Adversarial Continuous Learning in Unsupervised Domain Adaptation

Youshan Zhang and Brian D. Davison

Lehigh University, Computer Science and Engineering, Bethlehem, PA, USA
{yoz217,bdd3}@lehigh.edu

Abstract. Domain adaptation has emerged as a crucial technique to address the problem of domain shift, which exists when applying an existing model to a new population of data. Adversarial learning has made impressive progress in learning a domain invariant representation via building bridges between two domains. However, existing adversarial learning methods tend to only employ a domain discriminator or generate adversarial examples that affect the original domain distribution. Moreover, little work has considered confident continuous learning using an existing source classifier for domain adaptation. In this paper, we develop adversarial continuous learning in a unified deep architecture. We also propose a novel correlated loss to minimize the discrepancy between the source and target domain. Our model increases robustness by incorporating high-confidence samples from the target domain. The transfer loss jointly considers the original source image and transfer examples in the target domain. Extensive experiments demonstrate significant improvements in classification accuracy over the state of the art.

Keywords: Adversarial learning · Unsupervised domain adaptation.

1 Introduction

There is a high demand for automatic recognition of multimedia data. The availability of massive labeled training data is a prerequisite for creating machine learning models. Unfortunately, it is time-consuming and expensive to manually annotate data. Therefore, it is often necessary to transfer knowledge from an existing labeled domain to a new unlabeled domain. However, due to the phenomenon of data bias or domain shift [17], machine learning models do not generalize well from an existing domain to a novel unlabeled domain.

Domain adaptation has been a promising method to mitigate the domain shift problem. Existing domain adaptation methods assume that the feature distributions of the source and target domains are different, but share the same label space. These methods either aim to build a bridge between source and target domains [4,35] or identify the shared feature space between two domains [23,11].

Recently, deep neural network methods have achieved great success in domain adaptation problems. Adversarial learning shows its power within deep neural networks to learn feature representations to minimize the discrepancy between

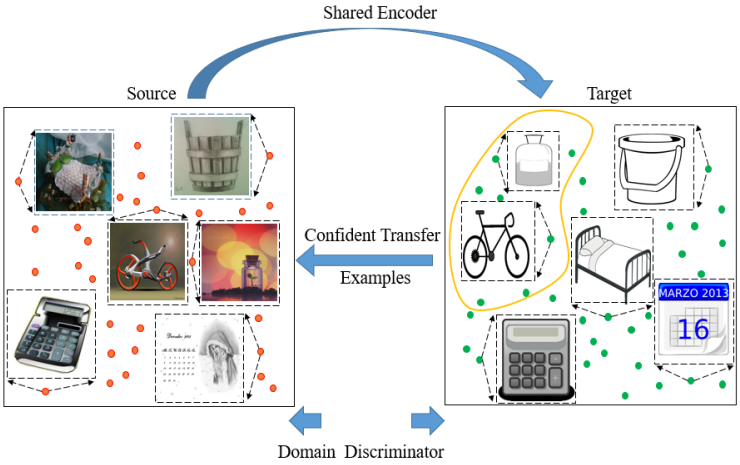


Fig. 1: The scheme of our proposed adversarial continuous learning in unsupervised domain adaptation (ACDA) model. It combines continuous learning and adversarial learning in a two-round classification framework. Initially, the shared encoder will learn a mapping from source images to target images and fool the domain discriminator (which attempts to distinguish examples from source versus target domains). In the second round, the shared encoder will be trained with a new training set which contains the original source images and confidence transfer examples from the target domain, resulting in an improved mapping. The yellow circle marks confident transfer examples.

the source and target domains [22,9]. These methods were inspired by the generative adversarial network (GAN) [5]. Adversarial learning also contains a feature extractor and a domain discriminator. The domain discriminator aims to distinguish the source domain from the target domain, while the feature extractor aims to learn domain-invariant representations to fool the domain discriminator [12,34,9]. The target domain risk is minimized via minimax optimization.

Although adversarial learning achieves remarkable results in domain adaptation, it still suffers from two challenges: (1) although the feature extractor is well trained on the source domain, its applicability to the target domain is lower; that is, the joint distributions of the two domains are not perfectly aligned; and, (2) generating proper transfer examples during training has not been well explored; such examples should enhance the positive transfer and alleviate the negative transfer, and these transfer examples should not affect the distributions of the original domains.

