Domain Adaptation with Adversarial Training on Penultimate Activations

Tao Sun [1], Cheng Lu [2], Haibin Ling [1]

[1] Stony Brook University, USA
[2] XPeng Motors, USA
{tao,hling}@cs.stonybrook.edu, luc@xiaopeng.com

Abstract

Enhancing model prediction confidence on target data is an important objective in Unsupervised Domain Adaptation (UDA). In this paper, we explore adversarial training on penultimate activations, *i.e.*, input features of the final linear classification layer. We show that this strategy is more efficient and better correlated with the objective of boosting prediction confidence than adversarial training on input images or intermediate features, as used in previous works. Furthermore, with activation normalization commonly used in domain adaptation to reduce domain gap, we derive two variants and systematically analyze the effects of normalization on our adversarial training. This is illustrated both in theory and through empirical analysis on real adaptation tasks. Extensive experiments are conducted on popular UDA benchmarks under both standard setting and source-data free setting. The results validate that our method achieves the best scores against previous arts. Code is available at https://github.com/tsun/APA.

Introduction

Unsupervised Domain Adaptation (UDA) aims to transfer knowledge from a label rich source domain to an unlabeled target domain (Pan and Yang 2009; Wang and Deng 2018; Wilson and Cook 2020). The two domains are relevant, yet there is often a distribution shift between them. Due to the domain gap, models from the source domain often predict incorrectly or less confidently on the target domain. Thus, how to enhance model prediction confidence on target data becomes critical in UDA.

The mainstream paradigm for UDA is *feature adaptation* (Sun and Saenko 2016; Tzeng et al. 2014; Ganin and Lempitsky 2015; Tzeng et al. 2017), which learns domain-invariant features. Despite its popularity, this paradigm has some intrinsic limitations in the situation with label distribution shift (Li et al. 2020a; Prabhu et al. 2021). Recently, *self-training* (Zhu 2005) has drawn increasing attention in UDA, which applies *consistency regularization* or *pseudo-labeling* loss on the unlabeled target data. Self-training effectively enhances prediction confidence on target data, showing superior results (Liang, Hu, and Feng 2020; Prabhu et al. 2021; Liu, Wang, and Long 2021; Sun, Lu, and Ling 2022).

Adversarial Training (Goodfellow, Shlens, and Szegedy 2015; Jeddi et al. 2020; Shu et al. 2021) aims to improve model's robustness by injecting adversarial examples. Originally, adversarial training is proposed for supervised learning and requires the ground-truth class labels. Miyato et al. (2018) generalize it to semi-supervised learning and propose Virtual Adversarial Training (VAT). VAT can be viewed as a consistency regularization, whose main idea is to force the model to make similar predictions on clean and perturbed images. VAT is later introduced into domain adaptation to improve local smoothness (Shu et al. 2018). Despite its popularity (Cicek and Soatto 2019; Kumar et al. 2018; Lee et al. 2019b), VAT is commonly used as an auxiliary loss. In contrast, training with VAT alone often leads to poor performance on domain adaptation benchmarks (Liu, Wang, and Long 2021; Lee et al. 2019b; Zhang et al. 2021). Another drawback is that VAT requires an additional back-propagation through the entire network to obtain gradients, which increases the computation cost.

The above observations motivate us to seek more effective ways of adversarial training for UDA problems. A common practice in UDA is to split the training process into a source-training stage and a target adaptation stage. Due to the relevance between the two domains, we find that weights of the linear classification layer change quite slowly in the second stage. In fact, some source-data free works (Liang, Hu, and Feng 2020) freeze the classifier during adaptation and focus on improving feature extractor. Since our goal is to enhance prediction confidence on unlabeled target samples, adversarial training on *penultimate activations*, *i.e.*, the input features of the final linear classification layer, is more correlated with this goal than adversarial training on input images or intermediate features. Our analysis shows that adversarial perturbations on other layer representations can be mapped into perturbations on penultimate activations. The mapping maintains the value of adversarial loss but leads to much higher accuracies. This indicates that it is more effective to manipulate the penultimate activations. Moreover, approximating the optimal adversarial perturbation for penultimate activations is also much easier as it only involves the linear classifier in the optimization problem.

We propose a framework of domain adaptation with adversarial training on penultimate activations. The method is applicable to both standard and source-data free setting. As activation normalization with ℓ_2 norm is commonly used in UDA to reduce domain gap (Xu et al. 2019; Prabhu et al. 2021), we derive two variants and systematically analyze the effects of normalization on our adversarial training. The shrinking effect on adversarial loss gradients is discussed. We also analyze the correlations among adversarial perturbations, activation gradients and actual activation changes after model update on real adaptation tasks.

To summarize, our contributions include: 1) we propose a novel UDA framework of adversarial training on penultimate activations; 2) we systematically analyze its relations and advantages over previous adversarial training on input images or intermediate features; and 3) we conduct extensive experiments to validate the superior performance of our method under both standard and source-data free settings.

Related Work

Unsupervised Domain Adaptation. Many UDA methods have been proposed recently (Pan and Yang 2009; Wang and Deng 2018; Wilson and Cook 2020). A mainstream paradigm is feature adaptation that learns domain invariant feature representations. Some approaches minimize domain discrepancy statistics (Long et al. 2017; Sun and Saenko 2016), and others utilize the idea of adversarial learning by updating feature extractor to fool a domain discriminator (Tzeng et al. 2014; Ganin and Lempitsky 2015; Tzeng et al. 2017). Despite the popularity of this paradigm, recent works (Li et al. 2020a; Prabhu et al. 2021) show its intrinsic limitations when the label distribution changes across domains. On the other hand, self-training has become a promising paradigm, which adopts the ideas from semi-supervised learning to exploit unlabeled data. One line of pseudo-labeling uses pseudo-labels generated by the model as supervision to update itself (Liang, Hu, and Feng 2020; Liu, Wang, and Long 2021; Sun, Lu, and Ling 2022). Another line of consistency regularization improves local smoothness using pairs of semantically identical predictions (Sun et al. 2022). Self-training does not force to align source and target domains. Hence it alleviates negative transfer under a large domain shift.

