA (Ours)	**0.751**	**0.970**	**0.996**

Table 4.4: Unsupervised adaptation performance on the Office dataset in the fully-transductive setting. ADDA achieves strong results on all three evaluated domain shifts and demonstrates the largest improvement on the hardest shift, $A \to W$.

Because the Office dataset is relatively small, fine-tuning a full network quickly leads to overfitting. As a result, we use ResNet-50 [He+16] as our base model due to its relatively low number of parameters and fine-tune only the lower layers of the target model, up to but not including conv5. Optimization is done using SGD for 20,000 iterations with a learning rate of 0.001, a momentum of 0.9, and a batch size of 64. The adversarial discriminator consists of three fully connected layers of dimensions 1024, 2048, and 3072, each followed by ReLUs, and one fully connected layer for the final output. We present the results of this experiment in Table 4.4.

We see that ADDA is competitive on this adaptation task as well, achieving state-of-the-art on all three of the evaluated domain shifts. Although the base architecture ADDA uses is different from previous work, which typically fine-tunes using AlexNet as a base, by comparing the source-only baselines we see that the ResNet-50 architecture does not perform significantly better. Additionally, we see the largest increase on the hardest shift $A \to W$ despite ResNet-50's poor performance on that shift, indicating that ADDA is effective even on challenging real-world adaptation tasks.

4.5 Conclusion

We have proposed a unified framework for unsupervised domain adaptation techniques based on adversarial learning objectives. Our framework provides a simplified and cohesive view by which we may understand the similarities and differences between recently proposed adaptation methods. Through this comparison, we are able to understand the benefits and key ideas from each approach and to combine these strategies into a new adaptation method, ADDA.

We presented an evaluation across four domain shifts for our unsupervised adaptation approach. Our method generalizes well across a variety of tasks, achieving strong results on benchmark adaptation datasets as well as a challenging cross-modality adaptation task.

Chapter 5

CyCADA: Cycle-Consistent Adversarial Domain Adaptation

Deep neural networks excel at learning from large amounts of data, but can be poor at generalizing learned knowledge to new datasets or environments. Even a slight departure from a network's training domain can cause it to make spurious predictions and significantly hurt its performance [Tze+17]. The visual domain shift from non-photorealistic synthetic data to real images presents an even more significant challenge. While we would like to train models on large amounts of synthetic data such as data collected from graphics game engines, such models fail to generalize to real-world imagery. For example, a state-of-the-art semantic segmentation model trained on synthetic dashcam data fails to segment the road in real images, and its overall per-pixel label accuracy drops from 93% (if trained on real imagery) to 54% (if trained only on synthetic data, see Table 5.7).

Feature-level unsupervised domain adaptation methods address this problem by aligning the features extracted from the network across the source (e.g. synthetic) and target (e.g. real) domains, without any labeled target samples. Alignment typically involves minimizing some measure of distance between the source and target feature distributions, such as maximum mean discrepancy [LW15], correlation distance [SS16], or adversarial discriminator accuracy [GL15; Tze+17]. This class of techniques suffers from two main limitations. First, aligning marginal distributions does not enforce any semantic consistency, e.g. target features of a car may be mapped to source features of a bicycle. Second, alignment at higher levels of a deep representation can fail to model aspects of low-level appearance variance which are crucial for the end visual task.

Generative pixel-level domain adaptation models perform similar distribution alignment—not in feature space but rather in raw pixel space—translating source data to the "style" of a target domain. Recent methods can learn to translate images given only unsupervised

This chapter is based on joint work with Judy Hoffman, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell [Hof+18].

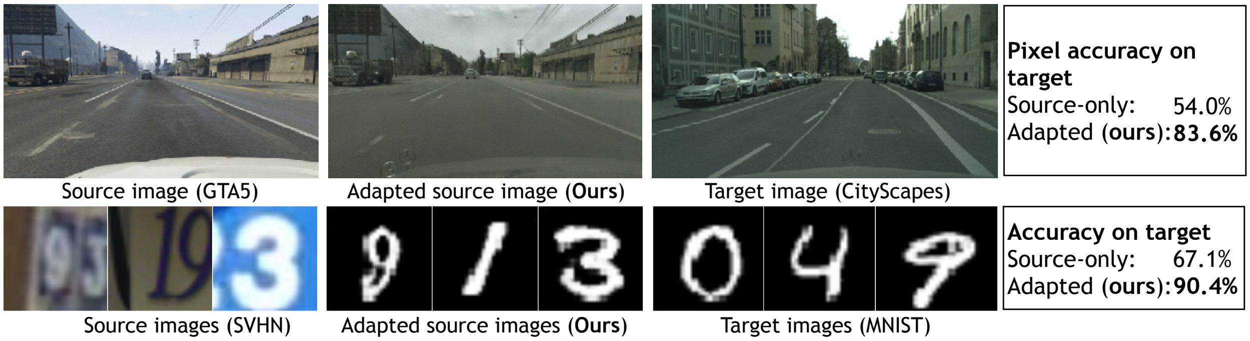


Figure 5.1: We propose CyCADA, an adversarial unsupervised adaptation algorithm which uses cycle and semantic consistency to perform adaptation at multiple levels in a deep network. Our model provides significant performance improvements over source model baselines.

data from both domains [Bou+17; Shr+17; LBK17]. Such image-space models have only been shown to work for small image sizes and limited domain shifts. A more recent approach [Bou+18] was applied to larger images, but in a controlled environment with visually simple images for robotic applications. Image to Image translation techniques, such as CycleGAN [Zhu+17], have produced visually appealing results which preserve local content in natural scenes, but are not designed with an end task in mind and so may not always preserve semantics. For example, a model adapting from digits taken from Google Street View to handwritten digits can learn to make a printed 8 look like a hand-written 1.

We propose Cycle-Consistent Adversarial Domain Adaptation (CyCADA), which adapts representations at both the pixel-level and feature-level while enforcing semantic consistency. We enforce both structural and semantic consistency during adaptation using a cycle-consistency loss (ie. the source should match the source mapped to target mapped back to source) and semantics losses based on a particular visual recognition task. The semantics losses both guide the overall representation to be discriminative and enforce semantic consistency before and after mapping between domains. Our approach offers a unified domain adversarial learning model which combines the interpretability and low level structural consistency of prior image-level approaches [LT16; Bou+17; Shr+17; Zhu+17; LBK17] together with the regularization and strong empirical performance of prior feature-level approaches [GL15; Tze+17], as illustrated in Table 5.1.

We apply our CyCADA model to the task of digit recognition across domains and the task of semantic segmentation of urban scenes across domains. Experiments show that our model achieves competitive results on digit adaptation, cross-season adaptation in synthetic data, and on the challenging synthetic-to-real scenario. In the latter case, it improves per-pixel accuracy from 54% to 83%, nearly closing the gap to the target-trained model and providing 16% relative improvement over previous state-of-the-art methods.

We demonstrate that enforcing both semantic (task-specific) consistency and cycle consistency between input and stylized images prevents label flipping on the large shift between SVHN and MNIST (example, prevents a SVHN 9 from being mapped into an MNIST 2).

	Pixel Loss	Feature Loss	Semantic Consistent	Cycle Consistent
CycleGAN [Zhu+17]	✓			✓
Feature Adapt[†]		✓	✓	
Pixel Adapt[‡]	✓		✓	
CyCADA	✓	✓	✓	✓

Table 5.1: Our model, CyCADA, may use pixel, feature, and semantic information during adaptation while learning an invertible mapping through cycle consistency. [†][GL15; Tze+17], [‡][TPW17; Bou+17; LBK17]

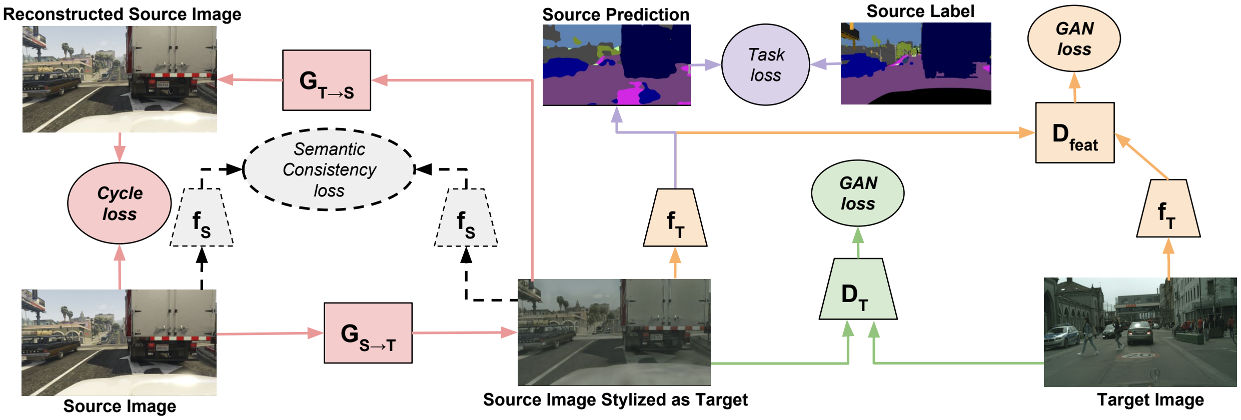


Figure 5.2: Cycle-consistent adversarial adaptation overview. By directly remapping source training data into the target domain, we remove the low-level differences between the domains, ensuring that our task model is well-conditioned on target data. We depict here the image-level adaptation as composed of the pixel GAN loss (green), the source cycle loss (red), and the source and target semantic consistency losses (black dashed) – used when needed to prevent label flipping. For clarity the target cycle is omitted. The feature-level adaptation is depicted as the feature GAN loss (orange) and the source task loss (purple).

On our semantic segmentation tasks (GTA to CityScapes) we did not observe label flipping to be a major source of error, even without the semantic consistency loss, but found cycle consistency to be critical. Because of this, and due to memory constraints, we focus on cycle consistency to preserve structure during transfer for the segmentation tasks. Overall, our experiments confirm that domain adaptation can benefit greatly from a combination of pixel and representation transformations, with the joint adaptation model achieving the highest performance across a range of visual recognition tasks.

5.1 Cycle-Consistent Adversarial Domain Adaption

We consider the problem of unsupervised adaptation, where we are provided source data X_S, source labels Y_S, and target data X_T, but no target labels. The goal is to learn a model f_T that correctly predicts the label for the target data X_T.

Pretrain Source Task Model. We begin by simply learning a source model f_S that can perform the task on the source data. For K-way classification with a cross-entropy loss, this corresponds to

$$\mathcal{L}_{\text{task}}(f_S, X_S, Y_S) = -\mathbb{E}_{(x_s,y_s)\sim(X_S,Y_S)} \sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log \left(\sigma(f_S^{(k)}(x_s))\right) \tag{5.1}$$

where σ denotes the softmax function. However, while the learned model f_S will perform well on the source data, typically domain shift between the source and target domain leads to reduced performance when evaluating on target data.

Pixel-level Adaptation. To mitigate the effects of domain shift, we follow previous adversarial adaptation approaches and learn to map samples across domains such that an adversarial discriminator is unable to distinguish the domains. By mapping samples into a common space, we enable our model to learn on source data while still generalizing to target data.

To this end, we introduce a mapping from source to target $G_{S\to T}$ and train it to produce target samples that fool an adversarial discriminator D_T. Conversely, the adversarial discriminator attempts to classify the real target data from the source target data. This corresponds to the loss function

$$\mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T, X_T, X_S) = \mathbb{E}_{x_t\sim X_T}\left[\log D_T(x_t)\right] + \mathbb{E}_{x_s\sim X_S}\left[\log(1 - D_T(G_{S\to T}(x_s)))\right]. \tag{5.2}$$

This objective ensures that $G_{S\to T}$, given source samples, produces convincing target samples. In turn, this ability to directly map samples between domains allows us to learn a target model f_T by minimizing $\mathcal{L}_{\text{task}}(f_T, G_{S\to T}(X_S), Y_S)$ (see Figure 5.2 green portion).

However, while previous approaches that optimized similar objectives have shown effective results, in practice they can often be unstable and prone to failure. Although the GAN loss in Equation 5.2 ensures that $G_{S\to T}(x_s)$ for some x_s will resemble data drawn from X_T, there is no way to guarantee that $G_{S\to T}(x_s)$ preserves the structure or content of the original sample x_s.

In order to encourage the source content to be preserved during the conversion process, we impose a cycle-consistency constraint on our adaptation method [Zhu+17; Yi+17; Kim+17] (see Figure 5.2 red portion). To this end, we introduce another mapping from target to source $G_{T\to S}$ and train it according to the same GAN loss $\mathcal{L}_{\text{GAN}}(G_{T\to S}, D_S, X_S, X_T)$. We then require that mapping a source sample from source to target and back to the source reproduces the original sample, thereby enforcing cycle-consistency. In other words, we want $G_{T\to S}(G_{S\to T}(x_s)) \approx x_s$ and $G_{S\to T}(G_{T\to S}(x_t)) \approx x_t$. This is done by imposing an L1 penalty on the reconstruction error, which is referred to as the *cycle-consistency loss*:

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G_{S\to T},&G_{T\to S},X_S,X_T) = \\ &\mathbb{E}_{x_s\sim X_S}\left[||G_{T\to S}(G_{S\to T}(x_s))-x_s||_1\right]+\mathbb{E}_{x_t\sim X_T}\left[||G_{S\to T}(G_{T\to S}(x_t))-x_t||_1\right].\end{aligned} \tag{5.3}$$

Additionally, as we have access to source labeled data, we explicitly encourage high semantic consistency before and after image translation. This helps to prevent label flipping described above and illustrated in Figure 5.4(a). We use the pretrained source task model f_S, as a noisy labeler by which we encourage an image to be classified in the same way after translation as it was before translation according to this classifier. Let us define the predicted label from a fixed classifier, f, for a given input X as $p(f,X) = \arg\max(f(X))$. Then we can define the semantic consistency before and after image translation as follows:

$$\begin{aligned}\mathcal{L}_{\text{sem}}(G_{S\to T},&G_{T\to S},X_S,X_T,f_S) = \\ &\mathcal{L}_{\text{task}}(f_S,G_{T\to S}(X_T),p(f_S,X_T))+\mathcal{L}_{\text{task}}(f_S,G_{S\to T}(X_S),p(f_S,X_S))\end{aligned} \tag{5.4}$$

See Figure 5.2 black portion. This can be viewed as analogous to content losses in style transfer [GEB16] or in pixel adaptation [TPW17], where the shared content to preserve is dictated by the source task model f_S.

