Drop to Adapt:
Learning Discriminative Features for Unsupervised Domain Adaptation

Seungmin Lee*
Seoul National Univ.

Dongwan Kim*
Seoul National Univ.

Namil Kim
NAVER LABS

Seong-Gyun Jeong
CODE42.ai

Abstract

Recent works on domain adaptation exploit adversarial training to obtain domain-invariant feature representations from the joint learning of feature extractor and domain discriminator networks. However, domain adversarial methods render suboptimal performances since they attempt to match the distributions among the domains without considering the task at hand. We propose Drop to Adapt (DTA), which leverages adversarial dropout to learn strongly discriminative features by enforcing the cluster assumption. Accordingly, we design objective functions to support robust domain adaptation. We demonstrate efficacy of the proposed method on various experiments and achieve consistent improvements in both image classification and semantic segmentation tasks. Our source code is available at `https://github.com/postBG/DTA.pytorch`.

1. Introduction

The advent of deep neural networks (DNNs) has shown exceptional performances on various visual recognition tasks using large-scale datasets [8, 21, 13]. Training a DNN model begins with curating data and its associated label. In general, the annotation process is expensive and time-consuming. Moreover, we are unable to collect appropriate data in some cases, if events are rarely encountered or related to dangerous situations. Hence, researchers [32, 34, 10, 35] are paying attention to leverage synthetic data in a simulation environment, where annotating labels is effortless to a wide range of scenarios.

To take full advantage of synthetic datasets, domain adaptation has become an active research area. In the domain adaptation setting, we leverage rich annotations on a source domain to achieve strong performance on a target domain regardless of poor annotations. Nevertheless, a model trained only on the source domain provides disappointing

---

* denotes equal contribution.
This work was done while the authors were at NAVER LABS. Correspondence to {`profile2697`, `dongwan123`}`@gmail.com`

(a) Before adaptation

(b) Adapted model

(c) AdD on feature extractor
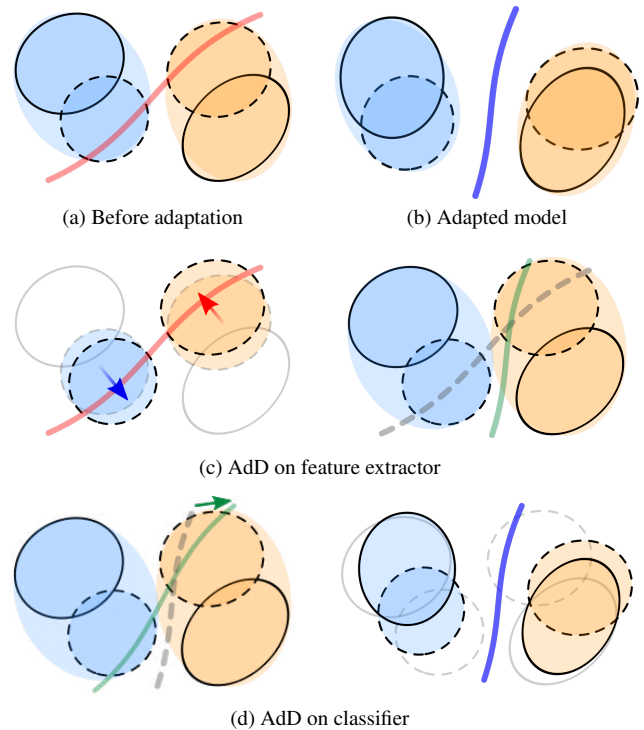
(d) AdD on classifier

Figure 1. We illustrate the domain adaptation process with adversarial dropout (AdD). We depict the source and target domains as solid and dashed lines, respectively. Decision boundary of a model only trained on the source domain easily violates the *cluster assumption* in that it passes through target feature-dense regions (a). We can apply AdD on both the feature extractor (c) and classifier (d). When AdD is used on the feature extractor, the decision boundary is pushed away from feature dense regions. On the contrary, AdD on the classifier pushes features away from the decision boundary. Eventually, our domain adapted model draws a robust decision boundary that avoids clusters (b).

outcomes when the target domain shows inherently different characteristics. This issue is known as *domain shift* and is one of the main reasons for performance drops on the target domain. Therefore, we propose a novel method that can reduce the domain shift for domain adaptation.

In this paper, we tackle unsupervised domain adaptation (UDA), where the target domain is completely unlabelled. Recent works have proposed to align source and target domain distributions through domain adversarial training [11, 44, 12]. These methods employ an auxiliary domain discriminator to obtain domain-invariant feature representation. The main assumption in domain adversarial training is that if the feature representation is domain-invariant, a classifier trained on the source domain's features will operate on the target domain as well. However, the weaknesses of domain adversarial methods have been pointed out in [37, 36, 40]. Since the domain discriminator simply aligns source and target features without considering the class labels, it is likely that the resulting features will not only be domain-invariant, but also non-discriminative with respect to class labels. Consequently, it is hard to reach the optimal performance on classification.

Our approach is based on the cluster assumption, which states that decision boundaries should be placed in low density regions in the feature space [5]. Without model adaptation, the feature extractor generates indiscriminate features for unseen data from the target domain, and the classifier may draw decision boundaries that pass through feature-dense regions on the target domain. Thus, we learn a domain adapted model by pushing the decision boundary away from the target domain's features. Our method, *Drop to Adapt* (DTA), employs adversarial dropout [30] to enforce the cluster assumption on the target domain. More precisely, to support various tasks, we introduce element-wise and channel-wise adversarial dropout operations for fully-connected and convolutional layers, respectively. Fig. 1 overviews our method, and we design the associated loss functions in Section 3.3.