To address the above challenges, we take advantage of the source classifier, and generate transfer examples from the target domain, and then adversarially learn them during the second training. In this two-round paradigm, the feature extractor is not only trained with source data, but also the positive samples from the target domain. In addition, the transferred examples are adversarially learned during the training.

In this paper, we employed a two-round paradigm, and utilize five loss functions in one framework: classification loss, adversarial domain discrepancy loss,

deep correlation loss, transfer loss and domain alignment loss. By aggregating these loss functions, our model can reduce the domain shift effect and thus enhance transferability from the source domain to the target domain. The scheme of our model is shown in Fig. 1.

Our principal contributions are three-fold:

1. We propose a novel method: adversarial continuous learning in unsupervised domain adaptation (ACDA). The proposed ACDA model adversarially learns high confidence examples from the target domain and confuses the domain discriminator;
2. We are the first to propose a deep correlation loss to help ensure that predictions are locally consistent (with those of nearby examples);
3. To better represent the learned features and train a robust classifier, we dynamically align both marginal and conditional distributions of source and target domains in a two-level domain alignment setting.

Extensive experiments on three highly competitive benchmark image datasets show that our ACDA model can significantly improve the classification accuracy over the state of the art.

2 Related work

Domain adaptation has emerged as a prominent method to address the domain shift problem. Recently, adversarial learning models have been found to be a better mechanism for identifying invariant representations in domain adaptation.

Adversarial learning based models aim to define a domain confusion objective to identify the domains via a domain discriminator, and the feature extractor fools the discriminator [22,31]. The target domain risk is minimized via playing the minimax game. The domain-adversarial neural network (DANN) considers a minimax loss to integrate a gradient reversal layer to promote the discrimination of source and target domains [2]. The adversarial discriminative domain adaptation (ADDA) method uses an inverted label GAN loss to split the source and target domain, and features can be learned separately [22]. Zhang et al. [28] reweighed the target samples using the degree of confusion between source and target domains. The target samples are assigned by higher weights, which can confuse the domain discriminator. Miyato et al. incorporated virtual adversarial training (VAT) in semi-supervised contexts to smooth the output distributions as a regularization of deep networks [16]. Later, virtual adversarial domain adaptation (VADA) improved adversarial feature adaptation using VAT and harnessed the cluster assumption (decision boundaries cannot cross high-density data regions). It generated adversarial examples against the source classifier and adapted on the target domain [20]. Different from VADA methods, transferable adversarial training (TAT) adversarially generates transferable examples that fit the gap between source and target domain, yet these examples can affect original distributions of two domains [9].

3 Methods

3.1 Motivation

Existing domain adaptation theory shows that the risk in the target domain can be minimized by bounding the source risk and discrepancy between source and target domains (Theorem 1, from Ben-David et al. [1]). Inspired by GAN [5], adversarial learning [22,9] is designed to reduce the discrepancy between two domains. However, Tzeng et al.'s adversarial discriminative domain adaptation (ADDA) model does not generate any examples during training, and it produces hypotheses that require too large of an adaptability correction [22]. Although Liu et al.'s transferable adversarial training (TAT) model generates adversarial examples based on source classifier and domain discriminator, the generated examples are not real examples in either domain, and it still affects the distributions of both domains [9]. In contrast, we choose to identify high-confidence examples from the target domain and use them to help train the classifier. This not only confuses the domain discriminator, but also deceives the source classifier to push the decision boundary towards the target domain without changing the original data distributions.

Theorem 1 [1] *Let θ be a hypothesis; $\epsilon_s(\theta)$ and $\epsilon_t(\theta)$ represent the source and target risk, respectively.*

$$\epsilon_t(\theta) \leq \epsilon_s(\theta) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}}) + \lambda \tag{1}$$

where $d_{\mathcal{H}}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}})$ is the $\mathcal{H}$-divergence of source and target domains, λ is the adaptability to quantify the error in hypothesis space of two domains, which should be sufficiently small.