Adversarial training. Adversarial training aims to produce robust models by injecting adversarial examples (Goodfellow, Shlens, and Szegedy 2015; Jeddi et al. 2020; Shu et al. 2021). Many adversarial attack methods have been designed to generate adversarial examples. Among them, gradient-based attack is a major category that obtains adversarial perturbations from adversarial loss gradients on clean data (Goodfellow, Shlens, and Szegedy 2015; Madry et al. 2017). Traditionally, the labeling information is required to generate perturbation directions. Miyato et al. (2018) propose Virtual Adversarial Training (VAT) for semi-supervised learning. Adversarial training can be applied on input images (Goodfellow, Shlens, and Szegedy 2015; Miyato et al. 2018) or intermediate features (Jeddi et al. 2020; Shu et al. 2021). Recently, Chen et al. (2022) show that adversarial augmentation on intermediate features yields good performance across diverse visual recognition tasks.

Adversarial training in DA. Shu et al. (2018) first use VAT to incorporate the locally-Lipschitz constraint in conditional entropy minimization. VAT is also used as a smooth regularizer in (Cicek and Soatto 2019; Kumar et al. 2018; Lee et al. 2019b; Li et al. 2020b). Jiang et al. (2020a) devise a bidirectional adversarial training network for SSDA, and penalize VAT with entropy. Liu et al. (2019) generate transferable examples to fill in the domain gap in the feature space. Lee et al. (2019b) propose to learn discriminative features through element-wise and channel-wise adversarial dropout. Kim and Kim (2020) perturb target samples to reduce the intra-domain discrepancy in SSDA.

Approach

Notations and Preliminaries

In UDA, there is a source domain $\mathcal{P}_s(\mathcal{X}, \mathcal{Y})$ and a target domain $\mathcal{P}_t(\mathcal{X}, \mathcal{Y})$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input image space and the label space respectively. We have access to labeled source domain samples $\mathcal{D}_s = \{(\boldsymbol{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ and unlabeled target domain samples $\mathcal{D}_t = \{(\boldsymbol{x}_i^t)\}_{i=1}^{n_t}$. The goal is to learn a classifier $h = g \circ f$, where $f : \mathcal{X} \to \mathcal{Z}$ denotes the feature extractor, $g : \mathcal{Z} \to \mathcal{Y}$ denotes the class predictor, and $\mathcal{Z}$ is the latent feature space. In particular, we assume that g is a linear classifier with trainable weights W and biases B, *i.e.*, $g(z) = Wz + B$. $z = f(\boldsymbol{x})$ is then called the penultimate activation (Seo, Lee, and Kwak 2021).

VAT (Miyato et al. 2018) smooths the model in semi-supervised learning by applying adversarial training on unlabeled data. Its objective is

$$\begin{aligned} &\ell(\boldsymbol{x}) = D\big[p(y|\boldsymbol{x}), p(y|\boldsymbol{x} + \boldsymbol{r}^{(v)})\big] \\ &\text{with } \boldsymbol{r}^{(v)} \triangleq \underset{\|\boldsymbol{r}\|_2 \leq \epsilon}{\arg\max}\ \ell_r(\boldsymbol{r}) = D\big[p(y|\boldsymbol{x}), p(y|\boldsymbol{x} + \boldsymbol{r})\big] \end{aligned} \tag{1}$$

where $D[\cdot,\cdot]$ is Kullback-Leibler divergence. Let $\overline{\text{overlines}}$ indicate ℓ_2 normalization, *e.g.*, $\overline{\boldsymbol{z}} = \boldsymbol{z}/\|\boldsymbol{z}\|_2$ for a vector $\boldsymbol{z}$. In (Miyato et al. 2018), $\boldsymbol{r}^{(v)}$ is approximated by

$$\boldsymbol{r}^{(v)} = \epsilon \overline{\nabla_{\boldsymbol{r}} \ell_r(\boldsymbol{r})|_{\boldsymbol{r}=\xi \boldsymbol{d}}} \tag{2}$$

in which ϵ is the perturbation magnitude, ξ is a small constant and $\boldsymbol{d}$ is a random unit vector. The perturbation is "virtually" adversarial because it does not come from the ground-truth labels.

Adversarial Training on Penultimate Activations

Instead of adversarial training on input images or intermediate features, we propose Adversarial training on Penultimate Activations (APA) for unsupervised domain adaptation. We first describe motivations and details of our method, along with two variants with activation normalization in this section. Then we will discuss its relation and advantages over adversarial training on other layer representations, and make an in-depth analysis in the next section.

Two-stage training pipeline. In consistent with previous works (Prabhu et al. 2021; Liang, Hu, and Feng 2020), we split the training process into a source-training stage and a target adaptation stage, shown in Fig. 1. During the first stage, models are trained on source data only with standard cross entropy loss. In the second stage, the obtained source

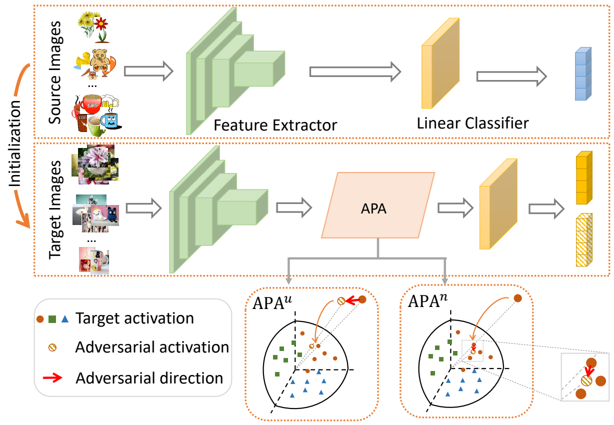

Figure 1: Framework of the proposed Adversarial Training on Penultimate Activations. APA^u and APA^n apply adversarial perturbations on the un-normalized and normalized activations, respectively.

models are adapted to the target domain using target data (and source data when available). The adversarial training is applied in the second stage to enhance prediction confidence on unlabeled target samples.

Motivations. Since the classifier is initialized with source data and the two domains are relevant, we first want to know how fast the classifier weights change during the second stage. In Fig. 2, we plot the averaged cosine similarity between initial classifier weights $W^{(0)}$ and the weights W during training on 12 tasks of Office-Home. It shows that the weights change quite slowly. Therefore, it is reasonable to assume that the decision boundaries of g change negligibly within a short period of training epochs. Some work (Liang, Hu, and Feng 2020) freeze the classifier during adaptation. We do not choose that, but find freezing the classifier does not make much difference to the performance (*c.r.* Tab. 7).

To improve prediction confidence on unlabeled target data, a natural way is to move their penultimate activations away from the decision boundaries. This can be realized through adversarial training. Alternatively, we can apply adversarial training on input images or intermediate features to update the penultimate activations in an indirect manner. But this would be less effective. We will discuss this in detail in the approach analysis section.