We summarize our contributions as follows: 1) We propose a generalized framework in UDA, which is built upon adversarial dropout [30]. Our implementation supports both convolutional and fully connected layers; 2) We test on various domain adaptation benchmarks for image classification, and achieve competitive results compared to state-of-the-art methods; and 3) We extend the proposed method to a semantic segmentation task in UDA, where we perform adaptation from the simulation to real-world environments.

2. Related Work

Domain adaptation has been studied extensively. Ben-David *et al.* [1, 2] examined various divergence metrics between two domains, and defined an upper bound for the target domain error. Based on these studies, image-translation methods minimize the discrepancy between the two domains at an image-level [42, 51, 3].

On the other hand, feature alignment methods have attempted to match feature distributions between the source and target domains[11, 44, 23]. In particular, Ganin *et*

al. [11] proposed a domain adversarial training method that aims to generate domain-invariant features by deceiving a domain discriminator. Many recent works use domain adversarial training as a key component in their adaptation procedure [12, 4, 15, 40, 31, 47, 46]. However, the domain classifier cannot consider class labels; thus, the generated features tend to be sub-optimal for classification.

To overcome the weaknesses of domain adversarial training, more recent works directly deal with the relationship between the decision boundary and feature representations based on the cluster assumption [5]. Several works [25, 9, 40] exploit semi-supervised learning for domain adaptation. Besides, MCD [37] and ADR [36] use a minimax training method to push target feature distributions away from the decision boundary, where both methods are composed of the feature extractor and the classifiers. More precisely, in [36], two different classifiers are sampled via stochastic dropout. Then, for the same target data sample, the classifiers are updated to maximize the discrepancy between the two predictions. Lastly, the feature extractor is updated multiple times to minimize this discrepancy. The minimax training process leaves the classifier in a noise sensitive state. Therefore, it must be newly trained for optimal performance.

Though our work is partly inspired by ADR, the proposed method is more efficient and simpler to train compared to the prior arts [36, 37]. Instead of updating the classifier for maximizing discrepancy, we employ adversarial dropout [30] on the classifier to achieve a similar effect. Furthermore, this adversarial dropout can be applied to the feature extractor as well. Without the need of a minimax training scheme, DTA has a straightforward and reliable adaptation process.

Dropout is a simple yet effective regularization method that randomly drops a fraction of the neurons during the training process [41]. According to Srivastava *el al.* [41], dropout has the effect of ensembling multiple subsets of a network. Park *et al.* [29] spotlighted the efficacy of the dropout on convolutional layers. Tompson *el al.* [43] pointed out that activations of convolutional layers are usually surrounded by similar activations within the same feature map; thus, dropping individual neurons does not have a strong effect in convolution layers. Instead, they proposed spatial dropout, which drops entire feature maps instead of individual neurons. Building on spatial dropout, Hou *el al.* [16] proposed a weighted channel dropout that uses variable drop rates for individual channels, where the drop rates depend on the channel's averaged activation value. The weighted channel dropout is only applied to deep layers of the network, where activations are known to have high specificity [50, 49, 48]. Similarly, for channel-wise adversarial dropout, we remove entire feature maps in an adversarial way.

3. Proposed Method

3.1. Unsupervised Domain Adaptation

We first define the unsupervised domain adaptation (UDA) problem in general, and relevant notations to our work. In the UDA setting, we use data from two distinctive domains: the source domain $\mathcal{S} = \{X_s, Y_s\}$ and the target domain $\mathcal{T} = \{X_t\}$. A data point from the source domain $\mathbf{x}_s \in X_s$ has an associated label $y_s \in Y_s$, whereas one from the target domain $\mathbf{x}_t \in X_t$ has no paired ground-truth label. We employ a feature extractor $f(\mathbf{x}; \mathbf{m}_f)$, where $\mathbf{m}_f$ represents a dropout mask which can be applied at an arbitrary layer of the feature extractor. The feature extractor takes a data point from two domains $\mathbf{x} \sim \mathcal{S} \cup \mathcal{T}$ and creates a latent vector, which is fed into a classifier $c(\cdot; \mathbf{m}_c)$. The classifier applies a dropout mask $\mathbf{m}_c$ at an arbitrary layer. We denote the entire neural network as a composition of the feature extractor and the classifier: $h(\mathbf{x}; \mathbf{m}_f, \mathbf{m}_c) = c(f(\mathbf{x}; \mathbf{m}_f); \mathbf{m}_c)$.

3.2. Adversarial Dropout

We leverage a non-stochastic dropout mechanism, Adversarial Dropout (AdD) [30], for unsupervised domain adaptation. Adversarial dropout was originally proposed as an effective regularization method for supervised and semi-supervised learning. More specifically, Park *et al.* [30] define two types of Adversarial Dropout: Supervised Adversarial Dropout (SAdD), and Virtual Adversarial Dropout (VAdD). With access to ground truth labels, SAdD is used to maximize the divergence between a model's prediction and ground truth label. Without labels, on the other hand, VAdD is used to maximize the divergence between two independent predictions to an input. Due to the lack of target domain labels, SAdD cannot be employed for our purpose. Thus, we exclusively work with VAdD, which is referred to as AdD for the sake of convenience.

AdD provides a simple and efficient mechanism of generating two divergent predictions for an input. Ultimately, our goal is to enforce the cluster assumption on target data by minimizing the divergence between predictions. To this end, we introduce element-wise AdD (EAdD) and propose its variant, channel-wise AdD (CAdD).