Feature-level Adaptation. We have thus far described an adaptation method which combines cycle consistency, semantic consistency, and adversarial objectives to produce a final target model. As a pixel-level method, the adversarial objective consists of a discriminator which distinguishes between two image sets, e.g. transformed source and real target image. Note that we could also consider a feature-level method which discriminates between the features or semantics from two image sets as viewed under a task network. This would amount to an additional feature level GAN loss (see Figure 5.2 orange portion):

$$\mathcal{L}_{\text{GAN}}(f_T,D_{\text{feat}},f_S(G_{S\to T}(X_S)),X_T). \tag{5.5}$$

Taken together, these loss functions form our complete objective:

$$\begin{aligned}\mathcal{L}_{\text{CyCADA}}&(f_T,X_S,X_T,Y_S,G_{S\to T},G_{T\to S},D_S,D_T) \\ &=\mathcal{L}_{\text{task}}(f_T,G_{S\to T}(X_S),Y_S) \\ &+\mathcal{L}_{\text{GAN}}(G_{S\to T},D_T,X_T,X_S) \\ &+\mathcal{L}_{\text{GAN}}(G_{T\to S},D_S,X_S,X_T) \\ &+\mathcal{L}_{\text{GAN}}(f_T,D_{\text{feat}},f_S(G_{S\to T}(X_S)),X_T) \\ &+\mathcal{L}_{\text{cyc}}(G_{S\to T},G_{T\to S},X_S,X_T) \\ &+\mathcal{L}_{\text{sem}}(G_{S\to T},G_{T\to S},X_S,X_T,f_S).\end{aligned} \tag{5.6}$$

This ultimately corresponds to solving for a target model f_T according to the optimization problem

$$f_T^* = \arg\min_{f_T}\min_{\substack{G_{S\to T}\\G_{T\to S}}}\max_{D_S,D_T}\mathcal{L}_{\text{CyCADA}} \tag{5.7}$$

Model	MNIST → USPS	USPS → MNIST	SVHN → MNIST
Source only	82.2 ± 0.8	69.6 ± 3.8	67.1 ± 0.6
DANN [Gan+16]	-	-	73.6
DTN [TPW17]	-	-	84.4
CoGAN [LT16]	91.2	89.1	-
ADDA [Tze+17]	89.4 ± 0.2	90.1 ± 0.8	76.0 ± 1.8
PixelDA [Bou+17]	**95.9**	-	-
UNIT [LBK17]	**95.9**	93.6	**90.5***
CyCADA (Ours)	**95.6** ± 0.2	**96.5** ± 0.1	**90.4** ± 0.4
Target Fully Supervised	96.3 ± 0.1	99.2 ± 0.1	99.2 ±0.1

Table 5.2: **Unsupervised domain adaptation across digit datasets.** Our model is competitive with or outperforms state-of-the-art models for each shift. For the difficult shift of SVHN to MNIST we also note that feature space adaptation provides additional benefit beyond the pixel-only adaptation. *UNIT trains with the extended SVHN (>500K images vs ours 72K).

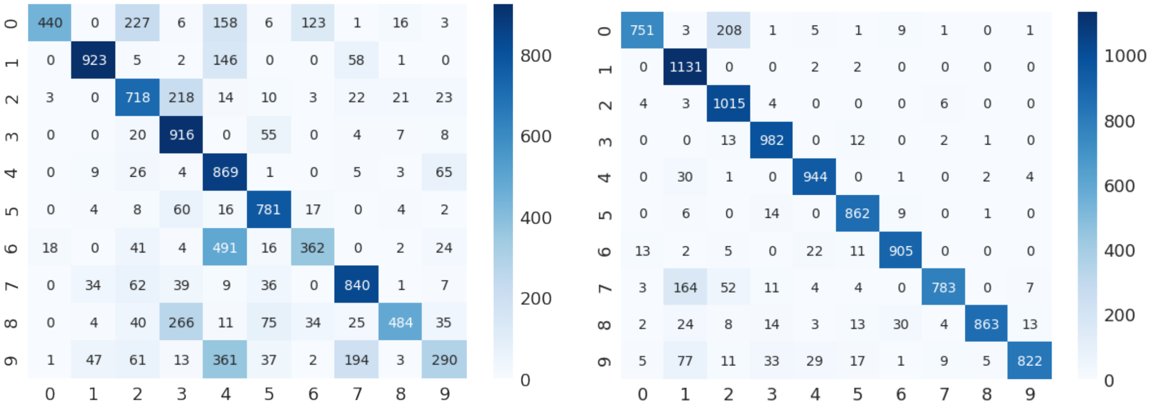
We have proposed a method for unsupervised adaptation which views prior adversarial objectives as operating at the pixel or feature level and generalizes to a method which may benefit from both approaches. In addition, we combine cycle-consistency together with semantic transformation constraints to regularize the mapping from one domain to another. In the next section, we apply CyCADA to both digit classification and to semantic segmentation, implementing G_S and G_T as a pixel-to-pixel convnet, f_S and f_T as a convnet classifier or a Fully-Convolutional Net (FCN), and D_S, D_T, and D_{feat} as a convnet with binary outputs.

5.2 Experiments

We evaluate CyCADA on several unsupervised adaptation scenarios. We first focus on adaptation for digit classification using the MNIST [LeC+98], USPS, and Street View House Numbers (SVHN) [Net+11] datasets. Next, we present results for the task of semantic image segmentation, using the GTA [Ric+16] and CityScapes [Cor+16] datasets, as well as an additional experiment with the SYNTHIA [Ros+16a] dataset.

5.2.1 Digit Adaptation

We evaluate our method across the adaptation shifts of USPS to MNIST, MNIST to USPS, and SVHN to MNIST. We train our model using the training sets, MNIST - 60,000 images, USPS - 7,291 images, standard SVHN train - 73,257 images. Evaluation is reported on the standard test sets: MNIST - 10,000 images, USPS - 2,007 images. We report classification accuracy for each shift compared to prior work and relevant baselines in Table 5.2 and find



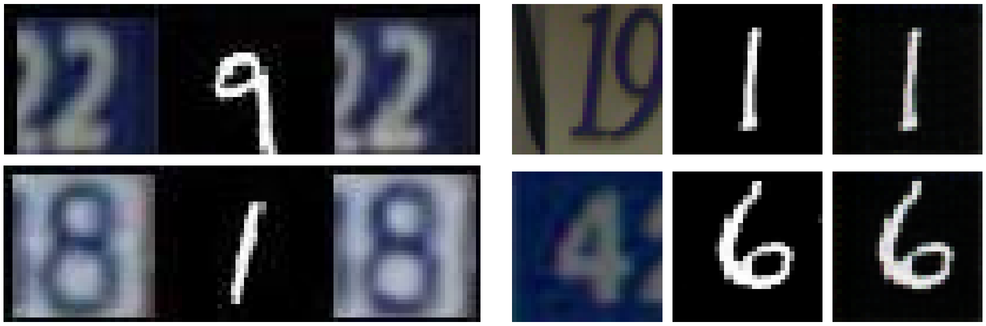
(a) Source only Model (b) CyCADA model

Figure 5.3: Confusion matrices for the SVHN → MNIST experiment before and after adaptation.

Model	Accuracy (%)
Source only	67.1
CyCADA - no feat adapt, no semantic loss	70.3
CyCADA - no feat adapt	71.2
CyCADA - no cycle consistency	75.7
CyCADA - no pixel adapt	83.8
CyCADA (Full)	**90.4**
Target Fully Supervised	99.2

Table 5.3: **Ablation of CyCADA on SVHN→MNIST Domain Shift.** We show that each component of our method, joint feature and pixel space adaptation, with semantic and cycle losses during pixel adaptation, contributes to the overall performance.

that our method outperforms competing approaches on average. For all digit experiments we use a variant of the LeNet architecture as the task net (Figure 5.5 *left*). Our feature discriminator network consists of 3 fully connected layers (Figure 5.5 *mid left*). The image discriminator network consists of 6 convolutional layers culminating in a single value per pixel (Figure 5.5 *mid right*). Finally, to generate one image domain from another we use a multilayer network which consists of convolution layers followed by two residual blocks and then deconvolution layers (Figure 5.5 *right*). All stages are trained using the Adam optimizer. For training the source task net model, we use learning rate 1e-4 and train for 100 epochs over the data with batch size 128. For feature space adaptation we use learning rate 1e-5 and train for max 200 epochs over the data. For pixel space adaptation we train our generators and discriminators with equal weighting on all losses, use batch size 100, learning rate 2e-4



(a) Without Semantic Loss (b) Without Cycle Loss

Figure 5.4: **Ablation: Effect of Semantic or Cycle Consistency.** Each triple contains the SVHN image (*left*), the image translated into MNIST style (*middle*), and the image reconstructed back into SVHN (*right*). (a) Without semantic loss, both the GAN and cycle constraints are satisfied (translated image matches MNIST style and reconstructed image matches original), but the image translated to the target domain lacks the proper semantics. (b) Without cycle loss, the reconstruction is not satisfied and though the semantic consistency leads to some successful semantic translations (*top*) there are still cases of label flipping (*bottom*).

(default from CycleGAN), and trained for 50 epochs. We ran each experiment 4 times and report the average and standard error across the runs.

Note, for the relatively easy shift of MNIST to USPS our method performs comparably with state-of-the-art approaches. In the reverse direction, when adapting from USPS images to MNIST, which involves a fraction of the supervised digit labeled data, our method outperforms competing approaches. For SVHN to MNIST our method outperforms all other deep distribution alignment approaches except for UNIT [LBK17], but the reported performance in UNIT uses the extended training set of ¿500,000 images from SVHN whereas we report performance using the standard set of only 73,257 images.

To further understand the types of mistakes which are improved upon and those which still persist after adaptation, we present the confusion matrices before and after our approach for the digit experiment of SVHN to MNIST (Figure 5.3). Before adaptation we see common confusions are 0s with 2s, 4s, and 7s. 6 with 4, 8 with 3, and 9 with 4. After adaptation all errors are reduced, but we still find that 7s are confused with 1s and 0s with 2s. These errors make some sense as with hand written digits, these digits sometimes resemble one another. It remains an open question to produce a model which may overcome these types of errors between highly similar classes.

Next, we perform a sequence of ablation studies on the three parts of our model. Table 5.3 reports the quantitative performance gain on the SVHN→MNIST domain shift for removing each piece of the model, demonstrating the importance of including each component. We also discuss and show qualitative comparisons below.

Ablation: Pixel vs Feature Level Transfer. We begin by evaluating the contribution

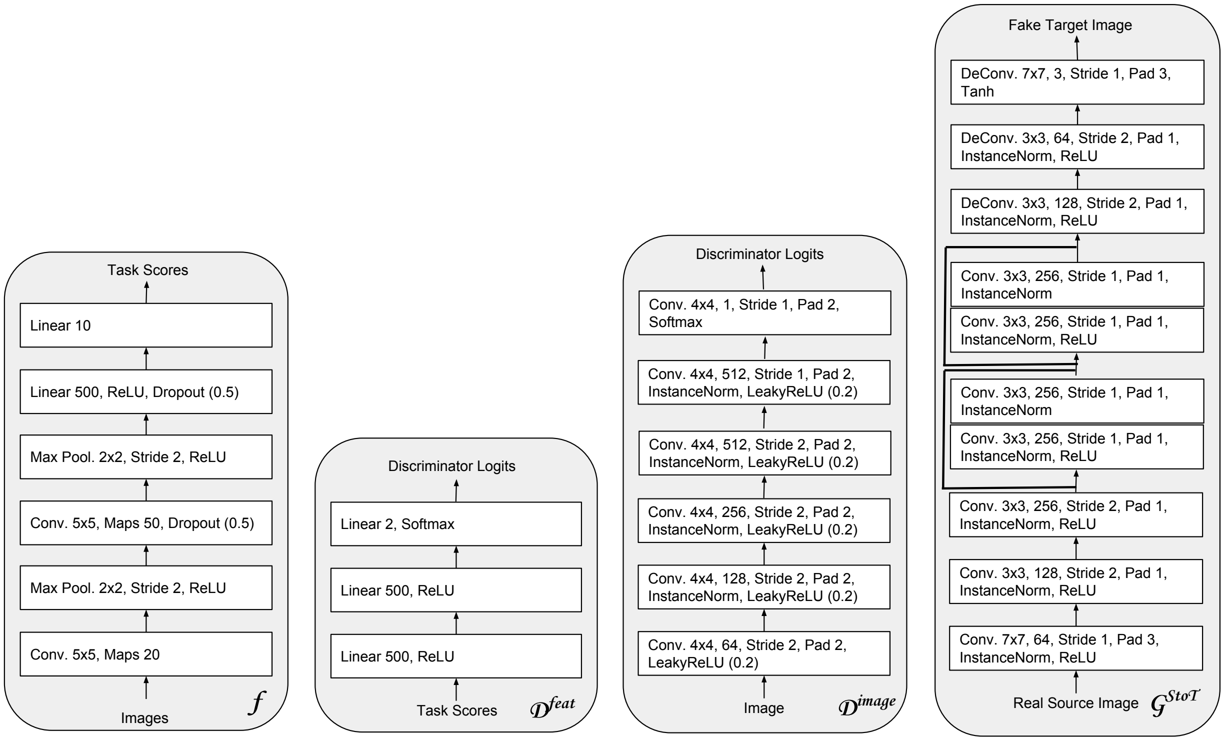
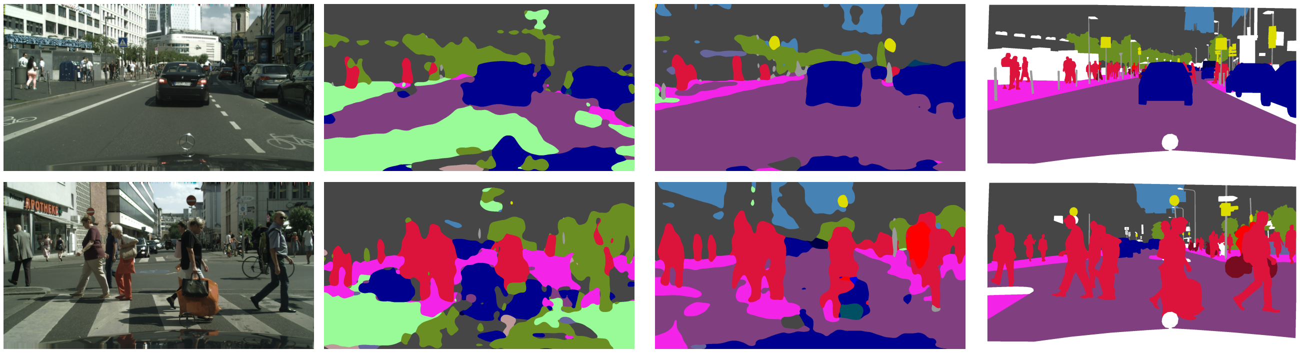