Method formulation. We adopt adversarial training on penultimate activations to improve prediction confidence of target data during the adaptation stage. The objective is

$$\begin{aligned} \ell^{(p)}(\boldsymbol{x}) &= D\big[g_\sigma(f(\boldsymbol{x})), g_\sigma(f(\boldsymbol{x}) + \boldsymbol{r}^{(p)})\big] \\ \text{with } \boldsymbol{r}^{(p)} &\triangleq \underset{\|\boldsymbol{r}\|_2 \le \epsilon}{\arg\max}\ \ell_r^{(p)}(\boldsymbol{r}) \\ &= \underset{\|\boldsymbol{r}\|_2 \le \epsilon}{\arg\max}\ D\big[g_\sigma(f(\boldsymbol{x})), g_\sigma(f(\boldsymbol{x}) + \boldsymbol{r})\big] \end{aligned} \tag{3}$$

where $g_\sigma(\cdot) \triangleq \sigma(g(\cdot))$ applies the `softmax` operator σ after the g. This `argmax` optimization problem only involves g as $f(\boldsymbol{x})$ is fixed and g is a *linear* function, thus can be solved efficiently. In contrast, approximating $\boldsymbol{r}^{(v)}$ needs to back-propagate through the entire highly nonlinear deep neural network, which is often computationally expensive. $\boldsymbol{r}^{(p)}$ can be approximated by

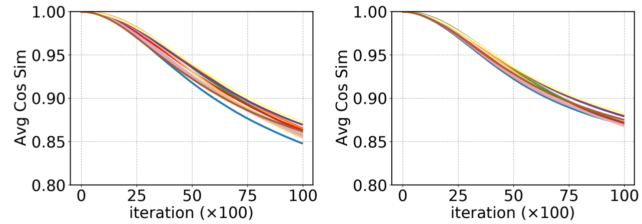

Figure 2: Averaged cosine similarity between initial classifier weights $W^{(0)}$ and current weights W on 12 tasks of Office-Home under (left) standard and (right) source-data free settings.

$$\boldsymbol{r}^{(p)} = \epsilon \cdot \overline{\nabla_{\boldsymbol{r}} \ell_r^{(p)}(\boldsymbol{r})|_{\boldsymbol{r}=\xi \boldsymbol{d}}} \tag{4}$$

The objective of APA is

$$\min_{f,g} \mathcal{L} = \mathbb{E}_{(\boldsymbol{x}_s, y_s) \overset{bal}{\sim} \mathcal{P}_s} \ell_{\text{ce}}(\boldsymbol{x}_s, y_s) + \beta \mathbb{E}_{\boldsymbol{x}_t \overset{bal}{\sim} \mathcal{P}_t} \ell^{(p)}(\boldsymbol{x}_t) \tag{5}$$

where ℓ_{ce} is cross entropy loss and β is a hyper-parameter. Following previous works (Jiang et al. 2020b; Prabhu et al. 2021) that tackle label shift, we use (pseudo) class-balanced sampling during training.

The method can be easily extended to source-data free setting. Since source samples are unavailable during the adaptation stage, we use pseudo labels of confident target samples as an additional supervision term. The objective is

$$\min_{f,g} \mathcal{L}_{\text{sf}} = \mathbb{E}_{(\boldsymbol{x}_t, \hat{y}_t) \overset{bal}{\sim} \mathcal{P}_t} I_{\hat{p}_t}^{\tau} \ell_{\text{ce}}(\boldsymbol{x}_t, \hat{y}_t) + \beta \ell^{(p)}(\boldsymbol{x}_t) \tag{6}$$

where $\hat{y}_t$ is the pseudo label of $\boldsymbol{x}_t$. $I_{\hat{p}_t}^{\tau}$ is 1 if $max(\hat{p}_t) \geq \tau$ and 0 otherwise. τ is a confidence threshold.

Variants with activation normalization. Using ℓ_2 normalization on penultimate activations is a common technique to reduce domain gap (Xu et al. 2019; Prabhu et al. 2021). It follows two variants of our method, APA^u and APA^n, that apply adversarial training on the un-normalized and normalized activations, respectively. Figure 1 illustrates the framework.

The adversarial loss for APA^u is

$$\begin{aligned} \ell^{(p_u)}(\boldsymbol{x}) &= D\big[g_\sigma(\overline{f(\boldsymbol{x})}), g_\sigma(\overline{f(\boldsymbol{x}) + \boldsymbol{r}^{(p_u)}})\big] \\ \text{with } \boldsymbol{r}^{(p_u)} &\triangleq \underset{\|\boldsymbol{r}\|_2 \le \epsilon}{\arg\max} D\big[g_\sigma(\overline{f(\boldsymbol{x})}), g_\sigma(\overline{f(\boldsymbol{x}) + \boldsymbol{r}})\big] \end{aligned} \tag{7}$$

and the adversarial loss for APA^n is

$$\begin{aligned} \ell^{(p_n)}(\boldsymbol{x}) &= D\big[g_\sigma(\overline{f(\boldsymbol{x})}), g_\sigma(\overline{f(\boldsymbol{x})} + \boldsymbol{r}^{(p_n)})\big] \\ \text{with } \boldsymbol{r}^{(p_n)} &\triangleq \underset{\|\boldsymbol{r}\|_2 \le \epsilon}{\arg\max} D\big[g_\sigma(\overline{f(\boldsymbol{x})}), g_\sigma(\overline{\overline{f(\boldsymbol{x})} + \boldsymbol{r}})\big] \end{aligned} \tag{8}$$

where the optimal perturbation $\boldsymbol{r}^{(p_u)}$ and $\boldsymbol{r}^{(p_n)}$ can be approximated in a similar way as Eq. 4. $f(\boldsymbol{x})$ is also fixed in the `argmax` optimization problems.

	Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg.
UDA	ResNet-50	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
	DANN	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
	CDAN	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
	SAFN	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3
	MDD	54.9	73.7	77.8	60.0	71.4	71.8	61.2	53.6	78.1	72.5	60.2	82.3	68.1
	SENTRY	61.8	77.4	80.1	66.3	71.6	74.7	66.8	**63.0**	80.9	74.0	**66.3**	84.1	72.2
	CST	59.0	79.6	83.4	68.4	77.1	76.7	68.9	56.4	83.0	75.3	62.2	85.1	73.0
	VAT	49.1	75.0	78.6	58.4	71.4	72.4	57.0	46.4	78.2	69.4	54.0	82.8	66.1
	APA^u	61.2	80.0	82.4	69.8	78.3	77.4	70.5	57.9	81.9	75.9	63.6	86.1	73.8
	APA^n	62.0	81.2	82.6	71.5	**80.6**	79.3	71.9	60.1	**83.4**	**76.9**	64.2	86.1	75.0
	APA^u+FM	63.1	80.6	82.6	**71.8**	79.7	79.4	71.4	61.5	82.7	75.9	65.5	86.3	75.0
	APA^n+FM	**64.0**	**81.6**	**83.7**	70.9	80.3	**80.3**	**72.8**	62.2	83.0	76.8	65.5	**86.8**	**75.7**
SFDA	SHOT	57.1	78.1	81.5	68.0	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8
	A^{2}Net	58.4	79.0	82.4	67.5	79.3	**78.9**	68.0	56.2	82.9	74.1	60.5	85.0	72.8
	NRC	57.7	**80.3**	82.0	68.1	**79.8**	78.6	65.3	56.4	83.0	71.0	58.6	**85.6**	72.2
	DIPE	56.5	79.2	80.7	70.1	**79.8**	78.8	67.9	55.1	**83.5**	74.1	59.3	84.8	72.5
	APA^u	59.8	76.6	80.4	**71.0**	72.8	74.2	70.2	56.7	80.7	76.5	**61.0**	82.9	71.9
	APA^n	59.3	75.3	81.0	68.3	76.3	76.6	**72.8**	57.5	81.8	76.1	57.9	84.1	72.2
	APA^u+FM	**61.6**	77.5	82.2	69.5	77.8	78.3	70.3	58.0	81.0	75.6	60.7	82.5	72.9
	APA^n+FM	61.2	78.0	**82.9**	70.2	78.2	77.0	71.6	**58.2**	82.3	**76.6**	58.7	82.7	**73.1**

Table 1: Accuracies (%) on Office-Home (ResNet-50).

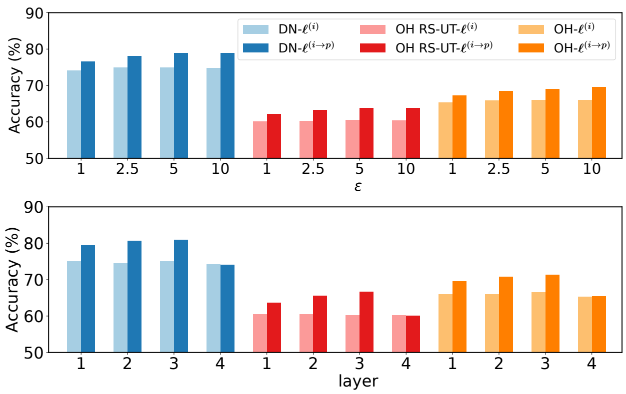

Figure 3: (Upper) Mapping input image perturbations of different magnitudes to penultimate activations. (Lower) Mapping intermediate perturbations at different layers to penultimate activations.

We apply an additional *perturbation projection* step to ensure that $\overline{f(\boldsymbol{x})} + \boldsymbol{r}^{(p_n)}$ lies on the unit ball via

$$\boldsymbol{r}^{(p_n)} \leftarrow \overline{\overline{f(\boldsymbol{x})} + \boldsymbol{r}^{(p_n)}} - \overline{f(\boldsymbol{x})} \tag{9}$$

Approach Analysis

Advantages over Adversarial Training on Input Images or Intermediate Features

Previous works apply adversarial training on input images or intermediate features in domain adaptation. Their performances when using adversarial training alone are unsatisfactory (Liu, Wang, and Long 2021; Lee et al. 2019b; Zhang et al. 2021). When the adversarial perturbation is applied far from the classifier, its impact on model prediction relies on many non-linear transformations in the remaining layers of the deep feature extractor. What is worse, the dimension of input images or intermediate features is usually large (*e.g.*, inputs of $224 \times 224 \times 3$ for a typical computer vision task), making it unfavorable to obtain satisfactory approximations of optimal perturbations from the optimization view.

We show that the perturbation on other layer representations can be mapped into perturbation on the penultimate activations. The mapping maintains the adversarial loss value but leads to higher accuracies.

Let $f = f^b \circ f^a$ be a decomposition of the feature extractor, in which both can be an identity mapping. We use underlines to highlight modules or variables that do not require gradient. When applying adversarial perturbation $\boldsymbol{r}^{(i)}$ on the output of f^a, we have

$$\ell^{(i)}(\boldsymbol{x}) = D\big[g_\sigma(f(\boldsymbol{x})),\ g_\sigma(f^b(f^a(\boldsymbol{x}) + \underline{\boldsymbol{r}^{(i)}}))\big] \tag{10}$$

The perturbation $\boldsymbol{r}^{(i)}$ can be mapped onto penultimate activation as $\boldsymbol{r}^{(i\to p)} \triangleq f^b(f^a(\boldsymbol{x}) + \boldsymbol{r}^{(i)}) - f(\boldsymbol{x})$, and the adversarial loss becomes

$$\ell^{(i\to p)}(\boldsymbol{x}) = D\big[g_\sigma(f(\boldsymbol{x})),\ g_\sigma(f(\boldsymbol{x}) + \underline{\boldsymbol{r}^{(i\to p)}})\big] \tag{11}$$

Comparing Eq. 10 and Eq. 11, their loss values are identical. However, the backpropagation is different. Figure 3 shows that training with Eq. 11 boosts the accuracies by a large margin than Eq. 10. The explanation is that when using $\ell^{(i\to p)}$, it directly refines the penultimate activation $f(\boldsymbol{x})$ of $\boldsymbol{x}$. In contrast, when using $\ell^{(i)}$, it updates the activation of the perturbed sample. Given that many UDA tasks work in a transductive manner where the goal is to make correct prediction for $\boldsymbol{x}$, adversarial training on representations other than penultimate activations is less correlated with this goal.