We first define a dropout mask $\mathbf{m}$ applied to an intermediate layer of a network h. For simplicity, we decompose a network h into the subsequent sub-networks h_l and h_u by the layer applied dropout $\mathbf{m}$, such as:

$$h(\mathbf{x}; \mathbf{m}) = h_u(\mathbf{m} \odot h_l(\mathbf{x})), \qquad (1)$$

where $\odot$ represents the element-wise multiplication. Note that $\mathbf{m}$ has the same dimensions to the output of $h_l(\mathbf{x})$.

Let $D[p, p'] \geq 0$ measure the divergence between two distributions p and p'. Then, the divergence between the

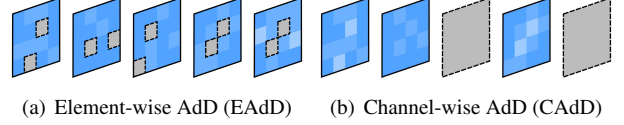(a) Element-wise AdD (EAdD)   (b) Channel-wise AdD (CAdD)

Figure 2. **Comparison of EAdD and CAdD.** EAdD drops units individually, regardless of spatial correlation. CAdD, on the other hand, drops entire feature maps, making it more suitable for convolutional layers.

predictions of $\mathbf{x}$ with different dropout masks, $\mathbf{m}$ and $\mathbf{m}^s$, is defined as:

$$D\left[h(\mathbf{x}; \mathbf{m}^s), h(\mathbf{x}; \mathbf{m})\right] \qquad (2)$$
$$= D\left[h_u(\mathbf{m}^s \odot h_l(\mathbf{x})), h_u(\mathbf{m} \odot h_l(\mathbf{x}))\right].$$

3.2.1 Element-wise Adversarial Dropout

The element-wise adversarial dropout (EAdD) mask $\mathbf{m}^{adv}$ is defined with respect to a stochastic dropout mask $\mathbf{m}^s$ as:

$$\mathbf{m}^{adv} = \underset{\mathbf{m}}{\operatorname{argmax}} D\left[h(\mathbf{x}; \mathbf{m}^s), h(\mathbf{x}; \mathbf{m})\right]$$
$$\text{where } \|\mathbf{m}^s - \mathbf{m}\| \leq \delta_e L, \qquad (3)$$

where L denotes the dimension of $\mathbf{m} \in \mathbb{R}^L$, and δ_e is a hyper parameter to control the perturbation magnitude with respect to $\mathbf{m}^s$. The objective is to find a minimally modified adversarial mask $\mathbf{m}^{adv}$ that maximizes the output divergence D between two independent forward passes of $\mathbf{x}$.

To find $\mathbf{m}^{adv}$, Park *et al.* [30] optimize a 0/1 knapsack problem with appropriate relaxations in the process. Their optimization process can be simplified into the following steps. First, an *impact value* is approximated for each element in $h_l(\mathbf{x})$, which is directly proportional to the element's contribution for increasing the divergence. When negative, the element has a decreasing effect on the divergence. Then, without breaching the boundary condition, the elements of $\mathbf{m}^s$ are adjusted to maximize divergence.

3.2.2 Channel-wise Adversarial Dropout

To use DTA in a wider range of tasks, we extend EAdD to convolutional layers. In these layers, however, standard dropout is relatively ineffective due to the strong spatial correlation between individual activations of a feature map [43]. EAdD dropout suffers from the same issues when naively applied to convolutional layers.

Hence, we formulate CAdD, which adversarially drops entire feature maps rather than individual activations. While the general procedure is similar to that of EAdD, we impose certain constraints on the mask to represent spatial dropout [43]. Fig. 2 highlights the difference between EAdD and CAdD.

Consider the activation of a convolutional layer, $h_l(\mathbf{x}) \in \mathbb{R}^{C \times H \times W}$, where C, H, and W denote the channel, height, and width dimensions of the activation, respectively. We define a channel-wise dropout mask $\mathbf{m}(i) \in \mathbb{R}^{H \times W}$, with the following constraints:

$$\mathbf{m}(i) = \mathbf{0} \text{ or } \mathbf{1}, \forall i \in \{1, \cdots, C\}. \tag{4}$$

Here, $\mathbf{m}(i)$ corresponds to the i-th activation map of $h_l(\mathbf{x})$, $\mathbf{0} \in \mathbb{R}^{H \times W}$ denotes a matrix of zeros, and $\mathbf{1} \in \mathbb{R}^{H \times W}$ denotes a matrix of ones, respectively. Then, the channel-wise adversarial dropout mask is defined as:

$$\mathbf{m}^{adv} = \underset{\mathbf{m}}{\arg\max} \, D\left[h(\mathbf{x}; \mathbf{m}^s), h(\mathbf{x}; \mathbf{m})\right],$$
$$\text{where } \frac{1}{HW} \sum_{i=1}^{C} \|\mathbf{m}^s(i) - \mathbf{m}(i)\| \leq \delta_c C. \tag{5}$$

As before, δ_c is the hyper parameter that controls degree of the perturbation.

The process of finding the channel-wise adversarial dropout mask $\mathbf{m}^{adv}$ is similar to those of element-wise adversarial dropout. For CAdD, however, the impact value is approximated for each activation map of $h_l(\mathbf{x})$ due to the constraints in Eq. (4). We provide the further details about the approximation in Appendix A of our supplementary material.