3.2 Problem and notation

For unsupervised domain adaptation, given a source domain $\mathcal{D}_{\mathcal{S}} = \{\mathcal{X}_{\mathcal{S}i}, \mathcal{Y}_{\mathcal{S}i}\}_{i=1}^{n_s}$ of n_s labeled samples in C categories and a target domain $\mathcal{D}_{\mathcal{T}} = \{\mathcal{X}_{\mathcal{T}j}\}_{j=1}^{n_t}$ of n_t samples without any labels ($\mathcal{Y}_{\mathcal{T}}$ for evaluation only). Our ultimate goal is to learn a classifier f under a feature extractor G, that provides lower generalization error in target domain.

In this paper, we present our approach: adversarial continuous learning for domain adaptation (ACDA). It incorporates two paradigms: it selects high-confidence examples from the target domain for transferring to the training of the source classifier, and then adversarially trains those high-confidence transfer examples together with the original labeled source and unlabeled target domains, while the number of categories (C) is the same as during training.

3.3 Source classifier

The task in the source domain is trained using the cross-entropy loss in Eq. 2:

$$\mathcal{L}_{\mathcal{S}}(f(G(\mathcal{X}_{\mathcal{S}})), \mathcal{Y}_{\mathcal{S}}) = -\frac{1}{n_s} \sum_{i=1}^{n_s} \sum_{c=1}^{C} \mathcal{Y}_{\mathcal{S}ic} \log(f_c(G(\mathcal{X}_{\mathcal{S}i}))), \tag{2}$$

where $\mathcal{Y}_{\mathcal{S}ic} \in [0,1]^C$ is the probability of each class c among true labels, and $f_c(G(\mathcal{X}_{\mathcal{S}i}))$ is the predicted probability of each class.

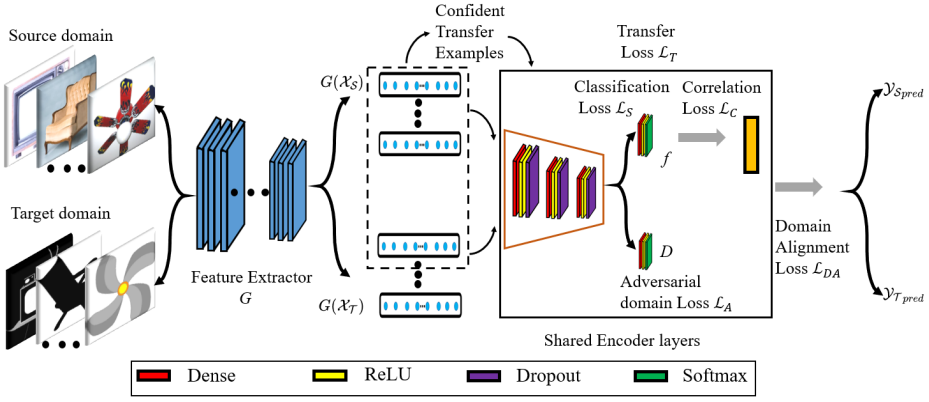


Fig. 2: The architecture of our proposed ACDA model. We first extract features from both source and target domains via G. In first round classification, the shared encoder layers are trained with examples from the labeled source domain and the unlabeled target domain. In the second round, the shared encoder layers are trained with additional high-confidence transfer examples with labels from the first round classifier in the target domain. The domain alignment loss reduces the difference of the marginal and conditional distributions between source and target domains. The red outline highlights the shared layers for both classifier f and domain discriminator D. $\mathcal{Y}_{\mathcal{S}pred}$ and $\mathcal{Y}_{\mathcal{T}pred}$ are the predicted labels of the source and target domains after performing domain distribution alignment.

3.4 Adversarial domain loss

Given the feature representation of feature extractor G, we can learn a discriminator D, which can distinguish between the two domains in Eq. 3.

$$\mathcal{L}_{\mathcal{A}}(G(\mathcal{X}_{\mathcal{S}}), G(\mathcal{X}_{\mathcal{T}})) = -\frac{1}{n_s}\sum_{i=1}^{n_s}\log(1 - D(G(\mathcal{X}_{\mathcal{S}i}))) - \frac{1}{n_t}\sum_{j=1}^{n_t}\log(D(G(\mathcal{X}_{\mathcal{T}j}))) \tag{3}$$