Interpretation of APA

APA applies adversarial training on penultimate activations to improve prediction confidence. Intuitively, $f(\boldsymbol{x})$ is expected to be moved away from the boundaries. Figure 4 illustrates this idea. $\boldsymbol{r}^{(p)}$ is the adversarial perturbation, $\boldsymbol{\nabla}^{(p)}$

	Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Avg.
UDA	ResNet-101	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4
	DANN	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
	CDAN	85.2	66.9	83.0	50.8	84.2	74.9	88.1	74.5	83.4	76.0	81.9	38.0	73.9
	SAFN	93.6	61.3	84.1	70.6	94.1	79.0	**91.8**	79.6	89.9	55.6	89.0	24.4	76.1
	SWD	90.8	82.5	81.7	70.5	91.7	69.5	86.3	77.5	87.4	63.6	85.6	29.2	76.4
	VAT	93.5	67.0	68.7	72.1	92.4	63.7	88.2	60.4	90.2	68.9	84.9	12.4	71.9
	APA^u	97.1	90.8	84.5	72.8	97.4	96.4	83.9	83.7	**95.8**	87.3	89.8	42.6	85.2
	APA^n	96.9	88.3	83.1	65.9	95.8	95.3	85.7	78.8	94.5	90.1	88.5	46.4	84.1
	APA^u+FM	**97.2**	**92.4**	**89.1**	**77.5**	**97.6**	**97.3**	85.2	**85.2**	95.0	91.9	90.3	39.3	86.5
	APA^n+FM	97.0	90.0	86.4	76.3	97.2	**97.3**	89.4	82.6	94.5	**94.6**	**90.8**	**52.2**	**87.4**
SFDA	SHOT	94.3	88.5	80.1	57.3	93.1	94.9	80.7	80.3	91.5	89.1	86.3	**58.2**	82.9
	A^{2}Net	94.0	87.8	**85.6**	66.8	93.7	95.1	85.8	81.2	91.6	88.2	86.5	56.0	84.3
	HCL	93.3	85.4	80.7	**68.5**	91.0	88.1	**86.0**	78.6	86.6	88.8	80.0	74.7	83.5
	DIPE	95.2	87.6	78.8	55.9	93.9	95.0	84.1	81.7	92.1	88.9	85.4	58.0	83.1
	APA^u	96.2	89.9	83.2	65.7	95.8	95.3	81.9	78.2	91.0	91.5	88.5	44.1	83.4
	APA^n	95.2	88.0	77.3	64.5	93.2	96.2	79.2	77.9	90.7	89.4	88.1	45.8	82.1
	APA^u+FM	96.4	**91.0**	84.4	68.1	**96.7**	**97.6**	83.5	83.1	**93.6**	92.2	**90.1**	51.9	**85.7**
	APA^n+FM	**96.5**	90.6	83.4	64.3	95.8	97.1	82.3	**83.6**	93.3	**93.5**	89.6	55.2	85.4

Table 2: Accuracies (%) on VisDA (ResNet-101).

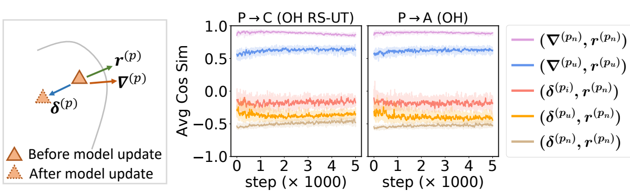

Figure 4: (Left) Illustration of adversarial perturbation, activation gradient, and actual activation change. (Right) Averaged cosine similarity of these variables during the training process of two UDA tasks. (See text for details)

is the gradient of adversarial loss with respect to the penultimate activation, and $\boldsymbol{\delta}^{(p)}$ is the direction where the activation actually moves after network parameters are updated.

In the right of Fig. 4, we show the correlation among these variables during the real training processes of two UDA tasks. In the figure, $\boldsymbol{\nabla}^{(p_u)} = \partial\ell^{(p_u)}(\boldsymbol{x})/\partial f(\boldsymbol{x})$, $\boldsymbol{\nabla}^{(p_n)} = \partial\ell^{(p_n)}(\boldsymbol{x})/\partial \overline{f(\boldsymbol{x})}$, $\boldsymbol{\delta}^{(p_*)} = \overline{f^*(\boldsymbol{x})} - \overline{f(\boldsymbol{x})}$, where $f^*(\boldsymbol{x})$ are the new activations after one-step gradient descend with learning rate 0.001 based on the corresponding adversarial losses. As can be seen, there is a strong positive correlation between adversarial perturbation and active gradient. In particular, $\boldsymbol{\nabla}^{(p_n)}$ is highly close to $\boldsymbol{r}^{(p_n)}$ in direction. $\boldsymbol{\delta}^{(p_*)}$ are negatively correlated with $\boldsymbol{r}^{(p_n)}$. Recall that $\boldsymbol{r}^{(p_n)}$ is the optimal adversarial direction by construction, hence we can claim that the adversarial training indeed pushes target samples away from decision boundaries. Comparing the three variables, $\boldsymbol{\delta}^{(p_n)}$ has the strongest correlation with $\boldsymbol{r}^{(p_n)}$, followed by $\boldsymbol{\delta}^{(p_u)}$ and $\boldsymbol{\delta}^{(p_i)}$. This validates that APA is more effective in improving model prediction confidence than image-based training.

Shrinking Effect of Normalization

APA^u and APA^n have different properties due to different placing of normalization. With a similar mapping trick,

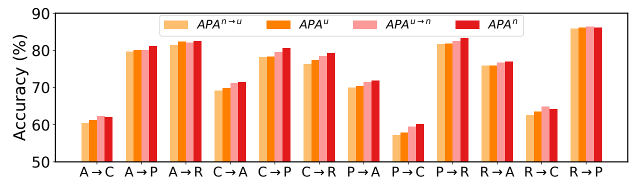

Figure 5: Comparing APA variants on Office-Home.

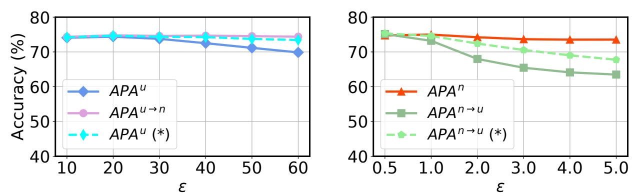

Figure 6: Effects of perturbation magnitude ϵ on Office-Home. (*) Compensated with activation norm ratio.

we can create adversarial losses with identical loss values but different gradients. Let $\boldsymbol{r}^{(p_{n\to u})} \triangleq \boldsymbol{r}^{(p_n)} \cdot \|f(\boldsymbol{x})\|$ and $\boldsymbol{r}^{(p_{u\to n})} \triangleq \overline{f(\boldsymbol{x}) + \boldsymbol{r}^{(p_u)}} - \overline{f(\boldsymbol{x})}$, it is easy to prove that $\ell^{(p_n)}|_{\boldsymbol{r}^{(p_{u\to n})}}$ equals $\ell^{(p_u)}$ and $\ell^{(p_u)}|_{\boldsymbol{r}^{(p_{n\to u})}}$ equals $\ell^{(p_n)}$ in terms of objective value.