3.3. Drop to Adapt

Unlike the prior arts [37, 36], the proposed algorithm leverages a unified objective function to optimize all network parameters. The overall loss function is defined as a weighted sum of four objective functions:

$$L(\mathcal{S}, \mathcal{T}) = L_T(\mathcal{S}) + \lambda_1 L_{DTA}(\mathcal{T}) + \lambda_2 L_E(\mathcal{T}) + \lambda_3 L_V(\mathcal{T}), \tag{6}$$

where L_T, L_{DTA}, L_E, and L_V represent the objectives for task-specific, domain adaptation, entropy minimization and Virtual Adversarial Training (VAT) [27], respectively. Also, the associated hyper-parameters, λ_1, λ_2, and λ_3, control the relative importance of the terms.

Task-specific objective. We define the task-specific objective function L_T regarding the source domain $\mathcal{S}$. In practice, this objective function can be replaced according to the given task. As an example, we present the cross entropy which is widely used for classification:

$$L_T(\mathcal{S}) = -\mathbb{E}_{\mathbf{x}_s, y_s \sim \mathcal{S}}[\mathbf{y}_s^T \log h(\mathbf{x}_s)], \tag{7}$$

where $\mathbf{y}_s$ is one-hot encoded vector of y_s.

Domain adaptation objective. As the main component, we present the objective function for the domain adaptation first. The objective consists of two parts to affect on the feature extractor L_{fDTA} and the classifier L_{cDTA}:

$$L_{DTA}(\mathcal{T}) = L_{fDTA}(\mathcal{T}) + L_{cDTA}(\mathcal{T}). \tag{8}$$

We aim to minimize the divergence between two predicted distribution regarding to an input $\mathbf{x}$: one with a random dropout mask $\mathbf{m}_f^s$ and another with an adversarial dropout mask $\mathbf{m}_f^{adv}$. Among the various divergence measures, we choose the Kullback-Leibler (KL) divergence in this work. Assuming that the feature extractor consists of convolutional layers, we employ channel-wise adversarial dropout for $\mathbf{m}_f^{adv}$:

$$L_{fDTA}(\mathcal{T}) = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{T}}\Big[D\left[h(\mathbf{x}_t; \mathbf{m}_f^s), h(\mathbf{x}_t; \mathbf{m}_f^{adv})\right]\Big]$$
$$= \mathbb{E}_{\mathbf{x}_t \sim \mathcal{T}}\Big[D_{KL}\left[h(\mathbf{x}_t; \mathbf{m}_f^s)\|h(\mathbf{x}_t; \mathbf{m}_f^{adv}))\right]\Big]. \tag{9}$$

We illustrate the effects of L_{fDTA} in Fig. 1(c). Initially, the decision boundary crosses high density regions in the feature space (Fig. 1(a)), which is in violation of the cluster assumption. By applying adversarial dropout on the feature extractor, we cause certain features to cross the decision boundary (Fig. 1(c), left). Then, to enforce consistent predictions, the model parameter are updated to push the decision boundary away from these features (Fig. 1(c), right).

Similarly, we apply AdD to the classifier, where the classifier is defined as a series of fully connected layers. Thus, we perform the element-wise adversarial dropout $\mathbf{m}_c^{adv}$ and compute the divergence:

$$L_{cDTA}(\mathcal{T}) = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{T}}\Big[D_{KL}\left[h(\mathbf{x}_t; \mathbf{m}_c^s)\|h(\mathbf{x}_t; \mathbf{m}_c^{adv}))\right]\Big]. \tag{10}$$

When adversarial dropout is applied on the classifier, we determine the most volatile areas in the feature space. These volatile regions are in the vicinity of the decision boundary, and predictions in these regions can be changed even by a small perturbation. (Fig. 1(d), left). Therefore, minimizing L_{cDTA} lets the features avoid falling into such volatile regions (Fig. 1(d), right).

Entropy minimization objective. We introduce the entropy minimization objective to enforce the cluster assumption further. This loss penalizes target samples for being close to the decision boundary, and thus, causes the model to learn more discriminative features:

$$L_E(\mathcal{T}) = -\mathbb{E}_{\mathbf{x}_t \sim \mathcal{T}}[h(\mathbf{x}_t)^T \log h(\mathbf{x}_t)]. \tag{11}$$

VAT objective. Lastly, we exploit VAT, which adversarially perturbs the target data at the input level. The VAT minimization objective is defined as:

$$L_V(\mathcal{T}) = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{T}} \left[\max_{\|r\| \leq \epsilon} D_{KL}\left[h(\mathbf{x}_t) \| h(\mathbf{x}_t + r) \right] \right], \quad (12)$$

where r represents the virtual adversarial perturbation on input $\mathbf{x}_t$. While DTA and VAT are similarly motivated, they regularize the network with different forms of perturbations: network parameter perturbations (DTA) and input perturbations (VAT). Thus, VAT provides an orthogonal regularization to DTA, leading to complementary effects.

Interpretation of DTA. Fig. 3 visualizes the effects of adversarial dropout using Grad-GAM [39], which accentuates the most discriminative regions for a prediction. As a baseline, we present Grad-CAM visualizations of a model trained only on the source domain (SO, see Fig. 3(b)). We apply AdD on the source only model (SO + AdD), and see that semantically meaningful areas are deactivated. In contrast, our domain adapted model (DTA, see Fig. 3(d)) stays relatively unaffected by AdD, as it keeps seeing the same discriminative regions (see Fig. 3(e)) regardless of AdD. The visualizations imply that AdD promotes activations on more hidden units, and lends to robust decision boundary across the domains.