3.5 Deep correlation loss

Our features are extracted from a well-trained model using the last fully connected layer; we assume that two highly similar examples should belong to the same class:

$$\mathcal{Y}_m = \mathcal{Y}_n \text{ if } Sim(G(\mathcal{X}_m),\ G(\mathcal{X}_n)) > Sim(G(\mathcal{X}_m),\ G(\mathcal{X}_{n\neq m})), \tag{4}$$

where $\mathcal{X}_m$ and $\mathcal{X}_n$ are samples from the same domain. Sim is the cosine similarity. We then rank all similarity scores and calculate the top-K cosine similarity matrix for both source and target domains. Hence, we can compare correlated labels with the predicted labels from the source classifier. The loss is defined as:

$$\mathcal{L}_{\mathcal{C}}(\mathcal{Y}_{pred}, \mathcal{Y}_{corr}) = \frac{1}{n_{s/t}}\sum_{i/j=1}^{n_{s/t}} ||\mathcal{Y}_{pred_{i/j}} - M(\mathcal{Y}_{pred}[\mathcal{Y}_{corr_{i/j}}])||, \tag{5}$$

where $\mathcal{Y}_{pred}$ is the prediction of either source domain or target domain from the first round source classifier, and $\mathcal{Y}_{corr}$ is the correlation label with the size of $n_{s/t} \times K$; it shows the top-K index, which is highly related to the instance that should be in the same class. $\mathcal{Y}_{pred}[\mathcal{Y}_{corr}]$ is the updated matrix for the predicted labels and $M(\cdot)$ selects the most frequent labels in the updated matrix as shown in Fig. 3. Therefore, the loss measures how different the predicted label is to its nearest neighbors, which are from the same domain. Notice that this loss can be applied in both source and target domains for inference since we are able to calculate the top-K correlated samples in each domain.

	$\mathcal{Y}_{pred}$	$\mathcal{Y}_{corr}$ (Top-3)	$\mathcal{Y}_{pred}[\mathcal{Y}_{corr}]$	$M(\mathcal{Y}_{pred}[\mathcal{Y}_{corr}])$
$\mathcal{X}_1$	0	2 3 4	1 0 0	0
$\mathcal{X}_2$	1	1 3 4	0 0 0	0
$\mathcal{X}_3$	0	1 2 4	0 1 0	0
$\mathcal{X}_4$	0	5 1 3	1 0 0	0
$\mathcal{X}_5$	1	4 1 3	0 0 0	0

Fig. 3: An example of top-3 correlation labels in updating predicted labels. Given five examples ($\mathcal{X}_1$ to $\mathcal{X}_5$), the prediction is the $\mathcal{Y}_{pred}$, which is from classifier f. The predictions of five examples are within two classes (0 and 1), and the true labels are all zeros. There are two incorrectly predicted results ($\mathcal{Y}_{pred_2}$ and $\mathcal{Y}_{pred_5}$). The top-3 $\mathcal{Y}_{corr}$ shows the top 3 instances that should be in the same class (e.g., $\mathcal{X}_1$ should have the same class as $\mathcal{X}_2$, $\mathcal{X}_3$ and $\mathcal{X}_4$.). The $M(\mathcal{Y}_{pred}[\mathcal{Y}_{corr}])$ changes the predicted labels and is the same as the truth label.

3.6 Continuous learning

The purpose of continuous learning in unsupervised domain adaptation is to bring high-confidence transfer examples (positive samples) from the target domain to the source domain with a high probability threshold. The high probability ensures fewer negative samples are from the target domain, which causes negative transfer in the source classifier (f). In the first round prediction, we get $\mathcal{Y}^{'}_{\mathcal{T}_p}$ of $\mathcal{X}_\mathcal{T}$ ($\mathcal{Y}^{'}_{\mathcal{T}_p}$ is the probability representation of prediction $\mathcal{Y}^{'}_{\mathcal{T}}$). The high-confidence transfer examples $\{\mathcal{X}^{'}_{\mathcal{T}_k}\}_{k=0}^{n_k}$, where $0 \leq n_k \leq n_t$, and n_k is determined by the hyper-parameter $\mathcal{P}$. Therefore, we have the following definition.

Definition: one sample in target domain is considered a high-confidence transfer example if and only if its prediction probability in the dominant class $\mathcal{Y}^{'}_{\mathcal{T}_{p_k}} \geq \mathcal{P}$. In the continuous learning setting, the new source domain consists of original source data plus high-confidence transfer examples from the target domain $\mathcal{X}'_\mathcal{S} = \mathcal{X}_\mathcal{S} + \mathcal{X}^{'}_\mathcal{T}$ with its new labels $\mathcal{Y}'_\mathcal{S} = \mathcal{Y}_\mathcal{S} + \mathcal{Y}^{'}_\mathcal{T}$.