Let $\zeta(\boldsymbol{v}) = \boldsymbol{v}/\|\boldsymbol{v}\|_2$ be the ℓ_2 normalization function, $\boldsymbol{x}$ omitted, $\zeta_u \triangleq \zeta(f + \boldsymbol{r}^{(p_u)})$, and $\zeta_n \triangleq \zeta(f) + \boldsymbol{r}^{(p_n)}$, then with some derivations

$$\left(\frac{\partial\ell^{(p_u)}}{\partial f}\right)^\top = \left(\frac{\partial\ell^{(p_u)}}{\partial\zeta_u}\right)^\top \boldsymbol{J}_\zeta(f + \boldsymbol{r}^{(p_u)}) \tag{12}$$

$$\left(\frac{\partial\ell^{(p_n)}}{\partial f}\right)^\top = \left(\frac{\partial\ell^{(p_n)}}{\partial\zeta_n}\right)^\top \boldsymbol{J}_\zeta(f) \tag{13}$$

The Jacobian matrix of $\zeta(\boldsymbol{v})$ is $\boldsymbol{J}_\zeta(\boldsymbol{v}) = (I - \boldsymbol{v}\boldsymbol{v}^\top/(\boldsymbol{v}^\top\boldsymbol{v}))/\|\boldsymbol{v}\|$. Using the mapping trick, we can have $\zeta_n = \zeta_u$ and $\partial\ell^{(p_n)}/\partial\zeta_n = \partial\ell^{(p_u)}/\partial\zeta_u$. Suppose the second term of $\boldsymbol{J}_\zeta$ is negligible, the gradient of f is shrunk by $\|f + \boldsymbol{r}^{(p_u)}\|$ in APA^u, while intact in APA^n. Figure 5

	Method	R→P	R→C	P→R	P→C	C→R	C→P	Avg.
UDA	ResNet-50	69.8	38.3	67.3	35.8	53.3	52.3	52.8
	DANN	71.6	46.5	68.4	38.8	58.8	58.0	56.9
	COAL	73.6	42.6	73.3	40.6	59.2	57.3	58.4
	InstaPBM	75.6	42.9	70.3	39.3	61.8	63.4	58.9
	MDD+IA	76.1	50.0	74.2	45.4	61.1	63.1	61.7
	SENTRY	76.1	56.8	73.6	54.7	**65.9**	64.3	65.3
	CST	79.6	55.6	**77.4**	47.1	61.6	62.0	63.9
	VAT	78.0	45.4	73.0	43.2	60.6	62.1	60.4
	APA^u	80.2	**60.3**	76.1	53.8	65.7	67.0	**67.2**
	APA^n	**80.3**	57.7	76.7	**54.9**	64.1	**67.5**	66.9
SFDA	SHOT	77.0	50.3	75.9	47.0	64.3	64.6	63.2
	APA^u	**77.8**	**55.8**	**76.3**	49.7	**65.0**	**64.7**	**64.9**
	APA^n	77.3	**55.8**	76.2	**50.8**	64.5	64.2	64.8

Table 3: Per-class average accuracies (%) on Office-Home (RS-UT) (ResNet-50).

	Method	R→C	R→P	R→S	C→R	C→P	C→S	P→R	P→C	P→S	S→R	S→C	S→P	Avg.
UDA	ResNet-50	58.8	67.9	53.1	76.7	53.5	53.1	84.4	55.5	60.2	74.6	54.6	57.8	62.5
	DANN	63.4	73.6	72.6	86.5	65.7	70.6	86.9	73.2	70.1	85.7	75.2	70.0	74.5
	COAL	73.8	75.4	70.5	89.6	69.9	71.3	89.8	68.0	70.5	88.0	73.2	70.5	75.9
	InstaPBM	80.1	75.9	70.8	89.7	70.2	72.8	89.6	74.4	72.2	87.0	79.7	71.7	77.8
	SENTRY	83.9	76.7	74.4	90.6	76.0	79.5	90.3	82.9	75.6	90.4	82.4	74.0	81.4
	CST	83.9	78.1	77.5	90.9	76.4	79.7	90.8	82.5	76.5	90.0	82.8	74.4	82.0
	VAT	73.6	74.8	65.2	88.1	66.1	64.5	88.2	71.4	70.7	87.3	77.4	69.9	74.8
	APA^u	85.0	79.7	79.0	91.2	**77.9**	79.3	91.3	82.4	78.5	90.7	84.3	77.7	83.1
	APA^n	**85.3**	**80.4**	**80.2**	**91.8**	77.2	**79.9**	**91.7**	**83.7**	**80.4**	**91.2**	**85.4**	**79.3**	**83.9**
SFDA	SHOT	79.4	75.4	72.8	88.4	74.0	75.5	89.8	77.7	76.2	88.3	80.5	70.8	79.1
	APA^u	84.5	**79.0**	77.3	**90.2**	**78.5**	**78.9**	**91.9**	**84.9**	**77.4**	91.6	**84.7**	**78.7**	**83.1**
	APA^n	**85.4**	78.8	**78.0**	90.1	78.1	78.4	91.3	84.8	76.5	**92.8**	83.6	76.4	82.8

Table 4: Per-class average accuracies (%) on Domainnet (ResNet-50).

compares different APA variants on Office-Home, where APA$^{n\to u}$ means using perturbation $\boldsymbol{r}^{(p_{n\to u})}$ in APA^u. The same is with APA$^{u\to n}$. As can be seen, perturbing normalized activations performs slightly better. Figure 6 shows as ϵ increases (consequently $\|f+\boldsymbol{r}^{(p_u)}\|$ increases), the accuracy of perturbing un-normalized activations drops due to shrunk gradients. This can be compensated by magnifying adversarial loss with activation norm ratio of $\|f+\boldsymbol{r}^{(p_u)}\|/\|f\|$.

Experiments

Setup

Datasets. Office-Home (OH) has 65 classes from four domains: Artistic (A), Clip Art (C), Product (P), and Real-world (R). We use both the original version and the RS-UT (Reverse-unbalanced Source and Unbalanced Target) version (Tan, Peng, and Saenko 2020) that is manually created to have a large label shift. **VisDA-2017** (Peng et al. 2017) is a synthetic-to-real dataset of 12 objects. **DomainNet** (Peng et al. 2019) (DN) is a large UDA benchmark. We use the 40-class version (Tan, Peng, and Saenko 2020) from four domains: Clipart (C), Painting (P), Real (R), Sketch (S).