Task Scores
Linear 10
Linear 500, ReLU, Dropout (0.5)
Max Pool. 2x2, Stride 2, ReLU
Conv. 5x5, Maps 50, Dropout (0.5)
Max Pool. 2x2, Stride 2, ReLU
Conv. 5x5, Maps 20
Images
f
Discriminator Logits
Linear 2, Softmax
Linear 500, ReLU
Linear 500, ReLU
Task Scores
D^feat
Discriminator Logits
Conv. 4x4, 1, Stride 1, Pad 2, Softmax
Conv. 4x4, 512, Stride 1, Pad 2, InstanceNorm, LeakyReLU (0.2)
Conv. 4x4, 512, Stride 2, Pad 2, InstanceNorm, LeakyReLU (0.2)
Conv. 4x4, 256, Stride 2, Pad 2, InstanceNorm, LeakyReLU (0.2)
Conv. 4x4, 128, Stride 2, Pad 2, InstanceNorm, LeakyReLU (0.2)
Conv. 4x4, 64, Stride 2, Pad 2, LeakyReLU (0.2)
Image
D^image
Fake Target Image
DeConv. 7x7, 3, Stride 1, Pad 3, Tanh
DeConv. 3x3, 64, Stride 2, Pad 1, InstanceNorm, ReLU
DeConv. 3x3, 128, Stride 2, Pad 1, InstanceNorm, ReLU
Conv. 3x3, 256, Stride 1, Pad 1, InstanceNorm
Conv. 3x3, 256, Stride 1, Pad 1, InstanceNorm, ReLU
Conv. 3x3, 256, Stride 1, Pad 1, InstanceNorm
Conv. 3x3, 256, Stride 1, Pad 1, InstanceNorm, ReLU
Conv. 3x3, 256, Stride 2, Pad 1, InstanceNorm, ReLU
Conv. 3x3, 128, Stride 2, Pad 1, InstanceNorm, ReLU
Conv. 7x7, 64, Stride 1, Pad 3, InstanceNorm, ReLU
Real Source Image
G^StoT

Figure 5.5: Network architectures used for digit experiments. We show here the task net (f), discriminator for feature level adaptation (D^{feat}), discriminator for image level adaptation (D^{image}), and generator for source to target (G) – same network used for target to source.



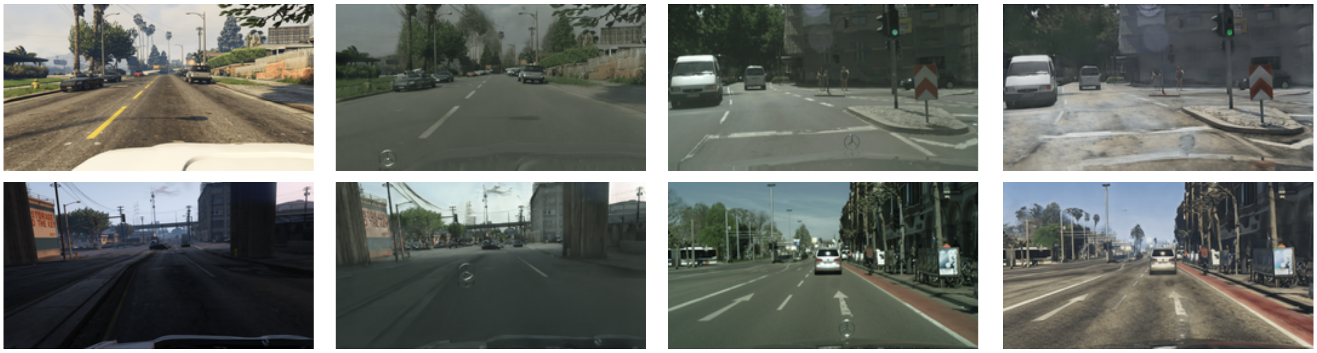
(a) Test Image (b) Source Prediction (c) CyCADA Prediction (d) Ground Truth

Figure 5.6: **GTA5 to CityScapes Semantic Segmentation.** Each test CityScapes image (a) along with the corresponding predictions from the source only model (b) and our CyCADA model (c) are shown and may be compared against the ground truth annotation (d).

GTA5 → Cityscapes

	Architecture	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle	**mIoU**	**fwIoU**	**Pixel acc.**
Source only	A	26.0	14.9	65.1	5.5	12.9	8.9	6.0	2.5	70.0	2.9	47.0	24.5	0.0	40.0	12.1	1.5	0.0	0.0	0.0	17.9	41.9	54.0
FCN-wld [Hof+16]	A	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0	27.1	-	-
CDA [ZDG17]	A	26.4	22.0	74.7	6.0	11.9	8.4	16.3	11.1	75.7	13.3	**66.5**	38.0	**9.3**	55.2	**18.8**	18.9	0.0	**16.8**	**14.6**	27.8	-	-
FCTN [ZLK18]	A	72.2	28.4	74.9	18.3	10.8	**24.0**	**25.3**	17.9	80.1	**36.7**	61.1	44.7	0.0	74.5	8.9	1.5	0.0	0.0	0.0	30.5	-	-
CyCADA (Ours)	A	**85.2**	**37.2**	**76.5**	**21.8**	**15.0**	23.8	22.9	**21.5**	**80.5**	31.3	60.7	**50.5**	9.0	**76.9**	17.1	**28.2**	**4.5**	9.8	0.0	**35.4**	**73.8**	**83.6**
Oracle - Target Supervised	A	96.4	74.5	87.1	35.3	37.8	36.4	46.9	60.1	89.0	54.3	89.8	65.6	35.9	89.4	38.6	64.1	38.6	40.5	65.1	60.3	87.6	93.1
Source only	B	42.7	26.3	51.7	5.5	6.8	13.8	23.6	6.9	75.5	11.5	36.8	49.3	0.9	46.7	3.4	5.0	0.0	5.0	1.4	21.7	47.4	62.5
CyCADA (Ours)	B	**79.1**	**33.1**	**77.9**	**23.4**	**17.3**	**32.1**	**33.3**	31.8	**81.5**	**26.7**	**69.0**	**62.8**	14.7	**74.5**	**20.9**	**25.6**	**6.9**	**18.8**	20.4	**39.5**	**72.4**	**82.3**
Oracle - Target Supervised	B	97.3	79.8	88.6	32.5	48.2	56.3	63.6	73.3	89.0	58.9	93.0	78.2	55.2	92.2	45.0	67.3	39.6	49.9	73.6	67.4	89.6	94.3

Table 5.4: Adaptation between GTA5 and Cityscapes, showing IoU for each class and mean IoU, freq-weighted IoU and pixel accuracy. CyCADA significantly outperforms baselines, nearly closing the gap to the target-trained oracle on pixel accuracy. We compare our model using two base semantic segmentation architectures (A) VGG16-FCN8s [LSD15] base network and (B) DRN-26 [YKF17].



(a) GTA5 (b) GTA5 → Cityscapes (c) CityScapes (d) CityScapes → GTA5

Figure 5.7: **GTA5 to CityScapes Image Translation.** Example images from the GTA5 (a) and Cityscapes (c) datasets, alongside their image-space conversions to the opposite domain, (b) and (d), respectively. Our model achieves highly realistic domain conversions.

of the pixel space and feature space transfer. We find that in the case of the small domain shifts between USPS and MNIST, the pixel space adaptation by which we train a classifier using images translated using CycleGAN [Zhu+17], performs very well, outperforming or comparable to prior adaptation approaches. Feature level adaptation offers a small benefit in this case of a small pixel shift. However, for the more difficult shift of SVHN to MNIST, we find that feature level adaptation outperforms the pixel level adaptation, and importantly, both may be combined to produce an overall model which outperforms all competing methods.

Ablation: No Semantic Consistency. We experiment without the addition of our semantic consistency loss and find that the standard unsupervised CycleGAN approach diverged when training SVHN to MNIST often suffering from random label flipping. Figure 5.4(a) demonstrates two examples where cycle constraints alone fail to produce the desired behavior for our end task. An SVHN image is mapped to a convincing MNIST style image and back to a SVHN image with correct semantics. However, the MNIST-like image has mismatched semantics. Our proposed approach uses the source labels to train a weak classification model which can be used to enforce semantic consistency before and after translation, resolving this issue.

Ablation: No Cycle Consistency. We study the importance of the cycle consistency loss. First note that without this loss there is no reconstruction guarantee, thus in Figure 5.4(b) we see that the translation back to SVHN fails. In addition, we find that while the semantic loss does encourage correct semantics it relies on the weak source labeler and thus label flipping still occurs (see right image triple).

5.2.2 Semantic Segmentation Adaptation

Next, we evaluate CyCADA on semantic segmentation. The task is to assign a semantic label to each pixel in the input image, e.g. *road*, *building*, etc. We limit our evaluation to the unsupervised adaptation setting, where labels are only available in the source domain, but we are evaluated solely on our performance in the target domain.

For each experiment, we use report three metrics of overall performance. Let n_{ij} be the number of pixels of class i predicted as class j, let $t_i = \sum_j n_{ij}$ be the total number of pixels of class i, and let N be the number of classes. Our three evaluation metrics are, mean intersection-over-union (mIoU), frequency weighted intersection-over-union (fwIoU), and pixel accuracy, which are defined as follows:

$$\textbf{mIoU} = \frac{1}{N} \cdot \frac{\sum_i n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

$$\textbf{fwIoU} = \frac{1}{\sum_k t_k} \cdot \frac{\sum_i n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

$$\textbf{Pixel accuracy} = \frac{\sum_i n_{ii}}{\sum_i t_i}.$$

Cycle-consistent adversarial adaptation is general and can be applied at any layer of a network. Since optimizing the full CyCADA objective in Equation 5.6 end-to-end is memory-intensive in practice, we train our model in stages. First, we perform image-space adaptation and map our source data into the target domain. Next, using the adapted source data with the original source labels, we learn a task model that is suited to operating on target data. Finally, we perform another round of adaptation between the adapted source data and the target data in feature-space, using one of the intermediate layers of the task model. Additionally, we do not use the semantic loss for the segmentation experiments as it would require loading generators, discriminators, and an additional semantic segmenter into memory all at once for two images. We did not have the required memory for this at the time of submission, but leave it to future work to deploy model parallelism or experiment with larger GPU memory.

To demonstrate our method's applicability to real-world adaptation scenarios, we also evaluate our model in a challenging synthetic-to-real adaptation setting. For our synthetic source domain, we use the GTA5 dataset [Ric+16] extracted from the game Grand Theft Auto V, which contains 24966 images. We consider adaptation from GTA5 to the real-world Cityscapes dataset [Cor+16], from which we used 19998 images without annotation for training and 500 images for validation. Both of these datasets are evaluated on the same set of 19 classes, allowing for straightforward adaptation between the two domains. We also provide an additional experiment evaluating cross-season adaptation in synthetic environments in Section 5.2.2.

Image-space adaptation also affords us the ability to visually inspect the results of the adaptation method. This is a distinct advantage over opaque feature-space adaptation methods, especially in truly unsupervised settings—without labels, there is no way to empirically evaluate the adapted model, and thus no way to verify that adaptation is improving task performance. Visually confirming that the conversions between source and target images are reasonable, while not a *guarantee* of improved task performance, can serve as a sanity check to ensure that adaptation is not completely diverging. This process is diagrammed in Figure 5.2.

Synthetic to real adaptation

To evaluate our method's applicability to real-world adaptation settings, we investigate adaptation from synthetic to real-world imagery. The results of this evaluation are presented in Table 5.4, ablation in Table 5.5, with qualitative results shown in Figure 5.6. We experiment with two different base architectures: the commonly used VGG16-FCN8s [LSD15] architecture as well as the state-of-the-art DRN-26 [YKF17] architecture.

For FCN8s, we train our source semantic segmentation model for 100k iterations using SGD with learning rate 1e-3 and momentum 0.9. For the DRN-26 architecture, we train our source semantic segmentation model for 115K iterations using SGD with learning rate 1e-3 and momentum 0.9. We use a crop size of 600x600 and a batch size of 8 for this training. For cycle-consistent image level adaptation, we followed the network architecture and hyperparameters of CycleGAN[Zhu+17]. All images were resized to have width of 1024 pixels while keeping the aspect ratio, and the training was performed with randomly cropped patches of size 400 by 400. Also, due to large size of the dataset, we trained only 20 epochs. For feature level adaptation, we train using SGD with momentum, 0.99, and learning rate 1e-5.

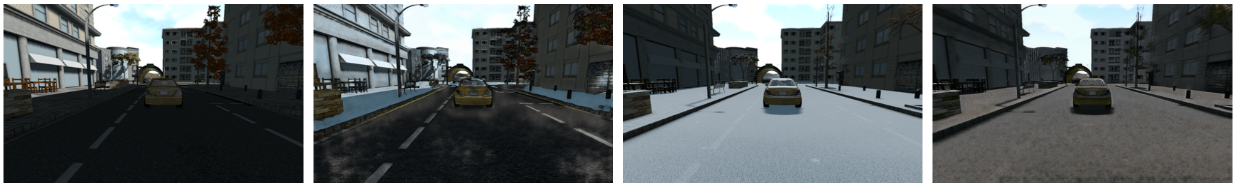
GTA5 → Cityscapes				
	Architecture	**mIoU**	**fwIoU**	**Pixel acc.**
Source only	A	17.9	41.9	54.0
CyCADA feat-only	A	29.2	71.5	82.5
CyCADA pixel-only no cycle	A	19.8	55.7	70.5
CyCADA pixel-only	A	34.8	73.1	82.8
CyCADA (Full)	A	**35.4**	**73.8**	**83.6**
Oracle - Target Supervised	A	60.3	87.6	93.1
Source only	B	21.7	47.4	62.5
CyCADA feat-only	B	31.7	67.4	78.4
CyCADA pixel-only no cycle	B	19.7	54.5	69.9
CyCADA pixel-only	B	37.0	63.8	75.4
CyCADA (Full)	B	**39.5**	**72.4**	**82.3**
Oracle - Target Supervised	B	67.4	89.6	94.3

Table 5.5: Ablation of our method, CyCADA on the GTA5 to Cityscapes adaptation. We compare our model using two base semantic segmentation architectures (A) VGG16-FCN8s [LSD15] base network and (B) DRN-26 [YKF17].

We weight the representation loss ten times less than the discriminator loss as a convenience since otherwise the discriminator did not learn a suitable model within a single epoch. Then the segmentation model was trained separately using the adapted source images and the ground truth labels of the source data. Due to memory limitations we can only include a single source and single target image at a time (crops of size 768x768), this small batch is one of the main reasons for using a high momentum parameter.