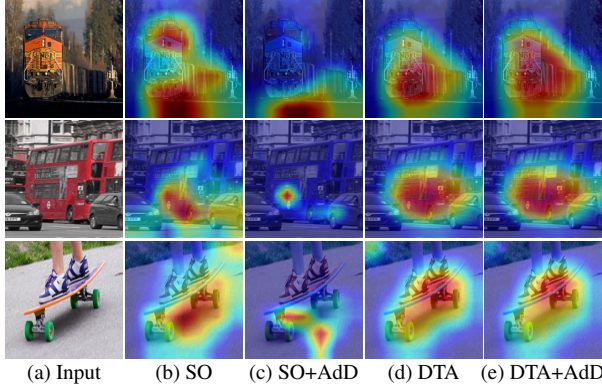

(a) Input (b) SO (c) SO+AdD (d) DTA (e) DTA+AdD

Figure 3. **Effect of adversarial dropout.** We visualize class activation maps on target domain images using GradCAM [39]. Adversarial dropout (c) effectively deactivates semantically meaningful regions for a prediction compared to its baseline model only trained on source domain (b). Our domain adapted model (DTA) produces reasonable predictions (d), even though 10% of units are eliminated by AdD (e).

4. Experimental Results

In this section, we evaluate the proposed method on small and large DA benchmarks. To demonstrate the generality of our model, we conduct the experiments in two major recognition tasks: classification and segmentation. In

Table 1. Results of experiment on small image datasets. *We compare with the MT+CT+TF for SE.

Source	SVHN	MNIST	USPS	STL	CIFAR
Target	MNIST	USPS	MNIST	CIFAR	STL
Source only (Ours)	76.5	96.3	76.9	60.1	78.2
SE* [9]	98.6	98.1	97.3	74.2	79.7
VADA [40]	94.5	-	-	73.5	80.0
DIRT-T [40]	99.4	-	-	75.5	-
Co-DA [19]	98.3	-	-	**76.4**	81.1
Co-DA+DIRT-T [19]	99.4	-	-	76.3	-
Ours	**99.4**	**99.5**	**99.1**	72.8	**82.6**
Target only (Ours)	99.6	97.8	99.6	90.4	70.0

each experiment, we select one domain as the source domain, and another as the target domain. We denote "Source only" as the target domain performance of a model trained on the source domain, and "Target only" as that of a model trained on the target domain. These two serve as baselines for the lower and upper bound performance in domain adaptation. We do not tune a set of data augmentation schemes nor do we report performance with ensemble predictions, as in French *el al.* [9]. Rather, all evaluation results are based on the same data augmentation strategy with a single model prediction.

4.1. DA on Small Datasets

To evaluate the influence of DTA model, we first perform experiments on small datasets. We use MNIST [20], USPS [17], and Street View House Numbers (SVHN) [28] for adaptation on digits recognition. For object recognition, we use CIFAR10 (CIFAR) [18] and STL10 (STL) [6]. For fair comparison against recent state-of-the-art methods such as Self-Ensembling (SE) [9], VADA [40], and DIRT-T [40], we conduct experiments on the same network architecture as in SE. Not that while VADA/DIRT-T use a slightly differernet architecture, the total number of parameters are comparable. The results can be found in Table 1, and a full list of hyperparameter settings can be found in Appendix B.

SVHN → MNIST. SVHN and MNIST are two digit classification datasets with a drastic distributional shift between the two. While MNIST consists of binary handwritten digit images, SVHN consists of colored images of street house numbers. Since MNIST has a significantly lower image dimensionality than SVHN, we adopt the dimension of MNIST to 32×32 of SVHN, with three channels. When the proposed DTA is applied, our approach demonstrates a significant improvement over previous works, and achieves a performance similar to the "Target only" performance on MNIST.

MNIST ↔ USPS. MNIST and USPS contain grayscale images, so the domain shift between these two datasets is relatively smaller compared to that of the SVHN → MNIST

Table 2. Results on VisDA-2017 classification using ResNet-101.

	aero.	bike	bus	car	horse	knife	moto.	person	plant	sktb.	train	truck	avg.
Source Only	46.2	27.6	31.4	78.1	71.8	1.3	71.7	14.3	63.5	31.0	93.7	3.2	50.8
DAN [23]	87.1	63.0	76.5	42.0	90.3	42.9	85.9	53.1	49.7	36.3	85.8	20.7	61.1
DANN [11]	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4
MCD [37]	87.0	60.9	83.7	64.0	88.9	79.6	84.7	76.9	88.6	40.3	83.0	25.8	71.9
ADR [36]	87.8	79.5	83.7	65.3	92.3	61.8	88.9	73.2	87.8	60.0	85.5	**32.3**	74.8
Ours	**93.7**	**82.2**	**85.6**	**83.8**	**93.0**	**81.0**	**90.7**	**82.1**	**95.1**	**78.1**	**86.4**	32.1	**81.5**

setting. In both adaptation directions, we achieve an accuracy close to the performance of fully supervised learning on the target domain. In fact, we obtain higher accuracy on USPS when adapting from MNIST, than when trained directly on USPS. This is because the USPS training is relatively small, allowing us to achieve improved performance by adapting from MNIST, using DTA.