Baseline methods. We compare our proposed APA with several lines of methods. Methods for vanilla UDA include DANN (Ganin and Lempitsky 2015), CDAN (Long et al. 2018), MDD (Zhang et al. 2019), SWD (Lee et al. 2019a), SAFN (Xu et al. 2019), CST (Liu, Wang, and Long 2021). Methods that handle label shift include COAL (Tan, Peng, and Saenko 2020), InstaPBM (Li et al. 2020a), MDD+IA (Jiang et al. 2020b), SENTRY (Prabhu et al. 2021). SHOT (Liang, Hu, and Feng 2020), A^{2}Net (Xia, Zhao, and Ding 2021), NRC (Yang et al. 2021), DIPE (Wang et al. 2022), HCL (Huang et al. 2021) are methods for source-data free setting. We also compare with self-training based methods using Entropy Minimization, Mutual Information Maximization (MI) (Gomes, Krause, and Perona 2010; Shi and Sha 2012), VAT (Miyato et al. 2018), and FixMatch (FM) (Sohn et al. 2020) loss. The results summarized in Tab. 1–4 are taken from the corresponding papers whenever available.

Implementation details. We implement our methods with PyTorch. The pre-trained ResNet-50 or ResNet-101 (He et al. 2016) models are used as the backbone network of f. Then f contains a bottleneck layer with Batch Normalization. The classification head g is a single Fully-Connected layer. For all tasks, we use batch size 16, $\beta = 0.1$, $\tau = 0.75$, $\epsilon = 30$ for APA^u, $\epsilon = 1.0$ for APA^n, with an only exception on VisDA where we use $\beta = 0.04$ for APA^n instead. More details can be found in the supp. material.

Results

Results on standard benchmarks. Tables 1,2 present evaluation results on three standard benchmarks. Our method achieves consistently the best scores against previous arts under vanilla setting. The performance is further boosted

Projection	R→C	R→P	R→S	C→R	C→P	C→S	P→R	P→C	P→S	S→R	S→C	S→P	Avg.
✓	**85.3**	**80.4**	**80.2**	**91.8**	**77.2**	**79.9**	**91.7**	**83.7**	**80.4**	**91.2**	**85.4**	**79.3**	**83.9**
×	84.1	79.5	79.8	91.3	76.0	**79.9**	90.7	82.0	79.5	90.8	83.6	78.7	83.0

Table 5: Effects of using perturbation projection in APA^n on DomainNet.

when combined with FixMatch (Sohn et al. 2020). For source-data free setting, A^{2}Net (Xia, Zhao, and Ding 2021) is a strong comparison method employing adversarial inference, contrastive matching, and self-supervised rotation. APA^u+FM improves +1.4% over it on VisDA.

Results on label-shifted benchmarks. Tables 3,4 present evaluation results on two benchmarks with a large label distribution shift. Our methods improve over others tailored for this setting by a large margin. Note that we adopt the same hyper-parameters for all tasks, and the performance could likely be further improved by better tuning their values.

Analysis

Effects of activation normalization. As shown in Fig. 7, using activation normalization consistently improves on all tasks. The performance gain is most significant when the domain gap is large (*e.g.*, tasks with low accuracy). This validates that activation normalization can bring source and target distribution closer, thus reduce the impact of accumulated error in self-training.

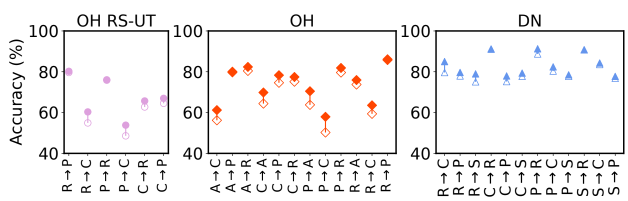

Figure 7: Comparison between using activation normalization (solid markers) and not (hollow markers).

Comparison with other self-training losses. To fairly compare with other self-training methods, we implement them under the same framework and hyper-parameters as APA. The only difference is the loss used for target samples. Results are listed in Tab. 6. Among them, FixMatch and SENTRY require additional random augmented target samples. Our method achieves the best scores on all datasets.

loss	DN	OH RS-UT	OH
ENT	79.00	62.76	70.68
MI	78.38	64.55	69.81
FixMatch	76.35	62.12	69.46
SENTRY	80.31	66.64	73.04
VAT	74.76	60.37	66.05
APA^u	83.08	**67.19**	73.76
APA^n	**83.88**	66.87	**74.98**

Table 6: Comparison with other self-training losses.

Why projecting $\boldsymbol{r}^{(p_n)}$ back to unit sphere? In Eq. 9, we post-process $\boldsymbol{r}^{(p_n)}$ to ensure $\overline{f(\boldsymbol{x})} + \boldsymbol{r}^{(p_n)}$ of unit norm. An alternative way is to add another normalization operation without using perturbation projection by optimizing

$$\ell^{(p_n')}(\boldsymbol{x}) = D\big[g_\sigma(\overline{f(\boldsymbol{x})}), g_\sigma(\overline{\overline{f(\boldsymbol{x})} + \boldsymbol{r}^{(p_n)}})\big] \quad (14)$$

However, this will encounter similar shrinking effect of normalization as discussed. Table 5 shows that using perturbation projection consistently performs better than this strategy on all DomainNet tasks.

Effects of freezing classifier g. In previous experiments, we allow parameters of the classifier g to update during the adaptation stage. Whereas g and the corresponding decision boundaries in the feature space usually change negligibly. Some previous work (Liang, Hu, and Feng 2020) freeze g during adaptation stage. Table 7 shows that the two strategies perform comparably well in our method.

freeze	DN	OH RS-UT	OH
✓	**83.32**	66.34	**73.80**
×	83.08	**67.19**	73.76

Table 7: Effects of freezing parameters of g during the adaptation stage in APA^u.

Sensitivity of hyper-parameters. APA mainly involves two hyper-parameters, the perturbation magnitude ϵ and the adversarial loss weight β. Note that the average norm of penultimate activations is about 30. In Fig. 6, We explicitly present results with large perturbations to show its robustness. Figure 8 plots the effects of β. Our method is insensitive to the ϵ and β within a wide range.

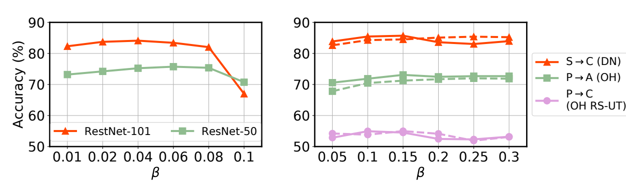

Figure 8: Effects of β in APA^n (solid lines) and APA^u (dashed lines).