Similar to training in the first round classification, we also train the new source domain via adversarial learning (in which high-confidence transfer examples effectively belong to both the target and new source domains); hence the transfer loss includes three new loss functions in the black box of Fig. 2.

$$\begin{aligned}\mathcal{L}_\mathcal{T}(\mathcal{X}'_\mathcal{S}, \mathcal{Y}'_\mathcal{S}, \mathcal{X}_\mathcal{T}, \mathcal{Y}_{\mathcal{T}corr}) &= \mathcal{L}'_\mathcal{S}(f(G(\mathcal{X}'_\mathcal{S})), \mathcal{Y}'_\mathcal{S}) \\ &+ \mathcal{L}_\mathcal{C}(\mathcal{Y}'_{\mathcal{S}pred}, \mathcal{Y}'_{\mathcal{S}corr}) + \mathcal{L}_\mathcal{A}(G(\mathcal{X}'_\mathcal{S}), G(\mathcal{X}_\mathcal{T})),\end{aligned} \tag{6}$$

where $\mathcal{L}'_{\mathcal{S}}(f(G(\mathcal{X}'_{\mathcal{S}})), \mathcal{Y}'_{\mathcal{S}}) = -\frac{1}{n'_s}\sum_{i=1}^{n'_s}\sum_{c=1}^{C}\mathcal{Y}'_{\mathcal{S}ic}\log\ (f(G(\mathcal{X}'_{\mathcal{S}_i})))$; $\mathcal{Y}'_{\mathcal{S}pred}$ and $\mathcal{Y}'_{\mathcal{S}corr}$ are predicted and correlated labels of the new source domain, respectively. The generated high-confidence transfer examples are together trained with original source data and new domain discrepancy is minimized in Eq. 7.

$$\mathcal{L}_{\mathcal{A}}(G(\mathcal{X}'_{\mathcal{S}}), G(\mathcal{X}'_{\mathcal{T}})) = -\frac{1}{n'_s}\sum_{i=1}^{n'_s}\log(1 - D(G(\mathcal{X}'_{\mathcal{S}_i}))) - \frac{1}{n_t}\sum_{j=1}^{n_t}\log(D(G(\mathcal{X}_{\mathcal{T}j}))) \tag{7}$$

The discriminator also distinguishes between the new source domain and original target domain; the high-confidence transfer examples will further confuse the domain discriminator.

$$\mathcal{L}_{\mathcal{C}}(\mathcal{Y}'_{\mathcal{S}pred}, \mathcal{Y}'_{\mathcal{S}corr}) = \frac{1}{n'_s}\sum_{i=1}^{n'_s}||\mathcal{Y}'_{\mathcal{S}pred_i} - M(\mathcal{Y}'_{\mathcal{S}pred}[\mathcal{Y}'_{\mathcal{S}corr_i}])|| \tag{8}$$

The deep correlation loss will also be minimized in the new source domain and original target domain. We adversarially generate the high-confidence transfer examples from the target domain, which not only maintains the distributions of two domains but also against both source classifier and domain discriminator. As aforementioned, we can minimize target domain risk via bounding the source risk and discrepancy between the source and target domains [1]. In the first round classification, we have a labeled source domain and unlabeled target domain. In the second round classification, we have a labeled new source domain (labels for high-confidence transfer examples are from the prediction of the source classifier), and unlabeled target domain. Hence, the source classifier trains with additional high-confidence transfer examples from the target domain; these examples will push the decision boundary of source classifier towards the target domain. Therefore, the target domain risk will be further reduced.

3.7 Shared encoder layers

The shared encoder layers begin with three repeated blocks and each block has a Dense layer, a "Relu" activation layer, and a Dropout layer. The numbers of units of the dense layer are 512, 128, and 64, respectively. The rate of the Dropout layer is 0.5. It ends with a Dense layer (the number of units is the number of classes in each dataset (10, 31, and 65 in our experiments)).

3.8 A two-level dynamic distribution alignment

After the two-round classification, we can get the prediction of the target domain. However, we can further improve the predicted accuracy by employing a two-level dynamic distribution alignment, which can dynamically update the predicted labels in the target domain.