Once again, CyCADA achieves state-of-the-art results, recovering approximately 40% of the performance lost to domain shift. CyCADA also improves or maintains performance on all 19 classes. Examination of fwIoU and pixel accuracy as well as individual class IoUs reveals that our method performs well on most of the common classes. Although some classes such as *train* and *bicycle* see little or no improvement, we note that those classes are poorly represented in the GTA5 data, making recognition very difficult. We compare our model against [Shr+17] for this setting, but found this approach did not converge and resulted in worse performance than the source only model (see Appendix for full details).

We visualize the results of image-space adaptation between GTA5 and Cityscapes in Figure 5.7. The most obvious difference between the original images and the adapted images is the saturation levels—the GTA5 imagery is much more vivid than the Cityscapes imagery, so adaptation adjusts the colors to compensate. We also observe texture changes, which are perhaps most apparent in the road: in-game, the roads appear rough with many blemishes,



(a) Fall (b) Fall → Winter (c) Winter (d) Winter → Fall

Figure 5.8: **Cross Season Image Translation.** Example image-space conversions for the SYNTHIA seasons adaptation setting. We show real samples from each domain (Fall and Winter) alongside conversions to the opposite domain.

SYNTHIA Fall → Winter

	sky	building	road	sidewalk	fence	vegetation	pole	car	traffic sign	pedestrian	bicycle	lanemarking	traffic light	**mIoU**	**fwIoU**	**Pixel acc.**
Source only	91.7	80.6	79.7	12.1	71.8	44.2	26.1	42.8	49.0	38.7	45.1	41.3	24.5	49.8	71.7	82.3
FCNs in the wild	92.1	86.7	91.3	20.8	72.7	**52.9**	**46.5**	64.3	**50.0**	**59.5**	**54.6**	**57.5**	**26.1**	59.6	—	—
CyCADA pixel-only	**92.5**	**90.1**	**91.9**	**79.9**	**85.7**	47.1	36.9	**82.6**	45.0	49.1	46.2	54.6	21.5	**63.3**	**85.7**	**92.1**
Oracle (Train on target)	93.8	92.2	94.7	90.7	90.2	64.4	38.1	88.5	55.4	51.0	52.0	68.9	37.3	70.5	89.9	94.5

Table 5.6: Adaptation between seasons in the SYNTHIA dataset. We report IoU for each class and mean IoU, freq-weighted IoU and pixel accuracy. Our CyCADA method achieves state-of-the-art performance on average across all categories. *FCNs in the wild is by [Hof+16].

but Cityscapes roads tend to be fairly uniform in appearance, so in converting from GTA5 to Cityscapes, our model removes most of the texture. Somewhat amusingly, our model has a tendency to add a hood ornament to the bottom of the image, which, while likely irrelevant to the segmentation task, serves as a further indication that image-space adaptation is producing reasonable results.

Cross-season adaptation

As an additional semantic segmentation evaluation, we consider the SYNTHIA dataset [Ros+16a], which contains synthetic renderings of urban scenes. We use the SYNTHIA video sequences, which are rendered across a variety of environments, weather conditions, and lighting conditions. This provides a synthetic testbed for evaluating adaptation techniques. For comparison with previous work, in this work we focus on adaptation between seasons. We use only the front-facing views in the sequences so as to mimic dashcam imagery, and adapt from fall to winter. The subset of the dataset we use contains 13 classes and consists of 10,852 fall images and 7,654 winter images.

We start by exploring the abilities of pixel space adaptation alone (using FCN8s architecture) for the setting of adapting across seasons in synthetic data. For this we use the

SYNTHIA dataset and adapt from fall to winter weather conditions. Typically in unsupervised adaptation settings it is difficult to interpret what causes the performance improvement after adaptation. Therefore, we use this setting as an example where we may directly visualize the shift from fall to winter and inspect the intermediate pixel level adaptation result from our algorithm. In Figure 5.8 we show the result of pixel only adaptation as we generate a winter domain image (b) from a fall domain image (a), and visa versa (c-d). We may clearly see the changes of adding or removing snow. This visually interpretable result matches our expectation of the true shift between these domains and indeed results in favorable final semantic segmentation performance from fall to winter as shown in Table 5.6. We find that CyCADA achieves state-of-the-art performance on this task with image space adaptation alone, however does not recover full supervised learning performance (train on target). Some example errors includes adding snow to the sidewalks, but not to the road, while in the true winter domain snow appears in both locations. However, even this mistake is interesting as it implies that the model is learning to distinguish road from sidewalk during pixel adaptation, despite the lack of pixel annotations.

Cycle-consistent adversarial adaptation achieves state-of-the-art adaptation performance. We see that under the fwIoU and pixel accuracy metrics, CyCADA approaches oracle performance, falling short by only a few points, despite being entirely unsupervised. This indicates that CyCADA is extremely effective at correcting the most common classes in the dataset. This conclusion is supported by inspection of the individual classes in Table 5.6, where we see the largest improvement on common classes such as *road* and *sidewalk*.

Additional GTA to CityScapes Visualizations

We show additional image-space adaptation visualiztions for the GTA to CityScapes scenario in Figure 5.9.

5.3 Related Work

The problem of visual domain adaptation was introduced along with a pairwise metric transform solution by [Sae+10] and was further popularized by the broad study of visual dataset bias [TE11]. Early deep adaptive works focused on feature space alignment through minimizing the distance between first or second order feature space statistics of the source and target [Tze+14; LW15]. These latent distribution alignment approaches were further improved through the use of domain adversarial objectives whereby a domain classifier is trained to distinguish between the source and target representations while the domain representation is learned so as to maximize the error of the domain classifier. The representation is optimized using the standard minimax objective [GL15], the symmetric confusion objective [Tze+15], or the inverted label objective [Tze+17]. Each of these objectives is related to the literature on generative adversarial networks [Goo+14] and follow-up work for improved training procedures for these networks [Sal+16; ACB17].

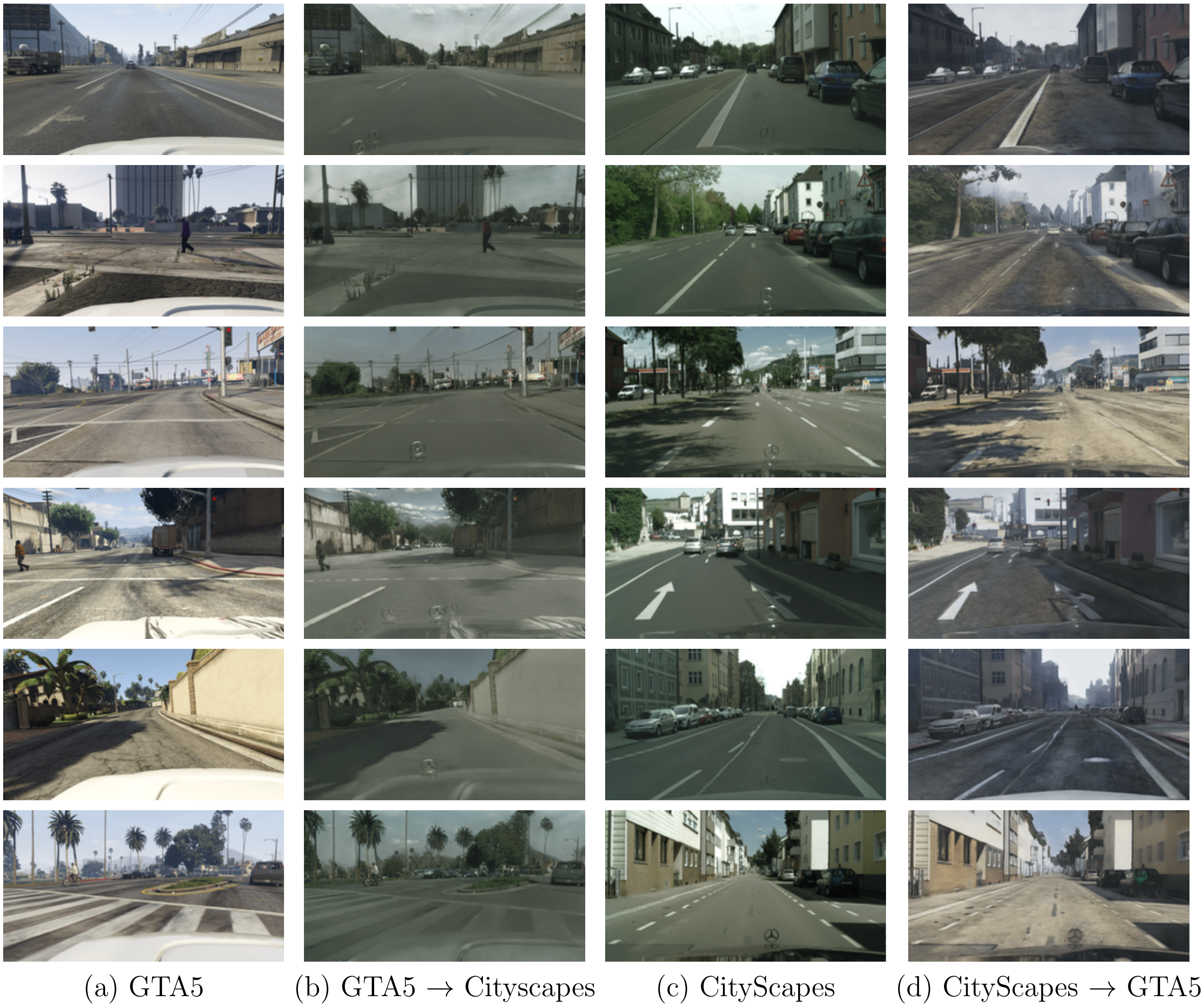


Figure 5.9: **GTA5 to CityScapes Image Translation.** Example images from the GTA5 (a) and Cityscapes (c) datasets, alongside their image-space conversions to the opposite domain, (b) and (d), respectively. Our model achieves highly realistic domain conversions.

The feature-space adaptation methods described above focus on modifications to the discriminative representation space. In contrast, other recent methods have sought adaptation in the pixel-space using various generative approaches. One advantage of pixel-space adaptation, as we have shown, is that the result may be more human interpretable, since an image from one domain can now be visualized in a new domain. CoGANs [LT16] jointly learn a source and target representation through explicit weight sharing of certain layers while each source and target has a unique generative adversarial objective. [Ghi+16] uses an additional reconstruction objective in the target domain to encourage alignment in the unsupervised adaptation setting.

In contrast, another approach is to directly convert the target image into a source style image (or visa versa), largely based on Generative Adversarial Networks (GANs) [Goo+14]. Researchers have successfully applied GANs to various applications such as image generation [DCF+15; RMC16; ZML17], image editing [Zhu+16] and feature learning [Sal+16; DKD17]. Recent work [Iso+17; San+17; Kar+16] adopt conditional GANs [MO14] for these image-to-image translation problems [Iso+17], but they require input-output image pairs for training, which is in general not available in domain adaptation problems.

There also exist lines of work where such training pairs are not given. [Yoo+16] learns a source to target encoder-decoder along with a generative adversarial objective on the reconstruction which is is applied for predicting the clothing people are wearing. The Domain Transfer Network [TPW17] trains a generator to transform a source image into a target image by enforcing consistency in the embedding space. [Shr+17] instead uses an L1 reconstruction loss to force the generated target images to be similar to their original source images.This works well for limited domain shifts where the domains are similar in pixel-space, but can be too limiting for settings with larger domain shifts. [LBK17] considers learning unique encoders which reach a shared latent space and can be reconstructed into the same domnain or translated into the other domain. Manually defined sharing of certain layers are used to encourage consistency between the two domain models. [Bou+17] use a content similarity loss to ensure the generated target image is similar to the original source image; however, this requires prior knowledge about which parts of the image stay the same across domains (e.g. foreground). Our method does not require pre-defining what content is shared between domains and instead simply translates images back to their original domains while ensuring that they remain identical to their original versions. BiGAN/ALI [DKD17; Dum+17] take an approach of simultaneously learning the transformations between the pixel and the latent space. Cycle-consistent Adversarial Networks (CycleGAN) [Zhu+17] produced compelling image translation results such as generating photorealistic images from impressionism paintings or transforming horses into zebras at high resolution using the cycle-consistency loss. This loss was simultaneously proposed by [Yi+17] and [Kim+17] to great effect as well. Our motivation comes from such findings about the effectiveness of the cycle-consistency loss.

An approach to adaptation which is complementary to this work involves seeking to produce approximate labels for the target domain and incorporate those into the training set for supervised learning [Hae+17].

Few works have explicitly studied visual domain adaptation for the semantic segmentation task. Adaptation across weather conditions in simple road scenes was first studied by [LF13]. More recently, a convolutional domain adversarial based approached was proposed for more general drive cam scenes and for adaptation from simulated to real environments [Hof+16]. [Ros+16b] learns a multi-source model through concatenating all available labeled data and learning a single large model and then transfers to a sparsely labeled target domain through distillation [HVD14]. [Che+17] use an adversarial objective to align both global and class-specific statistics, while mining additional temporal data from street view datasets to learn a static object prior. [ZDG17] instead perform segmentation adaptation by aligning label distributions both globally and across superpixels in an image.

5.4 Conclusion

We proposed an unsupervised domain adversarial learning method that unifies cycle-consistent image translation adversarial models with adversarial adaptation methods. CyCADA offers the interpretability of image-space adaptation, by visualizing the intermediate output of our method, while producing a discriminative and task relevant model through semantic consistency and representation space adaptation. We experimentally validated our model on a variety of adaptation tasks including digit adaptation and synthetic to real adaptation for semantic segmentation of driving scenes. We presented extensive ablations of our method demonstrating the importance of each component of our method, where the combination results in a state-of-the-art approach.

Chapter 6

SPLAT: Semantic Pixel-Level Adaptation Transforms for Detection

Image and pixel-level adaptation of image classifiers are relatively well studied (see citations below), but detection adaptation methods have recieved less attention. Existing detection adaptation approaches either pre-date deep learning (e.g., [Don+13]) or employ a domain classification loss on the proposal generating layer and box classifiction network head to align feature distributions across domains [Che+18].

When evaluated on benchmark challenges for driving detection domain adaptation, existing methods perform poorly, e.g., recovering less than half of the domain shift when adapting from *Sim10k* [JR+17] (a synthetic detection dataset generated from Grand Theft Auto V) to *Cityscapes* [Cor+16]. One potential reason is that feature distribution alignment is suboptimal when samples include many background patches or whole-image scenes that differ in object layout between domains. In contrast, adaptation approaches based on learning pixel-based image-to-image transforms [Hof+18] have shown promise for semantic segmentation tasks, recovering up to 84% of source-to-target loss, but this class of methods have not been heretofore reported for the task of detection.