Conclusion

This paper explores adversarial training on penultimate activations in UDA. We show its advantage through comparison with adversarial training on input images and intermediate features. Two variants are derived with activation normalization. Extensive experiments on popular benchmarks are conducted to show the superiority of our method over previous arts under both standard and source-data free setting. Our work demonstrates that adversarial training is a strong strategy in UDA tasks.

References

Chen, T.; Cheng, Y.; Gan, Z.; Wang, J.; Wang, L.; Liu, J.; and Wang, Z. 2022. Adversarial Feature Augmentation and Normalization for Visual Recognition. *Transactions on Machine Learning Research*.

Cicek, S.; and Soatto, S. 2019. Unsupervised domain adaptation via regularized conditional alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 1416–1425.

Ganin, Y.; and Lempitsky, V. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 1180–1189.

Gomes, R.; Krause, A.; and Perona, P. 2010. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 23.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Huang, J.; Guan, D.; Xiao, A.; and Lu, S. 2021. Model adaptation: Historical contrastive learning for unsupervised domain adaptation without source data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 3635–3649.

Jeddi, A.; Shafiee, M. J.; Karg, M.; Scharfenberger, C.; and Wong, A. 2020. Learn2perturb: an end-to-end feature perturbation learning to improve adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1241–1250.

Jiang, P.; Wu, A.; Han, Y.; Shao, Y.; Qi, M.; and Li, B. 2020a. Bidirectional Adversarial Training for Semi-Supervised Domain Adaptation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 934–940.

Jiang, X.; Lao, Q.; Matwin, S.; and Havaei, M. 2020b. Implicit Class-Conditioned Domain Alignment for Unsupervised Domain Adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 4816–4827.

Kim, T.; and Kim, C. 2020. Attract, perturb, and explore: Learning a feature alignment network for semi-supervised domain adaptation. In *European Conference on Computer Vision (ECCV)*, 591–607.

Kumar, A.; Sattigeri, P.; Wadhawan, K.; Karlinsky, L.; Feris, R.; Freeman, B.; and Wornell, G. 2018. Co-regularized alignment for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Lee, C.-Y.; Batra, T.; Baig, M. H.; and Ulbricht, D. 2019a. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10285–10295.

Lee, S.; Kim, D.; Kim, N.; and Jeong, S.-G. 2019b. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 91–100.

Li, B.; Wang, Y.; Che, T.; Zhang, S.; Zhao, S.; Xu, P.; Zhou, W.; Bengio, Y.; and Keutzer, K. 2020a. Rethinking distributional matching based domain adaptation. *arXiv preprint arXiv:2006.13352*.

Li, R.; Jiao, Q.; Cao, W.; Wong, H.-S.; and Wu, S. 2020b. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9641–9650.

Liang, J.; Hu, D.; and Feng, J. 2020. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 6028–6039.

Liu, H.; Long, M.; Wang, J.; and Jordan, M. 2019. Transferable adversarial training: A general approach to adapting deep classifiers. In *Proceedings of the International Conference on Machine Learning (ICML)*, 4013–4022.

Liu, H.; Wang, J.; and Long, M. 2021. Cycle self-training for domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34.

Long, M.; Cao, Z.; Wang, J.; and Jordan, M. I. 2018. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1645–1655.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2017. Deep transfer learning with joint adaptation networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2208–2217.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Miyato, T.; Maeda, S.-i.; Koyama, M.; and Ishii, S. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8): 1979–1993.

Pan, S. J.; and Yang, Q. 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10): 1345–1359.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 1406–1415.

Peng, X.; Usman, B.; Kaushik, N.; Hoffman, J.; Wang, D.; and Saenko, K. 2017. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*.

Prabhu, V.; Khare, S.; Kartik, D.; and Hoffman, J. 2021. Sentry: Selective entropy optimization via committee consistency for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 8558–8567.

Seo, M.; Lee, Y.; and Kwak, S. 2021. On the distribution of penultimate activations of classification networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 1141–1151.

Shi, Y.; and Sha, F. 2012. Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Shu, M.; Wu, Z.; Goldblum, M.; and Goldstein, T. 2021. Encoding Robustness to Image Style via Adversarial Feature Perturbations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 28042–28053.

Shu, R.; Bui, H. H.; Narui, H.; and Ermon, S. 2018. A DIRT-T approach to unsupervised domain adaptation. In *International Conference on Learning Representations (ICLR)*.

Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C. A.; Cubuk, E. D.; Kurakin, A.; and Li, C.-L. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 596–608.

Sun, B.; and Saenko, K. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision (ECCV)*, 443–450.

Sun, T.; Lu, C.; and Ling, H. 2022. Prior Knowledge Guided Unsupervised Domain Adaptation. In *European Conference on Computer Vision (ECCV)*, 639–655.

Sun, T.; Lu, C.; Zhang, T.; and Ling, H. 2022. Safe Self-Refinement for Transformer-based Domain Adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7191–7200.

Tan, S.; Peng, X.; and Saenko, K. 2020. Class-Imbalanced Domain Adaptation: an Empirical Odyssey. In *European Conference on Computer Vision (ECCV) Workshops*, 585–602.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7167–7176.

Tzeng, E.; Hoffman, J.; Zhang, N.; Saenko, K.; and Darrell, T. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*.

Wang, F.; Han, Z.; Gong, Y.; and Yin, Y. 2022. Exploring Domain-Invariant Parameters for Source Free Domain Adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7151–7160.

Wang, M.; and Deng, W. 2018. Deep visual domain adaptation: a survey. *Neurocomputing*, 312: 135–153.

Wilson, G.; and Cook, D. J. 2020. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology*, 11(5): 1–46.

Xia, H.; Zhao, H.; and Ding, Z. 2021. Adaptive Adversarial Network for Source-Free Domain Adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 9010–9019.

Xu, R.; Li, G.; Yang, J.; and Lin, L. 2019. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 1426–1435.

Yang, S.; van de Weijer, J.; Herranz, L.; Jui, S.; et al. 2021. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 29393–29405.

Zhang, Y.; Liu, T.; Long, M.; and Jordan, M. I. 2019. Bridging theory and algorithm for domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 97, 7404–7413.

Zhang, Y.; Zhang, H.; Deng, B.; Li, S.; Jia, K.; and Zhang, L. 2021. Semi-supervised models are strong unsupervised domain adaptation learners. *arXiv preprint arXiv:2106.00417*.

Zhu, X. J. 2005. *Semi-supervised learning literature survey*. University of Wisconsin-Madison Department of Computer Sciences.