Manifold Embedded Distribution Alignment (MEDA), proposed by Wang et al. [25], aligns learned features from manifold learning. However, Zhang et

al. showed that there are defects in estimating the geodesic of sub-source and sub-target domains [35]. We modified the domain alignment loss as follows:

$$\begin{aligned}\mathcal{L}_{\mathcal{DA}}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}}) = \arg\min(&\mathcal{L}_{\mathcal{G}}(f(G(\mathcal{X}_{\mathcal{S}})), \mathcal{Y}_{\mathcal{S}}) + \eta||f||^2 \\ &+ \lambda\overline{D_f}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}}) + \rho R_f(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}}))\end{aligned} \tag{9}$$

where f is the classifier from the shared encoder, $\mathcal{L}_{\mathcal{G}}$ is the sum of squares loss; $||f||^2$ is the squared norm of f; and the first two terms minimize the structure risk of shared encoder. $\overline{D_f}(\cdot, \cdot)$ represents the dynamic distribution alignment; $R_f(\cdot, \cdot)$ is a Laplacian regularization; η, λ, and ρ are regularization parameters. Specifically, $\overline{D_f}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}}) = (1-\mu)D_f(P_{\mathcal{S}}, P_{\mathcal{T}}) + \mu\sum_{c=1}^{C} D_f^c(Q_{\mathcal{S}}, Q_{\mathcal{T}})$, where μ is an adaptive factor to balance the marginal distribution $(P_{\mathcal{S}}, P_{\mathcal{T}})$, and conditional distribution $(Q_{\mathcal{S}}, Q_{\mathcal{T}})$, and $c \in \{1, \cdots, C\}$ is the class indicator [25].

For overall training procedures, we first train the labeled source examples and unlabeled target examples using Equations 2, 3, and 5 in the first round classification. We then train the labeled new source domain and unlabeled target domain using Eq. 6 in the second round classification. Finally, we perform domain distribution alignment using Eq. 9.

3.9 The overall training objective function

The architecture of our proposed ACDA model is shown in Fig. 2. Taken together, our model minimizes the following objective function:

$$\begin{aligned}\mathcal{L}(\mathcal{X}_{\mathcal{S}}, \mathcal{Y}_{\mathcal{S}}, \mathcal{X}_{\mathcal{T}}, \mathcal{Y}_{\mathcal{T}corr}) = &\arg\min \alpha\mathcal{L}_{\mathcal{S}}(f(G(\mathcal{X}_{\mathcal{S}})), \mathcal{Y}_{\mathcal{S}}) + (1-\alpha)\mathcal{L}_{\mathcal{C}}(\mathcal{Y}_{pred}, \mathcal{Y}_{corr}) \\ &+ \beta\mathcal{L}_{\mathcal{A}}(G(\mathcal{X}_{\mathcal{S}}), G(\mathcal{X}_{\mathcal{T}})) + \mathcal{L}_{\mathcal{T}}(\mathcal{X}'_{\mathcal{S}}, \mathcal{Y}'_{\mathcal{S}}, \mathcal{X}_{\mathcal{T}}, \mathcal{Y}_{\mathcal{T}corr}) + \mathcal{L}_{\mathcal{DA}}(\mathcal{D}_{\mathcal{S}}, \mathcal{D}_{\mathcal{T}})\end{aligned} \tag{10}$$

where f is the classifier from the shared encoder module; $\mathcal{L}_{\mathcal{S}}$ is the classification loss, which is the typical cross-entropy loss in Eq. 2; $\mathcal{Y}_{pred}$ is the predicted label for the target domain and $\mathcal{Y}_{corr}$ is the correlated label, which shows the K most highly related instances ($\mathcal{Y}_{\mathcal{T}corr}$ is the correlated label for the target domain). The $\mathcal{L}_{\mathcal{C}}$, $\mathcal{L}_{\mathcal{T}}$ and $\mathcal{L}_{\mathcal{DA}}$ represent the correlation loss, transfer loss, and the domain alignment loss, respectively. α and β are balance factors. $\{\mathcal{X}'_{\mathcal{S}}, \mathcal{Y}'_{\mathcal{S}}\}$ is the new source domain.