In this chapter, we propose an approach to detection domain adaptation based on pixel-transformed source-to-target imagery. Our Semantic Pixel-Level Adaptation Transform (SPLAT) method efficiently generates cross-domain image pairs and employs pseudo-label losses and/or alignment losses on paired images when training a target domain detector. Figure 6.1 summarizes the problem statement and Figure 6.2 overviews our general approach. Our experiments confirm that pixel-level transformations provide a significant boost for detection adaptation compared to previous methods using domain-confusion losses and outperform all published baselines on the benchmark *Sim10k-to-CityScapes* challenge by a large margin.

State-of-the-art pixel transformations for domain adaptation have often been based on cycle-transforms, e.g., as used in Cycle-GAN [Zhu+17] and CyCADA [Hof+18]. However cycle-

This chapter is based on joint work with Kaylee Burns, Kate Saenko, and Trevor Darrell [Tze+18].

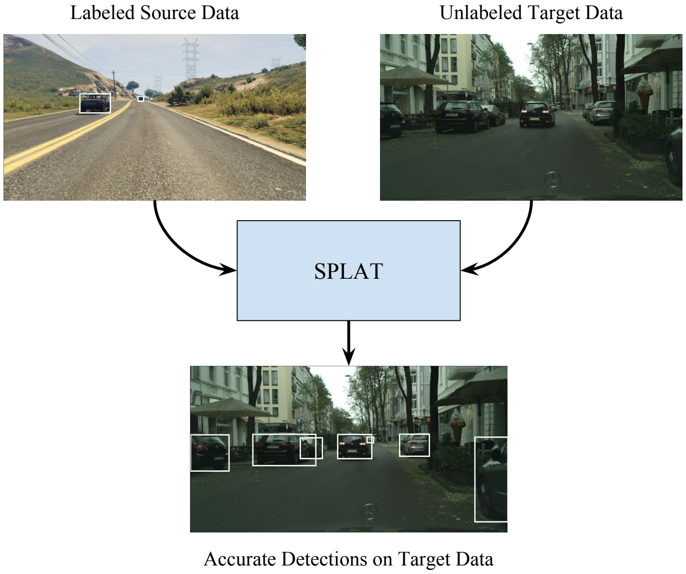


Figure 6.1: We propose a new method, SPLAT, that tackles the task of unsupervised domain adaptation for detection. SPLAT is trained on labeled source data and unlabeled target data, and learns to produce accurate detections in the target domain. We improve detection results by 12.5 points over previous methods, setting the new state of the art.

consistency comes at a price; Cycle-GAN based optimization has two separate transformers that must be learned simultaneously, and optimizing them requires four image generations per iteration. Yet fully bi-directional reconstruction is not necessary for detection adaptation, i.e., target-to-source transformations are not used at inference time in our final method.

We therefore propose a novel cycle-free approach to learning the pixel transform in our method. A transform cannot be learned reliably from unlabeled data without cycles, so we leverage available additional side labels on the source domain, specifically semantic-segmentation labels, to learn the transformation. Such annotations are easy to generate for synthetic data. We add a novel semantic pixel prediction loss to form our *SPLAT-lite* model, as described below, which provides constraints sufficient to efficiently learn effective pixel transforms for detection adaptation.

Below we show results of our general *SPLAT* approach and our fast *SPLAT-lite* method on a benchmark challenge for object detection adaptation problems, adapting from Sim 10K to Cityscapes. Our results confirm both the superiority of a pixel-transform approach for detection adaptation, and the greater efficiency of our cycle-free *SPLAT-lite* model for learning transforms for adaptation.

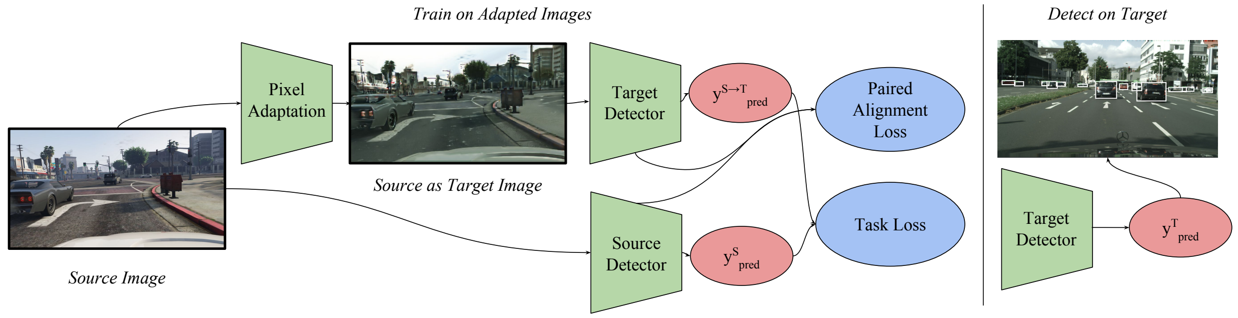


Figure 6.2: An overview of SPLAT, our proposed domain adaption technique for object detection. Our method leverages fast pixel adaptation when semantic segmentation labels are available in the source, and can also use slower, cycle-based adaptation techniques when they are not. We pair the adapted images with their corresponding source version to train a target detector by aligning the features of the source and target models, and/or training a task loss with labels inferred from source data. On the far right, we show some detection results on the target dataset, Cityscapes.

6.1 Related work

Domain adaptation was first investigated as a task for visual recognition by [Sae+10] to better understand and address the dramatic loss of performance on out-of-domain data. Visual differences between domains were shown to lead to degraded performance when models trained in one domain were naïvely applied to other domains due to so-called domain shift or dataset bias [TE11].

Early work focused on closing this gap for classification tasks at the object- or pixel-level (i.e. the tasks of classification and semantic segmentation), where each class is well represented in the set of labels. The success of these techniques have relied on two ideas, used in isolation or in concert: enforcing similarity of features deep within convolutional networks or learning pixel-level transformations from the source to the target that can be used as additional data during training.

Alignment approaches incorporate measures of distance between source and target feature distributions as a part of the task loss. The first of these methods looked at the differences between mean or variance as distribution distance metrics [LW15; SS16]. Other more recent approaches approximate distribution differences by learning a domain discriminator that classifies which features are from the source and which features are from the target [GL15; Tze+15; Tze+17]. Once the distance between domains can be measured in this way, a deep representation can be adversarially learned to directly minimize the effects of domain shift.

We can draw parallels between these adversarial adaptation approaches and generative adversarial networks (GANs) [Goo+14]; much like how adversarial adaptation techniques learn a representation that fools an adversarial domain discriminator, GANs optimize a generator to output images that fool an adversarial discriminator attempting to distinguish real from generated images. A variety of effort has been devoted to various techniques for stabilizing GANs, including but not limited modifications to the training protocol [Sal+16], alternatives to the minimax loss function [ACB17; Gul+17; Mao+17], and normalization of network parameters [Miy+18], to name a few. As they improved, GANs also began seeing widespread success in a variety of applications, including image generation [DCF+15; RMC16; ZML17], image editing [Zhu+16], and even unsupervised or semi-supervised representation learning for standard computer vision tasks [Sal+16; DKD17; Dum+17].

Another class of adaptation methods attempt to perform adaptation directly using the pixels of an image, rather than relying exclusively on intermediate feature spaces. Deep Reconstruction-Classification Networks [Ghi+16] adapt by learning a shared encoding that is trained to label source imagery but directly reconstruct target images. CoGANs [LT16] learn a partially-shared pair of GANs that output aligned source/target pairs of images when provided with a single output, and apply this model to domain adaptation.

The last group of adaptation approaches we discuss are methods that learn a transformation to redraw source images in a style that matches the target distribution. [Shr+17] learns a GAN-based model to refine synthetic images into more realistic training data such that the L1 distance to the original synthetic image is minimized. [Bou+17] also attempts to redraw source images into target images while learning a task network that can successfully label both source and source-as-target images. Finally, CyCADA [Hof+18] extends CycleGAN to domain adaptation by augmenting it with label information and performing an additional round of feature adaptation to transform source images while maintaining their semantic content. This class of approaches is most similar to our method, and so we recap relevant details here.

6.2 Background

Pixel adaptation approaches model the target distribution from the source dataset to take advantage of source labels. Many recent methods apply adversarial adaptation losses directly to the pixels of images, effectively redrawing source images in the style of the target domain. The most basic way of performing this pixel-level adaptation is to simply train a GAN such that the generator G takes source images X_s as input and attempts to match its output to the distribution of target images X_t:

$$\mathcal{L}_{\text{GAN}}(G, D, X_s, X_t) = \mathbb{E}_{x_t \sim X_t}[\log D(x_t)] + \mathbb{E}_{x_s \sim X_s}[\log(1 - D(G(x_s)))]. \tag{6.1}$$

Previous methods achieving strong adaptation performance relied on very computationally expensive content preservation losses. CycleGAN [Zhu+17] uses a content preservation loss relying on a reverse-direction generator G' and discriminator D' that adapt target samples into source samples, trained adversarially in a similar manner to G and D. The generators are constrained to be inverses of each other via a cycle-consistency loss:

$$\mathcal{L}_{\text{cycle}}(G, G', X_s, X_t) = \mathbb{E}_{x_s \sim X_s}||G'(G(x_s)) - x_s||_1 + \mathbb{E}_{x_t \sim X_t}||G(G'(x_t)) - x_t||_1. \tag{6.2}$$

Because CycleGAN was not originally proposed as a domain adaptation method, it was not designed to make use of the task labels Y_s. The authors of CyCADA [Hof+18] extend CycleGAN to an adaptation setting by proposing an additional semantic alignment loss. This second component introduces a source task network f_s that is trained on the labeled source data (X_s, Y_s). This task network is used to regularize the generator G so as to minimize the task loss on the adapted data $\mathcal{L}_T(T(G(x_s)), x_y)$. The semantic alignment loss takes the form

$$\mathcal{L}_{\text{SA}}(G, G', f_s, X_s, X_t) = \mathcal{L}_{\text{task}}(f_s, G(X_s), f_s(X_s)) + \mathcal{L}_{\text{task}}(f_s, G'(X_t), f_s(X_t)) \tag{6.3}$$

where $\mathcal{L}_{\text{task}}(f, X, Y)$ denotes the task loss (typically cross-entropy) of the task network f on samples X against the labels Y.

By adapting source images to the target domain, we can effectively apply supervisory information from the source domain to training examples that closely resemble the target unlabeled data, thereby leading to improved target performance. Generating images $X_{s\to t} = \{G(x_s) \mid x_s \in X_s\}$ that resemble the target images X_t in appearance but contain the content from the source images X_s allows us to form a new, pseudo-labeled dataset $(X_{s\to t}, Y_s)$. When the cross-domain generator G successfully accomplishes this goal, training a task model on this pseudo-labeled dataset can lead to strong performance in the target domain.

However, previous work [Hof+18] has shown that the GAN objective in Equation 6.1 alone can often train a poorly conditioned generator. Left unconstrained, the generator produces adapted source samples $X_{s\to t}$ that can vary wildly in content from the original source samples. In turn, this means that the original labels Y_s no longer correspond to the adapted samples. Thus, training a task model on $(X_{s\to t}, Y_s)$ leads to poor task performance. In order to prevent deviation from the label information Y_s, it is imperative to constrain G with an additional loss, such that G is encouraged to preserve the semantic content in X_s during pixel-level adaptation.

6.3 Semantic Pixel-Level Adaptation Transforms

We propose a method called Semantic Pixel-Level Adaptation Transforms (SPLAT) that performs pixel transformations from source to target to tackle the object detection adaptation problem. SPLAT is flexible and can optionally make use of additional label information such as segmentation labels when such data is available. In turn, the ability to make use of this extra information leads to a more efficient cycle-free pixel adaptation method that both runs faster than existing approaches and allows us to learn detection models that are more accurate in the target domain.

Existing pixel adaptation methods have demonstrated that it is possible to directly transform source images into target images in order to learn task models that are effective in the target domain. However, their applications were limited primarily to the realm of classification and segmentation. We show for the first time that these methods can be applied to detection domain adaptation, producing a cross-domain aligned set of images. We demonstrate that pixel-adaptation methods can be used to learn target domain object detectors that are robust to the negative effects of domain shift.

First, we learn a source-to-target generator G (e.g., via cycle constraints as in [Hof+18]), and use this generator on the source images X_s to produce adapted source images $X_{s\to t} = G(X_s)$. This provides us a pair-aligned set of images, X_s and $X_{s\to t}$, such that each image x_s in X_s has a counterpart $x_{s\to t} = G(x_s)$ containing the same content in a different style in $X_{s\to t}$.

6.3.1 Pseudo-labeling and pair alignment

We explore two methods for using these aligned datasets to learn a target task network: pseudo-labeling and explicit pair alignment. These methods can be used either independently or together at the same time.

The simplest way of using these two image sets together is to use the adapted images $X_{s\to t}$ along with the original source labels Y_s in order to form a surrogate, “pseudo-labeled” dataset that closely resembles the true target data. We can then learn a target task network f_t that minimizes

$$\mathcal{L}_{\text{pseudo}}(f_t, X_{s\to t}, Y_s) = \mathcal{L}_{\text{task}}(f_t, X_{s\to t}, Y_s) \tag{6.4}$$

on this newly formed dataset, where once again $\mathcal{L}_{\text{task}}$ corresponds to the task loss—in this case, a combination of cross-entropy for box classification and a regression loss for refining predictions.

We also propose a method to optionally pseudo-label an unlabeled source dataset $X_{s'}$ when such data is available. We utilize a source network f_s that is trained on the available labeled source dataset (X_s, Y_s). Since labeled source data is plentiful, it is reasonable to expect that inferred labels $\hat{Y}_{s'} = f_s(X_{s'})$ will be fairly accurate. Thus, even despite the lack of true ground truth data, we can still form a pseudo-labeled target dataset $(G(X_{s'}), \hat{Y}_{s'})$. We can then use this dataset to learn a target network f_t. This process as described attempts to use an unlabeled source dataset $X_{s'}$ to minimize the loss

$$\mathcal{L}_{\text{unsup}}(f_s, f_t, G, X_{s'}) = \mathcal{L}_{\text{task}}(f_t, G(X_{s'}), f_s(X_{s'})). \tag{6.5}$$

This enables our model to use as much source data as is available, even if that source data is unlabeled or labeled for another task such as segmentation. This process is depicted in Figure 6.2.

We also explore the possibility of using corresponding pairs of images from X_s and $X_{s\to t}$ to constrain our target task network f_t with an additional /emphpair alignment loss:

$$\mathcal{L}_{\text{pair}}(\phi_s, \phi_t, G, X_s) = \mathbb{E}_{x_s\sim X_s}||\phi_s(x_s) - \phi_t(G(x_s))||_2^2. \tag{6.6}$$

This loss enforces that the target network’s intermediate features ϕ_t should closely resemble the intermediate features ϕ_s of the original source network f_s when applied to paired images.