CIFAR ↔ STL. CIFAR and STL are 10-class object recognition datasets with colored images. We remove the non-overlapping classes and redefine the task as a 9-class classification task. Furthermore, we downscale the 96×96 image dimesion of STL to match the 32×32 dimension of CIFAR. In the **CIFAR → STL** setting, our method's performance surpasses others by a comfortable margin. For the same reasons presented in the **MNIST → USPS** setting, our adapted model outperforms the target only model on this dataset pair. In **STL → CIFAR**, however, our method is slightly weak. This is because STL contains a very small dataset, with only 50 images per class. Since DTA regularizes the decision boundary of the model, the inherent assumption is that the model can achieve low generalization error on the source domain. This assumption holds in most cases, but breaks down when STL is the source domain.

To summarize, we achieve a substantial margin of improvement over the source only model across all domain configurations. In four of the five configurations, our method outperforms the recent state-of-the-art results. Next, we evaluate our method on more practical settings that embody real-life domain adaptation scenarios.

4.2. DA on Large Datasets

We apply our method to adaptation on large-scale, large-image datasets. In particular, we evaluate on VisDA-2017 [32] image classification and VisDA-2017 image segmentation tasks.

Classification. The VisDA-2017 image classification is a 12-class domain adaptation problem. The source domain consists of 152,397 synthetic images, where 3D CAD models are rendered from various conditions. The target domain consists of 55,388 real images taken from the MS-COCO dataset [21]. Since the objective is to learn from labeled
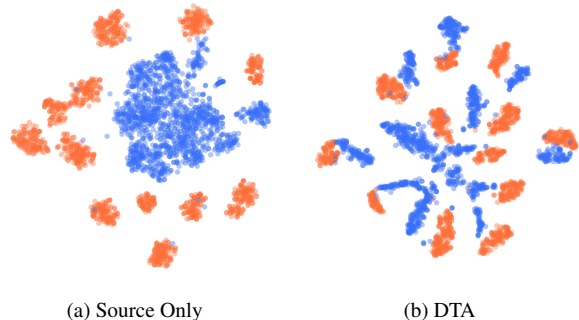

(a) Source Only (b) DTA

Figure 4. **t-SNE.** t-SNE visualization of VisDA-2017 classification dataset using ResNet-101, before and after adaptation with DTA. t-SNE hyperparameters are consistent in both visualizations.

synthetic images and correctly predict the class of real images, this dataset has been frequently used in many domain adaptation works [23, 12, 37, 36, 9]. For fair comparison with recent works, we follow the protocol of ADR [36] in our experiments. Specifically, we apply the EAdD after the second fully connected layer, and CAdD within the last convolution layer of ResNet-50 [14] and ResNet-101 models. Both models are initialized with weights from an ImageNet [8] pre-trained model. For more details on implementation, we refer our readers to Appendix B.

The per-class adaptation performance with a ResNet-101 backbone can be found in Table 2. The table clearly shows that our proposed method surpasses previous methods by a large margin. Note that all methods in this table use the same ResNet-101 backbone. Compared to the performance of a source only model, we achieve a 30.7% improvement (or 60.4% relative improvement) on the average accuracy. Furthermore, DTA shows a significant improvement across all categories; in fact, it achieves the best per-class performance in all classes, except the "truck" class, where it falls behind ADR by a mere 0.2%. Although our source only model is slightly lower than that of both MCD [37] and ADR, our proposed method effectively generalizes a model from the source to target domain, with stronger adaptation performance over MCD and ADR by margins of 9.6% and 6.7%, respectively.

In Table 4, we show that it is feasible to apply DTA on a different backbone network with success. Similarly to

Table 3. Results on GTA → Cityscapes, using a modified FCN with ResNet-50 as the base network.

	road	sidewalk	building	wall	fence	pole	t light	t sign	veg	terrian	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
Source only	25.3	13.7	56.8	2.7	**17.2**	**21.2**	20.0	**8.7**	75.3	11.2	72.0	45.7	4.9	42.2	14.2	20.2	0.4	**19.5**	0.0	24.8
DANN	72.4	19.1	73.0	3.9	9.3	17.3	13.1	5.5	71.0	20.1	62.2	32.6	5.2	68.4	12.1	9.9	0.0	5.8	0.0	26.4
ADR	87.8	15.6	**77.4**	20.6	9.7	19.0	19.9	7.7	82.0	**31.5**	74.3	43.5	9.0	**77.8**	17.5	27.7	**1.8**	9.7	0.0	33.3
Ours	**88.8**	**36.9**	76.9	**20.9**	15.4	19.6	**21.8**	7.9	**82.9**	26.7	**76.1**	**51.7**	**9.4**	76.1	**22.4**	**28.9**	1.7	15.2	0.0	**35.8**

Table 4. Results on VisDA-2017 classification using ResNet-50. *SE report ensemble of multiple predictions. All other methods, including ours, report the average achieved by a single prediction.

Method	avg.
Source Only (Ours)	45.6
DAN [23]	53.0
RTN [25]	53.6
DANN [11]	55.0
JAN-A [26]	61.6
GTA [38]	69.5
SimNet [33]	69.6
CDAN-E [24]	70.0
Ours	**76.2**
SE* [9]	82.8

DTA on ResNet-101, our model outperforms recent previous methods, and demonstrates a significance improvement over the source only model. While SE reports the best overall performance, we do not consider it to be comparable to other methods - including ours - because the reported accuracy is a result of 16 ensembled predictions.