4 Experiments

In this section, we show how the ACDA model can enhance image recognition accuracy. Our model are evaluated using three public image datasets: Office + Caltech-10, Office-31 and Office-Home [19,25]. These datasets are widely used in many publications [6,4,25], and are the benchmarking data for evaluating the performance of domain adaptation algorithms by training in one domain and testing on another. In addition, we conduct ablation studies to investigate the impact of each component in the ACDA model[1].

[1] Source code is available at https://github.com/YoushanZhang/Transfer-Learning/tree/main/Code/Deep/ACDA.

4.1 Datasets

Office + Caltech-10 [4] consists of 2,533 images in four domains: Amazon (A), Webcam (W), DSLR (D) and Caltech (C) in ten classes. In experiments, C→A represents learning knowledge from domain C which is applied to domain A. There are twelve tasks in Office + Caltech-10 dataset. **Office-31** [19] is another benchmark dataset for domain adaptation, consisting of 4,110 images in 31 classes from three domains: Amazon (A), Webcam (W), and DSLR (D). We evaluate all methods across all six transfer tasks. **Office-Home** [24] contains 15,588 images from 65 categories. It has four domains: Art (Ar), Clipart (Cl), Product (Pr) and Real-World (Rw). There are also twelve tasks in this dataset.

4.2 Implementation details

We extract features for the three datasets from a fine-tuned Resnet50 network [7]. We add one more Dense layer and the number of units is the same as the number of classes in each dataset. The 1,000 features are then extracted from the second-to-last fully connected layer [29,33]. Parameters in domain distribution alignment are $\eta = 0.1$, $\lambda = 10$, and $\rho = 10$, which are fixed based on previous research [25]. $\alpha = 0.5$, $\beta = 0.2$, $K = 3$, $\mathcal{P} = 0.9999$, batch size = 32 and number of iterations = 1000 are determined by performance on the source domain. We also compare our results with 19 state-of-the-art methods (including both traditional methods and deep neural networks).

4.3 Results

The performance on Office + Caltech-10, Office-Home and Office-31 are shown in Tables 1-3. For a fair comparison, we highlight in bold those methods that are re-implemented using our extracted features. Our ACDA model outperforms all state-of-the-art methods in terms of average accuracy (especially in the Office-Home dataset). It is compelling that our ACDA model substantially enhances the classification accuracy on difficult adaptation tasks (e.g., D→A task in the Office-31 dataset and the challenging Office-Home dataset (which has a larger number of categories and different domains are visually dissimilar)). In addition,

Table 1: Accuracy (%) on Office + Caltech-10 dataset

Task	C→A	C→W	C→D	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W	**Ave.**
GFK [4]	94.6	94.9	95.5	92.6	90.5	94.3	93.5	95.7	**100**	93.6	95.9	98.6	95.0
GSM [35]	96.0	95.9	96.2	**94.6**	89.5	92.4	94.1	95.8	**100**	93.9	95.1	98.6	95.2
TJM [13]	94.7	86.8	86.6	83.6	82.7	76.4	88.2	90.9	98.7	87.4	92.5	98.3	88.9
JGSA [27]	95.1	97.6	96.8	93.9	94.2	96.2	95.5	95.9	**100**	94.0	96.3	99.3	96.2
ARTL [10]	**96.3**	94.9	96.2	93.9	98.3	97.5	94.7	**96.7**	**100**	**94.4**	96.2	99.7	96.6
MEDA [25]	**96.3**	98.3	96.2	**94.6**	99.0	**100**	**94.8**	96.6	**100**	93.6	96.0	99.3	97.0
DAN [11]	92.0	90.6	89.3	84.1	91.8	91.7	81.2	92.1	**100**	80.3	90.0	98.5	90.1
DDC [23]	91.9	85.4	88.8	85.0	86.1	89.0	78.0	83.8	**100**	79.0	87.1	97.7	86.1
DCORAL [21]	89.8	97.3	91.0	91.9	**100**	90.5	83.7	81.5	90.1	88.6	80.1	92.3	89.7
RTN [14]	93.7	96.9	94.2	88.1	95.2	95.5	86.6	92.5	**100**	84.6	93.8	99.2	93.4
MDDA [18]	93.6	95.2	93.4	89.1	95.7	96.6	86.5	94.8	**100**	84.7	94.7	99.4	93.6
ACDA	96.2	**100**	**100**	93.9	**100**	**100**	93.9	96.2	**100**	93.9	**96.7**	**100**	**97.6**