These two methods of learning from adapted images $X_{s\to t}$ are complementary, and can be used independently or in concert. In Section 6.4.3 we evaluate the effect each of these loss functions has on our final model performance.

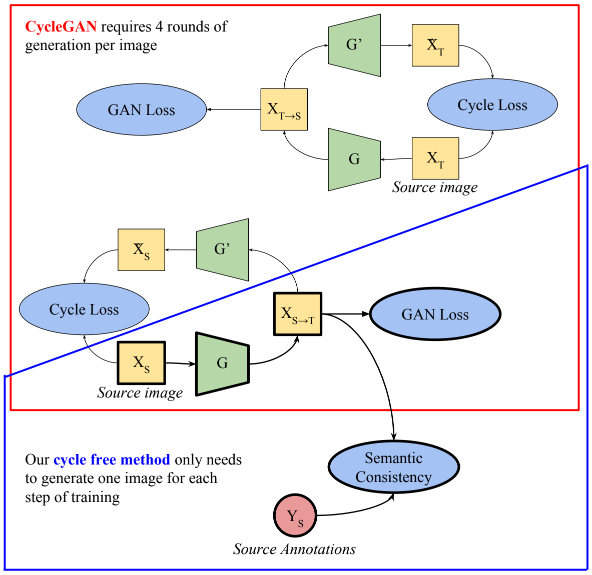


Figure 6.3: A visual comparison between our proposed cycle-free pixel adaptation method (used in *SPLAT-lite*) and CycleGAN. By incorporating source annotations while learning our pixel transformer, we can eliminate many of the components present in previous methods. In particular, we note that our model requires one-fourth as many image generation passes as CycleGAN does per iteration. In practice, using this label information leads to greatly improved efficiency and stronger performance on the final task.

6.3.2 Cycle-free pixel adaptation

Our detection adaptation method is agnostic to the specific pixel transformation model that is used. In practice, performing detection with existing pixel-level adaptation methods such as CyCADA yields surprisingly strong performance. However, these models can be quite difficult to train and do not make use of the semantic labels present in adaptation tasks. Thus, as an additional contribution, we introduce a novel, lightweight pixel-level adaptation method that eliminates the cumbersome cycle loss in favor of a loss that incorporates label information.

Existing pixel-adaptation methods such as CycleGAN/CyCADA produce impressive transformed images; however, training such models is often computationally quite expensive. For example, as explained in Section 6.2, in order to compute the cycle loss used in CycleGAN, each iteration we must perform four image generation passes: $G(x_s)$, $G'(x_t)$, $G'(G(x_s))$, and $G(G'(x_t))$, as well as an additional discriminator update to train G'. To prevent content from changing during pixel transformation, CyCADA uses a semantic consistency loss that requires two additional task network passes, $T(x_s)$ and $T(x_t)$. Running this many networks each iteration quickly becomes prohibitively expensive. Indeed, the authors of CyCADA [Hof+18] note that they were forced to drop the semantic consistency $\mathcal{L}_{\text{SA}}$ due to memory constraints.

We argue that such cumbersome models are overkill for adaptation, especially when pixel-level semantic segmentation labels are available for the source domain. As an alternative, we propose a much more lightweight method for pixel-level adaptation. By directly incorporating available pixel-level semantic labels Y_s, we can devise a novel alternative loss that effectively constrains the source-to-target generator G without requiring the use of a reverse-direction generator G'. In particular, we propose constraining G such that the source task network T is still able to predict the ground-truth labels Y_s on the adapted images $X_{s\to t}$:

$$\mathcal{L}_{\text{label}}(G, T, X_s, Y_s) = \mathcal{L}_{\text{task}}(T, G(X_s), Y_s). \tag{6.7}$$

Our lightweight model eliminates the computationally expensive cycle loss, and instead combines the GAN loss for image appearance alignment with the novel label preservation loss. This enables content preservation during domain translation and leads to efficient pixel-level domain adaptation. The resulting pixel-adapted imagery can then be used to form a new training set $(X_{s\to t}, Y_s)$ and train a task network via either simple pseudo-labeling or pair alignment, as described above. We call this version of our model *SPLAT-lite*. In full, our cycle-free pixel adaptation method optimizes the loss function

$$\mathcal{L}_{\text{SPLAT}}(G, T, X_s, X_t, Y_s) = \mathcal{L}_{\text{GAN}}(G, D, X_s, X_t) + \mathcal{L}_{\text{label}}(G, T, X_s, Y_x). \tag{6.8}$$

Figure 6.3 visually depicts our cycle-free method alongside a comparison to CycleGAN. It is visually apparent how much simpler our model is compared to previous work. In practice, this simplified training scheme leads to faster training times, since less computation is required for each training iteration. In addition, since our proposed method only uses one adversarial discriminator, optimization is simpler and proceeds in a more stable manner.

6.4 Experiments

We focus our evaluation of our proposed method on a popular and challenging synthetic-to-real car detection challenge. We begin with an overview of our experimental setup and architecture design in Sections 6.4.1 and 6.4.2. Next, we report our results on detection adaptation, comparing it with previous adaptation approaches and discussing how well feature- and image-level adaptation techniques generalize to detection in Section 6.4.3. To better understand the effect of our cycle-free pixel transformer, we then ablate SPLAT with different pixel adaptation techniques, and evaluate the relative speed and accuracy of these approaches in Section 6.4.4.

6.4.1 Datasets

We test detection adaptation performance from synthetic to real driving scenarios. Our source domain consists of Grand Theft Auto V imagery and our target domain is real-world driving data captured from dashboard mounted cameras. Consistent with prior work, we train our source detection model on *Sim10k* [JR+17], a synthetic detection dataset generated from

RGB image $x \in \mathbb{R}^{256\times256\times3}$
ResBlock down 64
ResBlock down 32
ResBlock up 16
ResBlock up 8
BN, ReLU, 3×3 conv 3, Tanh

(a) Cycle-free generator

RGB image $x \in \mathbb{R}^{256\times256\times3}$
ResBlock down 8
ResBlock down 16
ResBlock down 32
ResBlock down 64
ResBlock down 128
ResBlock down 128
ReLU
Global sum pooling
Dense → 1

(b) Cycle-free discriminator

Table 6.1: The cycle-free generator and discriminator architectures used in our SPLAT-lite experiments. We model our architectures after SN-GAN, but use spectral normalization in both the generator and the discriminator. Architectures are described using the same notation as [Miy+18]; see the original paper for details.

Grand Theft Auto V, and test on *Cityscapes* [Cor+16]. *Sim10k* contains 10,000 images and 58,701 bounding boxes, all of which are used to train our adaptation model. There are no bounding box annotations *Cityscapes*, so we construct tight bounding boxes around instance-level segmentations. We treat the 2975 training images as our target dataset and evaluate on the 500 validation images. We only test detection results for car.

Our model is amenable to training with additional source data that is either unlabeled or has labels for a different task. In our experiments, we take advantage of an additional in-domain source dataset with segmentation labels to improve the speed of our pixel-adaptation model. We use a distinct set of images generated from Grand Theft Auto V and annotated with semantic segmentation labels. This is the *GTA5* dataset [Ric+16]. It contains 24966 images that we use to train the pixel adaptation component of SPLAT-Lite. In our pixel adaptation analysis, we describe the training details of SPLAT-lite and show that, by compromising training speed, we can achieve comparable accuracy without additional source data.

6.4.2 Implementation details

We use Faster R-CNN [Ren+15] as our detection module with the hyperparameters suggested in the authors' implementation [Gir+18]. We scale each image to a height of 512 pixels and a maximum width of 1024 pixels to fit in GPU memory. Our backbone architecture is the ResNet-based feature pyramid network [Lin+17], and we initialize from weights pretrained on ImageNet [Rus+15]. We report adaptation results for our model using a cycle-free pixel transformation model, and opt not to include a pair alignment loss.

Method	**mAP @ 0.5**
Source only	31.1
Domain Adaptive Faster R-CNN [Che+18]	39.0
SPLAT-lite (ours)	**51.5**
Oracle	70.7

Table 6.2: We report mean average precision at a threshold of 0.5 IoU on the Cityscapes validation set. Our pixel-level adaptation model outperforms previous approaches, which limited themselves to feature-level adaptation, by a significant margin. In fact, we are able to recover over 50% of the difference between the simple source-only baseline and the upper bound target oracle result.

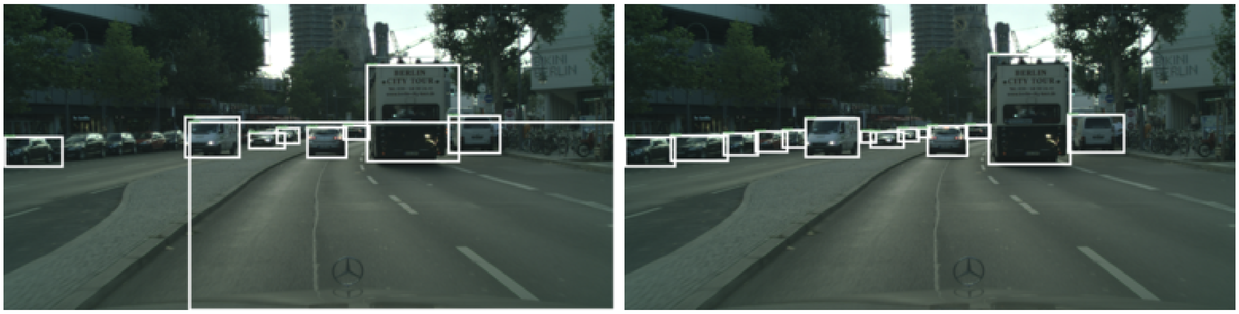
Our cycle-free pixel adaptation transformer in our primary results is trained on the *GTA5* dataset. The generator and discriminator are variants of architectures used by SN-GAN [Miy+18], but modified to perform image-to-image generation—the exact architectures we use are outlined in Table 6.1. The task network used in the label preservation loss is an FCN-8s [LSD15] fully convolutional network, again trained on the *GTA5* dataset. In Section 6.4.4, we also experiment with variants of the SPLAT architecture and compare it with previous pixel-adaptation methods to validate our claims of improved performance and efficiency.

Our training procedure is as follows. We begin by training a source Faster R-CNN model on *Sim10k*. That model is used to generate detections on the *GTA5* dataset. Next, we accept all proposals with a confidence of 0.5 or greater and treat them as ground truth annotations for *GTA5*. We transform *GTA5* into the target *Cityscapes* style using our learned pixel transformer. Finally, we train Faster R-CNN using the transformed images and the detections inferred by the source detectors on the original *GTA5* images.

6.4.3 Detection adaptation results

We separately investigate the effects of the pseudo-label and aligned pair losses. Our results show that, for the detection adapation task, training on the pseudo-labeled dataset of source adapted images strictly outperforms training with a feature loss on aligned pairs. We first examine these losses independently, then explore combining both. All of our experiments are evaluated the validation set of *Cityscapes*.

When using these losses independently, we find that the pseudo-label loss outperforms the aligned pair loss by a significant margin: pseudo-labeling achieves 51.5 mAP versus the aligned pair loss, which achieves 43.3. Somewhat surprisingly, combining the two losses appears to underperform the pseudo-label loss alone, achieving a mAP of only 47.5. We hypothesize that directly matching the features between source and target is too restrictive and prevents the model from properly learning features that work well in the target domain, and report



(a) Before adaptation

(b) After adaptation

Figure 6.4: Comparing detections output by our model both before and after adaptation indicates that our model recognizes cars out of context and does not misattribute parts of cars, such as the hood ornament, for whole cars.

final results using only the pseudo-label loss.

Our final results are presented in Table 6.2. Our proposed method achieves state-of-the-art performance, outperforming competing baselines by 12.5 points. We also provide a comparison against a model trained using the fully-labeled Cityscapes training set, which we call the oracle model. This serves as an upper bound for what level of performance we can reasonably expect to achieve via adaptation. Comparing our SPLAT result against this oracle indicates that our model recovers over 50% of the missing performance due to domain shift.

A qualitative analysis of detections both pre- and post-adaptation yields additional insight into our adaptation method. We visualize detections produced by both the source-only baseline and our model on a sample image from the Cityscapes validation dataset in Figure 6.4. Before adaptation, the source-only model is decent at detecting cars that are driving in the center of the road. However, it misses many of the cars that are parked along the side of the road, and it appears to be distracted by the presence of the car that the camera is mounted on along the bottom of the image. Our adaptation method appears to fix both of these issues: almost every car parked on the side is detected, and no spurious detections are present.