For qualitative analysis, Figure 4 visualizes the feature representations of VisDA-2017 classification with t-SNE [45]. The source only model shows strong clustering of the source domain's synthetic image samples (blue), but fails to have similar influence on the target domain's real image samples (red). During training, DTA constantly enforces the clustering of target samples by stimulating the feature representations and decision boundary of the model. Therefore, we can clearly see an improved separation of target features with DTA, resulting in the best performance in VisDA-2017.

Segmentation. To further demonstrate our method's applicability to real-world adaptation settings, we evaluate DTA in the challenging VisDA-2017 semantic segmentation task. For the source domain, we use the synthetic GTA5 [34] dataset which consists of 24966 labeled images. As the target domain, we use the real-world Cityscapes [7], consisting of 5000 images. Both datasets are evaluated on the same category of 19 classes, with the mean Intersection-over-Union (mIoU) metric. For fair comparison with recent methods [12, 36], we follow the procedure of ADR and use a modified version of Fully Convolutional Networks

(FCN) [22] on a ResNet-50 backbone. We apply CAdD within the last convolutional layer of ResNet-50.

We report our results in Table 3, alongside results of existing methods. Our method clearly improves upon the mIoU of not only the source only model, but also competing methods. Even with the same training procedure and settings as in the classification experiments, DTA is extremely effective at adapting the most common classes in the dataset. This conclusion is supported in Figure 5, where we display examples of input images, ground truths, and the corresponding outputs of source only and DTA model. While the source only predictions are erroneous in most classes, DTA's predictions are relatively clean and accurate.

5. Discussion

Although the proposed DTA shows significant improvements on multiple visual tasks, we would like to understand the role of each component in DTA and how their combination operates in practice. We perform a series of ablation experiments and present the results in Table 5. All ablations are conducted on VisDA-2017 image classification dataset. To verify the effectiveness and generality, we use ResNet-50 and ResNet-101 models for all experiments in this ablation. The modified ResNet-based models consist of the original convolutional layers with FAdD after the second fully connected layer, and CAdD within the last convolutional layer. The entropy loss term in Eq. (11) is applied on all ablations except the "Source Only" setting.

To assess whether each module of DTA (VAT, fDTA, cDTA) plays an important role in the performance, we first experiment with individual modules. Overall, all three modules improve the performance over a source only model. We observe that the three components contribute to the accuracy of each category differently. In ResNet-101, while fDTA has a great impact on the "knife" category, VAT significantly boosts the performance of the "skakteboard" class. Theoretically, VAT [27] can be seen as the regularization by perturbing the input image, while the proposed methods can be seen as perturbations on the feature space of the model. Therefore, we can see that two combinations (fDTA + VAT), (cDTA + VAT) shows increased performance compared to the individually regularized model (*i.e.* 73.2% (VAT) / 77.0% (cDTA) → 81.2% in ResNet-
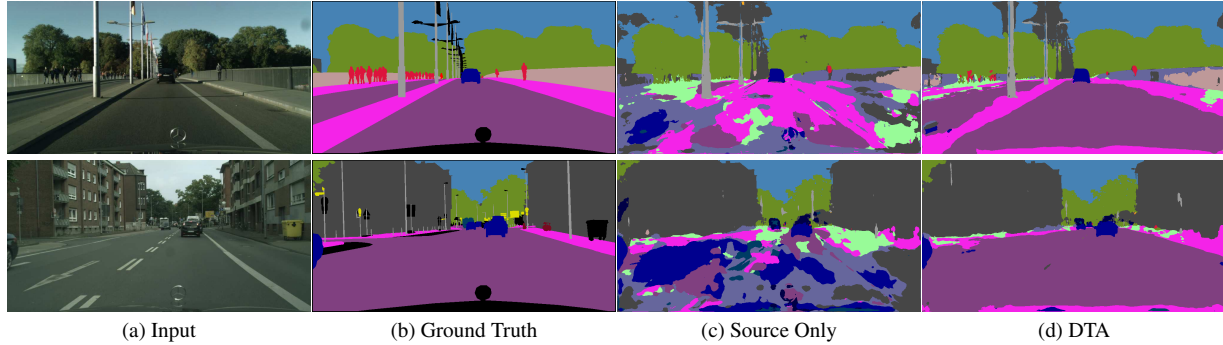
| (a) Input | (b) Ground Truth | (c) Source Only | (d) DTA |

Figure 5. **Semantic segmentation.** Qualitative results of the semantic segmentation task on GTA → Cityscapes, before and after adaptation with DTA. We use a modified FCN architecture with ResNet-50 as the base model.