6.4.4 Pixel adaptation methods

The results presented in Section 6.4.3 were for a particular instantiation of SPLAT using our proposed cycle-free pixel adaptation method. We refer to this instantiation of SPLAT as SPLAT-lite. In this section, we explore the effect that our choice of pixel adaptation method has on our final detection performance. We compare SPLAT-lite against a version of SPLAT trained using a CyCADA pixel transformer, which we refer to as SPLAT-cycle. In doing so, we show that SPLAT is flexible enough to effectively use different pixel-adaptation methods while simultaneously validating our new cycle-free pixel transformation approach against these methods. To demonstrate the efficiency of SPLAT-lite, we benchmark the time it takes to perform a training iteration on a single image (both forward and backward pass). These

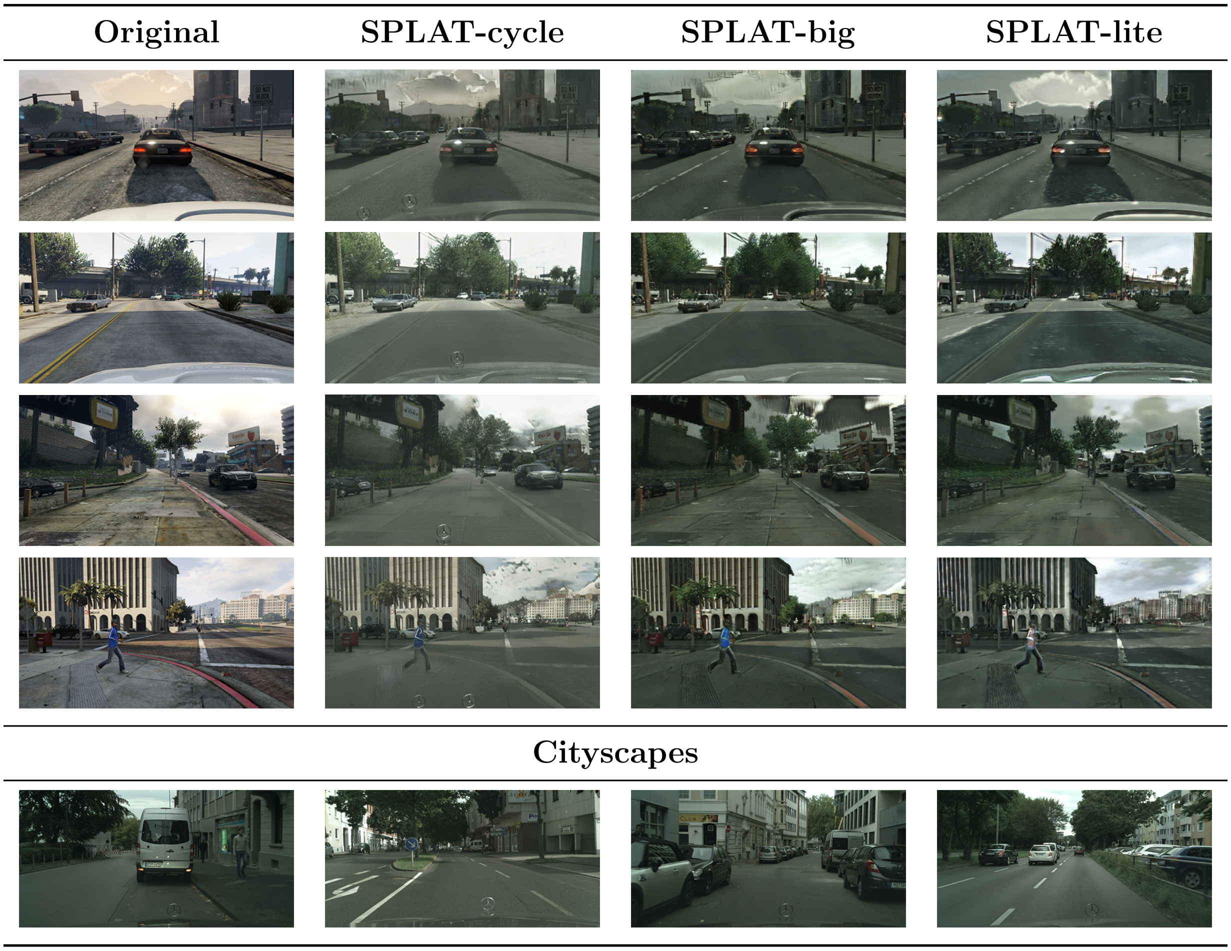


Table 6.3: Side-by-side comparisons of pixel adaptation on source imagery. SPLAT-lite is able to generate effective target-style imagery while training almost 4 times faster than CyCADA. The target Cityscapes images are provided as examples of real target images for comparison, and do not directly correspond to the source images.

timings use 256×256 RGB images as input and were produced on a Tesla P100 GPU.

Additionally, we experiment with a larger, overparametrized variant of our cycle-free model that uses the same architectures as in Table 6.1, but with 8 times as many channels in each of its layers in both the generator and the discriminator. We refer to this variant as SPLAT-big, and include it as a third version of SPLAT in our evaluation.

Results from this full comparison are shown in Table 6.4. We see that, in addition to achieving the strongest detection adaptation results in our framework, SPLAT-lite runs almost four times faster than SPLAT-cycle. Despite how much more lightweight SPLAT-lite is, leveraging task information during training enables it to produce target-style images that are better suited for domain adaptation.

A comparison of SPLAT-lite against SPLAT-big is also illuminating. Considering that

Method	Cityscapes mAP @ 0.5	Time per training iteration (s)	Speedup over SPLAT-cycle
SPLAT-cycle	48.1	0.377	×1.00
SPLAT-big	48.4	0.213	×1.77
SPLAT-lite	**51.5**	**0.098**	×**3.84**

Table 6.4: We compare the effects of using different pixel adaptation methods on both final adaptation performance and speed during training. Training benchmarks are produced on a Tesla P100 with a 256×256 image as input. Our proposed cycle-free pixel adaptation method SPLAT-lite has clear advantages over previous methods, both in the quality of the final model as well as the speed of the model during training.

SPLAT-lite is already able to produce effective adapted imagery, it is reasonable to suspect that SPLAT-big may be overparametrized. In turn, an overparametrized generator is more likely to destroy the content of the original image during cross-domain image generation, since the increased number of parameters gives it more freedom to produce different outputs. Nevertheless, we see that, although performance is worse than SPLAT-lite, the combination of the label alignment loss along with modern GAN stabilization techniques such as spectral normalization ensures that optimization proceeds in a stable manner, and the final result is competitive with cycle-based methods.

Examples of pixel-adapted source imagery are shown in Table 6.3. By removing cycle-consistency constraints, SPLAT-big and SPLAT-lite are able to avoid learning domain artifacts that are irrelevant to the task at hand. In contrast, with cycle-consistency enforced, the pixel-adaptation model hallucinates the hood ornament that is common in the Cityscapes dataset. Notice, in column 2 and row 5, the hood ornament is even generated in unreasonable contexts, such as on the sidewalk. The final instantiation of our model, SPLAT-lite, also generates less noise than either of its highly-parameterized counterparts.

6.5 Conclusion

Our proposed method SPLAT is a novel approach to detector adaptation that utilizes pixel-level transforms to adapt from source to target domains. SPLAT is a flexible model; it works on unlabeled target data, and when target labels are present, can additionally condition on them with a cycle-free loss to operate more accurately and efficiently. Our model is also able to make use of unlabeled source data by inferring additional label information, thereby increasing the amount of training data available to learned task networks. By incorporating these improvements, our final model adapts images 3.8 times faster than previous cycle-based methods while improving 12.5 mAP over previous methods on a challenging detection adaptation setting.

Chapter 7

Summary and Future Directions

Despite its seemingly narrow definition, domain adaptation is a problem that arises constantly in any machine learning setting. These adaptation problems can manifest in many different forms, whether it be from one dataset to another, synthetic to real imagery, or curated datasets to imagery in the wild. Under the right lens, one can even view train set overfitting as a form of domain shift!

In this thesis, we demonstrated the many ways in which convolutional neural networks can be used as a powerful tool for faciliating domain adaptation, thereby enabling models to generalize beyond the dataset they were trained on. We introduced the concept of a domain alignment loss, which attempts to measure the difference between two domains, and showed that end-to-end learnable representations can be optimized directly against these losses, resulting in representations that are invariant to these differences. These losses can take many forms, ranging from simple first-order statistical moments to ones learned adversarially by a domain discriminator. They can also be used at many different levels—at the uppermost layers of a pretrained convolutional network to align high-level features, or to learn models that directly modify the pixels of images to transform them into cross-domain counterparts. Finally, we demonstrated that these methods of learning domain-invariant representations can be applied to a variety of vision tasks including classification, detection, and segmentation, and can be even used to transfer information between related tasks.

Of course, the methods outlined in this thesis are not the be-all and end-all of domain adaptation, and barely scratch the surface of viable adaptation techniques. We conclude by briefly outlining three logical extensions of the work in this thesis, which we hope can serve as promising directions for future research in the field of visual domain adaptation.

Structural Domain Adaptation In Chapters 5 and 6 we demonstrated adaptation techniques that work well on scene-based computer vision tasks such as semantic segmentation and object detection. These methods rely on the use of image-to-image generators to transform images from one domain to match the appearance of the other domain. Due to the convolutional structure of these generators, they are especially effective at local appearance and textural transformations, such as altering the color balance of an image, or changing the

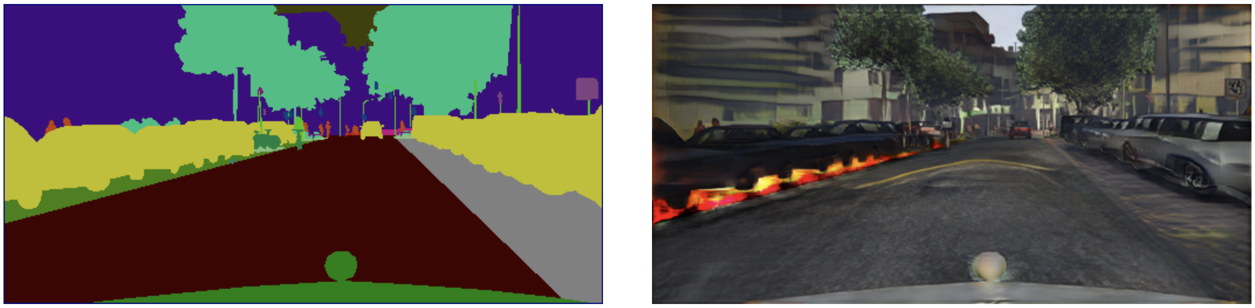


Figure 7.1: When we condition a label-to-image generator pretrained on GTA5 imagery on a Cityscapes label (left), the generated image (right) exhibits extreme artifacting in the presence of unseen label configurations. This indicates the presence of a domain shift not just in the appearance of the scene, but in the actual *scene structure* as well—in this case, we can see that GTA5 does not have many examples of cars parallel parked in a row.

texture of objects to better suit a domain. However, in practice the cause of domain shift is a combination of both local and global factors.

We demonstrate how local, appearance-level differences cannot fully account for domain shifts in a simple experiment. We train a SPADE-based label-to-image generator [Par+19] on the GTA dataset [Ric+16], so that it learns to effectively generate full driving imagery by conditioning on semantic segmentation labels. Next, rather than conditioning on GTA labels, we try conditioning this model on Cityscapes [Cor+16] labels instead. In theory, there should be no appearance-level difference, since labels from the two datasets come from the same label set and thus can be encoded identically. Nevertheless, as shown in Figure 7.1, for certain input labels the generator fails, generating extreme artifacts in the output image. We hypothesize that this failure arises because the generator has never seen this particular configuration of labels before—the GTA dataset tends to have fewer cars per scene than Cityscapes, and they are typically spaced further apart as well.

However, these same label-to-image generators that we use to demonstrate the structural domain shift also give us a potential tool we can use to try and address this issue. By allowing adaptation networks to directly process label maps, which explicitly represent the layout of a scene, we can make it easier for them to change the global composition of an image. One can imagine a "label modification" network that learns how to alter label maps in the source domain so that they better align with those from the target domain. These altered label maps could then be used to generate a synthetic target dataset, complete with modified labels that more closely mimic the layout of target scenes. This would form the backbone of a two-pronged approach to adaptation on structured scenes: first, address the global structural differences between the two domains, and after that shift has been mitigated, we can then address local appearance-level shifts using existing methods such as the one outlined in Chapter 5.

Stronger Safeguards and Diagnostics We have confidence that domain adaptation works because these techniques are evaluated on benchmark datasets. By testing methods against labeled test sets, we obtain quantitative metrics that allow us to directly compare the effectiveness of adaptation methods across a variety of different tasks and settings.

However, in truly unsupervised adaptation settings, there is no labeled test set for us to evaluate on, and thus short of manual inspection of inputs and outputs there is no way for us to confirm that adaptation has actually succeeded. This problem is especially exacerbated by the rise of adversarial adaptation techniques, which suffer from the same pitfalls that generative adversarial networks do. These methods, while effective, can be temperamental to train stably and are prone to collapsing entirely.

Thus, for adaptation to be truly useful in real-world settings, it is important that we have safeguards and diagnostics that can indicate when adaptation is failing or behaving unexpectedly. For example, methods that rely on cross-domain generation typically output intermediate images that can be inspected to verify that they resemble the target domain while still maintaining coherency. Figure 7.1, as previously discussed, is an example of a failure mode that manifests clearly in these intermediate visualizations.

Even better would be methods that could automatically identify when adaptation is failing without requiring manual user intervention. Existing work on network calibration and out-of-distribution detection provides guidance some ways such methods could work [Guo+17; Lee+18]. If we are able to properly calibrate our networks, then they should be able to identify when their predictions are likely to be wrong. Low confidence predictions could then serve as a potential way for models to indicate that adaptation has failed. Out-of-distribution detection provides a similar promise, where samples that the network is ill-suited to operate on could be automatically detected and flagged. This capability would be especially vital on models deployed in the real-world, as a way to identify settings in which further adaptation is required.

Non-adversarial Alignment Objectives Finally, although Chapters 3 through 6 focused primarily on adversarial objectives for aligning domains, we need not limit ourselves to this family of objectives. Non-adversarial alternatives to GANs is an emerging field of research, with recent methods beginning to approach the quality of adversarial methods [HLM19; Aic+20]. Although these methods are designed with image generation in mind, recall that in Chapter 4 we showed that adversarial adaptation can be viewed as a specific instantiation of GANs where the distribution being generated is the source feature distribution, rather than images. A similar analogy holds for many of these GAN alternatives, and these non-adversarial alternatives could enable us to sidestep the stability and collapse issues that plague adversarial methods while still maintaining comparable task performance.

As a simple, preliminary experiment, we try applying an analogous adaptation version of IMLE [LM18] to two domain shifts in the Office-Home dataset [Ven+17]. At a high level, this method trains a target network to mimic a fixed, pre-trained source network by matching up each source feature with its nearest neighbor target feature. The target network is then

Method	**Art→Clipart**	**Product→Clipart**
Source only	34.9	31.2
DANN [Gan+16]	43.6	43.6
DA-IMLE	51.9	52.1

Table 7.1: Evaluation of a simple domain adaptation variant of IMLE [LM18] on the Office-Home [Ven+17] dataset. Comparison to the adversarial baseline indicates that this holds promise as a competitive, non-adversarial alternative to modern adaptation methods.

optimized so that the distance between each source-target feature pair is minimized. While the results are not state-of-the-art, we see in Table 7.1 that this simple procedure outperforms basic adversarial adaptation by a sizable margin. Repeated runs of this experiment also indicate that this method is remarkably stable, with run-to-run fluctuations varying by less than a percentage point.

Curiously, this matching procedure seems to select out a subset of the target images comprising roughly half of the full target domain—the remaining half is never selected as a nearest neighbor, and thus is never used during optimization. We hypothesize that this essentially serves as a weak form of outlier rejection, where target images that cannot be reasonably related to source images are excluded entirely rather than being optimized against a match that is probably spurious.

Bibliography

[AB10] Lorenzo Torresani Alessandro Bergamo. “Exploiting weakly-labeled Web images to improve object classification: a domain adaptation approach”. In: *Neural Information Processing Systems (NIPS)*. 2010.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein Generative Adversarial Networks”. In: *International Conference on Machine Learning (ICML)*. 2017.

[Aic+20] Abhishek Aich et al. “Non-Adversarial Video Synthesis with Learned Priors”. In: *arXiv preprint arXiv:2003.09565* (2020).

[AZ11] Y. Aytar and A. Zisserman. “Tabula Rasa: Model Transfer for Object Category Detection”. In: *International Conference on Computer Vision (ICCV)*. 2011.

[BC14] Jimmy Ba and Rich Caruana. “Do Deep Nets Really Need to be Deep?” In: *Neural Information Processing Systems (NIPS)*. 2014.

[BDFF12] A. Berg, J. Deng, and L. Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge 2012”. In: (2012). URL: http://www.image-net.org/challenges/LSVRC/2012/.

[Bor+06] Karsten M. Borgwardt et al. “Integrating structured biological data by Kernel Maximum Mean Discrepancy”. In: *Bioinformatics*. 2006.

[Bou+16] Konstantinos Bousmalis et al. “Domain Separation Networks”. In: *Neural Information Processing Systems (NIPS)*. 2016, pp. 343–351.

[Bou+17] Konstantinos Bousmalis et al. “Unsupervised Pixel-Level Domain Adaptation With Generative Adversarial Networks”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Bou+18] Konstantinos Bousmalis et al. “Using simulation and domain adaptation to improve efficiency of deep robotic grasping”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018.

[Bro+93] Jane Bromley et al. “Signature verification using a “Siamese” time delay neural network”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 7.04 (1993).

[CBG13] S. Chopra, S. Balakrishnan, and R. Gopalan. “DLID: Deep Learning for Domain Adaptation by Interpolating between Domains”. In: *ICML Workshop on Challenges in Representation Learning*. 2013.

[Che+17] Yi-Hsin Chen et al. “No More Discrimination: Cross City Adaptation of Road Scene Segmenters”. In: *International Conference on Computer Vision (ICCV)*. 2017.

[Che+18] Yuhua Chen et al. “Domain Adaptive Faster R-CNN for Object Detection in the Wild”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2018.

[CHL05] Sumit Chopra, Raia Hadsell, and Yann LeCun. “Learning a similarity metric discriminatively, with application to face verification”. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005.

[Cor+16] Marius Cordts et al. “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2016.

[DCF+15] Emily L Denton, Soumith Chintala, Rob Fergus, et al. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”. In: *Neural Information Processing Systems (NIPS)*. 2015.

[DKD17] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. “Adversarial Feature Learning”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[Don+13] Jeff Donahue et al. “Semi-supervised Domain Adaptation with Instance Constraints”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2013.

[Don+14] Jeff Donahue et al. “DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition.” In: *International Conference on Machine Learning (ICML)*. 2014.

[Dum+17] Vincent Dumoulin et al. “Adversarially learned inference”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[DXT12] L. Duan, D. Xu, and Ivor W. Tsang. “Learning with Augmented Features for Heterogeneous Domain Adaptation”. In: *International Conference on Machine Learning (ICML)*. 2012.

[Fer+13] B. Fernando et al. “Unsupervised Visual Domain Adaptation Using Subspace Alignment”. In: *Proc. ICCV*. 2013.

[Gan+16] Yaroslav Ganin et al. “Domain-adversarial training of neural networks”. In: *Journal of Machine Learning Research* 17.59 (2016), pp. 1–35.

[GEB16] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. “Image style transfer using convolutional neural networks”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Ghi+16] Muhammad Ghifary et al. “Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation”. In: *European Conference on Computer Vision (ECCV)*. Springer. 2016.

[Gir+13] R. Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *arXiv e-prints* (2013). arXiv: 1311.2524.

[Gir+18] Ross Girshick et al. *Detectron*. https://github.com/facebookresearch/detectron. 2018.

[GKZ14] Muhammad Ghifary, W. Bastiaan Kleijn, and Mengjie Zhang. “Domain Adaptive Neural Networks for Object Recognition”. In: *CoRR* abs/1409.6041 (2014). URL: http://arxiv.org/abs/1409.6041.

[GL15] Yaroslav Ganin and Victor Lempitsky. “Unsupervised Domain Adaptation by Backpropagation”. In: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. 2015.

[Gon+12] B. Gong et al. “Geodesic Flow Kernel for Unsupervised Domain Adaptation”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2012.

[Goo+14] Ian Goodfellow et al. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems 27*. 2014.

[Gre+09] A. Gretton et al. “Covariate shift and local learning by distribution matching”. In: *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2009.

[Gul+17] Ishaan Gulrajani et al. “Improved Training of Wasserstein GANs”. In: *Neural Information Processing Systems (NIPS)*. 2017.

[Guo+17] Chuan Guo et al. “On calibration of modern neural networks”. In: *International Conference on Machine Learning (ICML)*. 2017.

[Gup+14] Saurabh Gupta et al. “Learning rich features from rgb-d images for object detection and segmentation”. In: *European Conference on Computer Vision (ECCV)*. 2014, pp. 345–360.

[Hae+17] P. Haeusser et al. “Associative Domain Adaptation”. In: *International Conference on Computer Vision (ICCV)*. 2017.

[He+16] Kaiming He et al. “Deep residual learning for image recognition”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[HLM19] Yedid Hoshen, Ke Li, and Jitendra Malik. “Non-adversarial image synthesis with generative latent nearest neighbors”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2019.

[Hof+13] J. Hoffman et al. “Efficient Learning of Domain-invariant Image Representations”. In: *Proc. ICLR*. 2013.

[Hof+14a] J. Hoffman et al. “One-Shot Learning of Supervised Deep Convolutional Models”. In: *arXiv 1312.6204; presented at ICLR Workshop*. 2014.

[Hof+14b] Judy Hoffman et al. “LSDA: Large Scale Detection through Adaptation”. In: *Neural Information Processing Systems (NIPS)*. 2014.

[Hof+16] Judy Hoffman et al. “FCNs in the Wild: Pixel-level Adversarial and Constraint-based Adaptation”. In: *CoRR* abs/1612.02649 (2016). URL: http://arxiv.org/abs/1612.02649.

[Hof+18] Judy Hoffman et al. “CyCADA: Cycle Consistent Adversarial Domain Adaptation”. In: *International Conference on Machine Learning (ICML)*. 2018.

[HVD14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the Knowledge in a Neural Network”. In: *NIPS Deep Learning and Representation Learning Workshop*. 2014.

[Iso+17] Phillip Isola et al. “Image-to-image translation with conditional adversarial networks”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Jia+14] Yangqing Jia et al. “Caffe: Convolutional Architecture for Fast Feature Embedding”. In: *arXiv preprint arXiv:1408.5093* (2014).

[JR+17] M. Johnson-Roberson et al. “Driving in the Matrix: Can Virtual Worlds Replace Human-Generated Annotations for Real World Tasks?” In: *IEEE International Conference on Robotics and Automation*. 2017.

[Kar+16] Levent Karacan et al. “Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts”. In: *arXiv preprint arXiv:1612.00215* (2016).

[KB15] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: (2015).

[KBDG04] D. Kifer, S. Ben-David, and J. Gehrke. “Detecting change in data streams”. In: *Proc. VLDB*. 2004.

[Kim+17] Taeksoo Kim et al. “Learning to discover cross-domain relations with generative adversarial networks”. In: *International Conference on Machine Learning (ICML)*. 2017.

[KSD11] B. Kulis, K. Saenko, and T. Darrell. “What You Saw is Not What You Get: Domain Adaptation Using Asymmetric Kernel Transforms”. In: *Proc. CVPR*. 2011.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Proc. NIPS*. 2012.

[LBK17] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. “Unsupervised Image-to-Image Translation Networks”. In: *Neural Information Processing Systems (NIPS)*. 2017.

[LeC+98] Y. LeCun et al. “Gradient-Based Learning Applied to Document Recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[Lee+18] Kimin Lee et al. “A simple unified framework for detecting out-of-distribution samples and adversarial attacks”. In: *Neural Information Processing Systems (NIPS)*. 2018.

[LF13] Evgeny Levinkov and Mario Fritz. “Sequential Bayesian Model Update under Structured Scene Prior for Semantic Road Scenes Labeling”. In: *International Conference on Computer Vision (ICCV)*. 2013.

[Lin+17] Tsung-Yi Lin et al. “Feature Pyramid Networks for Object Detection”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[LM18] Ke Li and Jitendra Malik. “Implicit maximum likelihood estimation”. In: *arXiv preprint arXiv:1809.09087* (2018).

[Lon+13] M. Long et al. “Transfer Feature Learning with Joint Distribution Adaptation”. In: *International Conference on Computer Vision (ICCV)*. 2013, pp. 2200–2207.

[Lon+15] Mingsheng Long et al. “Learning Transferable Features with Deep Adaptation Networks”. In: *International Conference on Machine Learning (ICML)*. 2015.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2015.

[LT16] Ming-Yu Liu and Oncel Tuzel. “Coupled generative adversarial networks”. In: *Neural Information Processing Systems (NIPS)*. 2016, pp. 469–477.

[LW15] Mingsheng Long and Jianmin Wang. “Learning transferable features with deep adaptation networks”. In: *International Conference on Machine Learning (ICML)*. 2015.

[Mao+17] Xudong Mao et al. “Least Squares Generative Adversarial Networks”. In: *International Conference on Computer Vision (ICCV)*. 2017.

[Miy+18] Takeru Miyato et al. “Spectral Normalization for Generative Adversarial Networks”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[MMR09] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. “Domain Adaptation: Learning Bounds and Algorithms”. In: *COLT*. 2009.

[MO14] Mehdi Mirza and Simon Osindero. “Conditional Generative Adversarial Nets”. In: *CoRR* abs/1411.1784 (2014). URL: http://arxiv.org/abs/1411.1784.

[Net+11] Yuval Netzer et al. “Reading Digits in Natural Images with Unsupervised Feature Learning”. In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*. 2011.

[Ngi+11] Jiquan Ngiam et al. “Multimodal deep learning”. In: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. 2011.

[Pan+09] Sinno Jialin Pan et al. “Domain Adaptation via Transfer Component Analysis”. In: *IJCA*. 2009.

[Par+19] Taesung Park et al. "Semantic Image Synthesis with Spatially-Adaptive Normalization". In: *Computer Vision and Pattern Recognition (CVPR)*. 2019.

[Ren+15] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Neural Information Processing Systems (NIPS)*. 2015.

[Ric+16] Stephan R. Richter et al. "Playing for Data: Ground Truth from Computer Games". In: *European Conference on Computer Vision (ECCV)*. 2016.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *International Conference on Learning Representations (ICLR)*. 2016.

[Ros+16a] German Ros et al. "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes". In: *Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Ros+16b] Germán Ros et al. "Training Constrained Deconvolutional Networks for Road Scene Semantic Segmentation". In: *CoRR* abs/1604.01545 (2016). URL: http://arxiv.org/abs/1604.01545.

[RSF18] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. "Beyond sharing weights for deep domain adaptation". In: *Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 41.4 (2018), pp. 801–814.

[Rus+15] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[Sae+10] K. Saenko et al. "Adapting Visual Category Models to New Domains". In: *Proc. ECCV*. 2010.

[Sal+16] Tim Salimans et al. "Improved techniques for training gans". In: *Neural Information Processing Systems (NIPS)*. 2016.

[San+17] Patsorn Sangkloy et al. "Scribbler: Controlling deep image synthesis with sketch and color". In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Sen+16] Ozan Sener et al. "Learning Transferrable Representations for Unsupervised Domain Adaptation". In: *Neural Information Processing Systems (NIPS)*. 2016.

[Ser+13] P. Sermanet et al. "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks". In: *CoRR* abs/1312.6229 (2013).

[Shr+17] Ashish Shrivastava et al. "Learning From Simulated and Unsupervised Images Through Adversarial Training". In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Sil+12] Nathan Silberman et al. "Indoor Segmentation and Support Inference from RGBD Images". In: *European Conference on Computer Vision (ECCV)*. 2012.

[SS16] Baochen Sun and Kate Saenko. “Deep CORAL: Correlation Alignment for Deep Domain Adaptation”. In: *ICCV workshop on Transferring and Adapting Source Knowledge in Computer Vision (TASK-CV)*. 2016.

[SZ15] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *International Conference on Learning Representations (ICLR)*. 2015.

[TC13] T. Tommasi and B. Caputo. “Frustratingly Easy NBNN Domain Adaptation”. In: *Proc. ICCV*. 2013.

[TE11] A. Torralba and A. Efros. “Unbiased Look at Dataset Bias”. In: *Proc. CVPR*. 2011.

[TPW17] Yaniv Taigman, Adam Polyak, and Lior Wolf. “Unsupervised Cross-Domain Image Generation”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[TTC14] T. Tommasi, T. Tuytelaars, and B. Caputo. “A Testbed for Cross-Dataset Analysis”. In: *TASK-CV Workshop, ECCV*. 2014.

[Tze+14] Eric Tzeng et al. “Deep Domain Confusion: Maximizing for Domain Invariance”. In: *CoRR* abs/1412.3474 (2014). URL: http://arxiv.org/abs/1412.3474.

[Tze+15] Eric Tzeng et al. “Simultaneous Deep Transfer Across Domains and Tasks”. In: *International Conference in Computer Vision (ICCV)*. 2015.

[Tze+17] Eric Tzeng et al. “Adversarial Discriminative Domain Adaptation”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Tze+18] Eric Tzeng et al. “SPLAT: semantic pixel-level adaptation transforms for detection”. In: *arXiv preprint arXiv:1812.00929* (2018).

[Ven+17] Hemanth Venkateswara et al. “Deep Hashing Network for Unsupervised Domain Adaptation”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Yi+17] Zili Yi et al. “DualGAN: Unsupervised dual learning for image-to-image translation”. In: *International Conference on Computer Vision (ICCV)*. 2017.

[YKF17] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. “Dilated Residual Networks”. In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Yoo+16] Donggeun Yoo et al. “Pixel-Level Domain Transfer”. In: *European Conference on Computer Vision (ECCV)*. 2016.

[YYH07] J. Yang, R. Yan, and A. Hauptmann. “Adapting SVM classifiers to data with shifted distributions”. In: *ICDM Workshops*. 2007.

[ZDG17] Yang Zhang, Philip David, and Boqing Gong. “Curriculum Domain Adaptation for Semantic Segmentation of Urban Scenes”. In: *International Conference on Computer Vision (ICCV)*. 2017.

[Zhu+16] Jun-Yan Zhu et al. “Generative visual manipulation on the natural image manifold”. In: *European Conference on Computer Vision (ECCV)*. 2016.

[Zhu+17] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *International Conference on Computer Vision (ICCV)*. 2017.

[ZLK18] Junting Zhang, Chen Liang, and C-C Jay Kuo. “A fully convolutional tri-branch network (FCTN) for domain adaptation”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018.

[ZML17] Junbo Zhao, Michael Mathieu, and Yann LeCun. “Energy-based Generative Adversarial Network”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[Dau07] H. Daumé III. “Frustratingly easy domain adaptation”. In: *ACL*. 2007.