Table 5. Ablation Studies on VisDA-2017 Classification Dataset

Methods	aero.	bike	bus	car	horse	knife	moto.	person	plant	sktb.	train	truck	avg.
ResNet-50													
Source Only	54.2	27.7	17.6	57.1	48.4	4.0	86.4	11.0	69.1	15.6	95.7	7.3	46.0
VAT	83.1	62.5	70.5	53.0	81.8	13.2	89.9	74.4	88.5	41.1	89.0	38.2	67.1
fDTA	88.8	58.2	82.8	82.3	90.4	0.1	92.8	77.3	94.2	78.5	86.9	0.2	72.5
fDTA + VAT	91.3	66.3	77.7	77.5	91.0	13.1	92.6	83.0	94.2	58.0	85.9	12.0	73.1
cDTA	92.4	72.9	75.1	72.6	92.8	7.4	90.8	82.1	95.0	66.6	87.8	31.6	74.7
cDTA + VAT	90.0	72.7	83.7	79.3	92.0	6.8	91.4	82.6	92.2	70.4	86.3	22.9	75.4
cDTA + fDTA	88.2	68.8	87.2	82.8	92.3	5.8	89.4	78.4	95.5	74.8	82.4	16.1	75.0
Ours	93.1	70.5	83.8	87.0	92.3	3.3	91.9	86.4	93.1	71.0	82.0	15.3	76.2
ResNet-101													
Source Only	46.2	27.6	31.4	78.1	71.8	1.4	71.6	14.3	63.5	31.0	93.7	3.2	50.8
VAT	90.1	43.9	83.9	85.6	90.9	1.4	95.0	78.6	93.8	57.9	86.2	13.4	73.2
fDTA	89.1	75.5	84.6	87.2	92.3	72.9	89.7	78.5	91.8	39.5	84.1	10.8	76.4
fDTA + VAT	93.0	84.8	81.8	78.1	93.2	70.1	88.8	82.0	94.0	81.5	87.4	39.6	80.5
cDTA	91.8	81.5	78.7	67.0	91.3	71.6	85.3	76.9	93.5	72.5	86.7	44.1	77.0
cDTA + VAT	93.8	86.1	82.9	78.3	92.2	83.9	88.2	80.6	94.1	82.2	88.0	40.0	81.2
cDTA + fDTA	91.7	77.7	78.8	75.2	91.0	73.2	88.4	78.8	93.2	56.6	88.7	35.6	77.4
Ours	93.7	82.2	85.6	83.8	93.0	81.0	90.7	82.0	95.1	78.1	86.4	32.1	81.5

101, 67.1% (VAT) / 72.5% (fDTA) → 73.1% in ResNet-50). These results suggest that it is beneficial to use VAT [27] with the proposed method. More specifically, both methods exhibit complementary effects for adaptation on a large domain shift. This advantage can also be observed in the comparison of fDTA + cDTA to a final version of the proposed method (VAT + fDTA + cDTA). One interesting point is that all these trends are mostly maintained in both backbone models; the only difference is the amount of margin between the performance of source only and individual models. From this fact, we conclude that the proposed method can act as a general regularization technique for adaptation, regardless of the model's capacity.

6. Conclusion

We presented a simple yet effective method for unsupervised domain adaptation despite large domain shifts. With two types of proposed adversarial dropout modules, EAdD and CAdD, we enforced the cluster assumption on the tar-

get domain. The proposed methods are easily integrated into existing deep learning architectures. Through extensive experiments on various small and large datasets, we demonstrated the effectiveness of the proposed method on two domain adaptation tasks, and in all cases we achieved significant improvement as compared to the source-only model and the state-of-the-art results.

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Mach. Learn.*, 2010.

[2] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *NIPS*, 2006.

[3] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*, 2017.

[4] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NIPS*, 2016.

[5] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *AISTATS*, 2005.

[6] Adam Coates, Honglak Lee, and Andrew Ng. An analysis of single layer networks in unsupervised feature learning. In *AISTATS*, 2011.

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[9] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *ICLR*, 2018.

[10] A Gaidon, Q Wang, Y Cabon, and E Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

[11] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

[12] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 2013.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[15] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018.

[16] Saihui Hou and Zilei Wang. Weighted channel dropout for regularization of deep convolutional neural network. In *AAAI*, 2019.

[17] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE TPAMI*, 1994.

[18] Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009. *Tech Report*.

[19] Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, Bill Freeman, and Gregory Wornell. Co-regularized alignment for unsupervised domain adaptation. In *NIPS*, 2018.

[20] Yann LeCun, Leon Bottou, Yoshua Bengio, and Pattrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 1998.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[23] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[24] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation. In *NIPS*, 2018.

[25] Mingsheng Long, H. Zhu, J. Wang, and Michael I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, 2016.

[26] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017.

[27] T. Miyato, S. i. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE TPAMI*, 2018.

[28] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011.

[29] Sungheon Park and Nojun Kwak. Analysis on the dropout effect in convolutional neural networks. *ACCV*, 2016.

[30] Sungrae Park, JunKeon Park, Su-Jin Shin, and Il-Chul Moon. Adversarial dropout for supervised and semi-supervised learning. In *AAAI*, 2018.

[31] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *AAAI*, 2018.

[32] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. VisDA: The visual domain adaptation challenge, 2017.

[33] Pedro O. Pinheiro. Unsupervised domain adaptation with similarity learning. In *CVPR*, 2018.

[34] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016.

[35] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[36] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Adversarial dropout regularization. In *ICLR*, 2018.

[37] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018.

[38] Swami Sankaranarayanan, Yogesh Balaji, Carlos D. Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *CVPR*, 2018.

[39] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE ICCV*, 2017.

[40] Rui Shu, Hung Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-T approach to unsupervised domain adaptation. In *ICLR*, 2018.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014.

[42] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. *ICLR*, 2017.

[43] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *CVPR*, 2015.

[44] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017.

[45] Laurens van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008.

[46] Riccardo Volpi, Pietro Morerio, Silvio Savarese, and Vittorio Murino. Adversarial feature augmentation for unsupervised domain adaptation. In *CVPR*, 2018.

[47] Ximei Wang, Liang Li, Weirui Ye, Mingsheng Long, and Jianmin Wang. Transferable attention for domain adaptation. In *AAAI*, 2019.

[48] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014.

[49] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

[50] Xiaopeng Zhang, Hongkai Xiong, Wengang Zhou, Weiyao Lin, and Qi Tian. Picking deep filter responses for fine-grained image recognition. In *CVPR*, 2016.

[51] Jun-Yan. Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.