Table 2: Accuracy (%) on Office-Home dataset (A: Ar, C:Cl, R: Rw, P: Pr)

Task	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	**Ave.**
GFK [4]	40.0	66.5	71.8	56.8	66.4	65.1	58.1	43.0	74.1	65.3	44.9	76.3	60.7
GSM [35]	49.4	75.5	80.2	62.9	70.6	70.3	65.6	50.0	80.8	72.4	50.4	81.6	67.5
TJM [13]	50.0	60.1	61.3	42.9	60.0	57.3	38.6	34.7	63.3	48.7	38.0	67.2	51.8
JGSA [27]	45.8	73.7	74.5	52.3	70.2	71.4	58.8	47.3	74.2	60.4	48.4	76.8	62.8
ARTL [10]	56.5	80.4	81.4	68.7	81.6	81.7	70.1	56.2	83.3	72.8	58.2	85.6	73.0
MEDA [25]	49.1	75.6	79.1	66.7	77.2	75.8	68.2	50.4	79.9	71.9	53.2	82.0	69.1
DAN [11]	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DANN [3]	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
JAN [15]	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
CDAN-RM [12]	49.2	64.8	72.9	53.8	62.4	62.9	49.8	48.8	71.5	65.8	56.4	79.2	61.5
CDAN-M [12]	50.6	65.9	73.4	55.7	62.7	64.2	51.8	49.1	74.5	68.2	56.9	80.7	62.8
TADA [26]	53.1	72.3	77.2	59.1	71.2	72.1	59.7	53.1	78.4	72.4	**60.0**	82.9	67.6
SymNets [34]	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
MDA [30]	54.8	81.2	82.3	71.9	82.9	81.4	71.1	53.8	82.8	75.5	55.3	86.2	73.3
ACDA	**58.4**	**83.1**	**84.4**	**74.6**	**84.1**	**83.5**	**74.4**	**58.7**	**85.2**	**77.2**	59.3	**87.5**	**75.9**

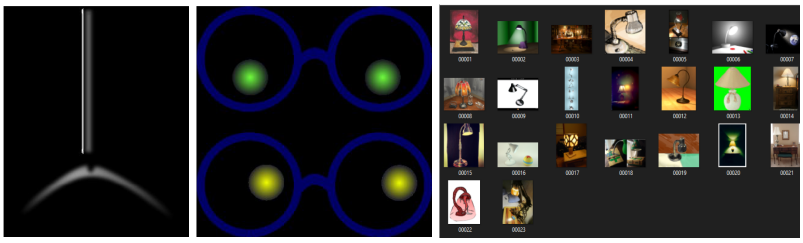
Table 3: Accuracy (%) on Office-31 dataset

Task	A→W	A→D	W→A	W→D	D→A	D→W	**Ave.**
GFK [4]	81.5	82.7	73.5	97.8	72.2	95.3	83.8
GSM [35]	85.9	84.1	75.5	97.2	73.6	95.6	85.3
TJM [13]	87.5	62.2	64.3	97.4	61.3	92.3	77.5
JGSA [27]	89.1	91.0	77.9	**100**	77.6	98.2	89.0
ARTL [10]	90.9	93.0	77.1	99.6	78.2	98.3	89.6
MEDA [25]	91.7	89.2	77.2	97.4	76.5	96.2	88.0
JAN [15]	85.4	84.7	70.0	99.8	68.6	97.4	84.3
TADA [26]	94.3	91.6	73.0	99.8	72.9	98.7	88.4
SymNets [34]	90.8	93.9	72.5	**100**	74.6	98.8	88.4
SSD [32]	92.7	91.6	77.5	99.4	77.8	98.0	89.4
CAN [8]	94.5	95.0	77.0	99.8	78.0	**99.1**	90.6
ACDA	**95.5**	**96.2**	**80.2**	99.2	**81.1**	98.4	**91.8**

the improvement on the Office + Caltech-10 dataset is not large. This caused by the high state-of-the-art classification accuracy (more than 97%, it is hence difficult to make a significant improvement). However, our model still provides more than 0.6% (absolute) improvement over the best baseline method, which translates to a relative reduction in error of 20%. These experiments demonstrate the efficiency of the ACDA model in aligning both the marginal and conditional distributions of two domains.

4.4 Ablation study

We first consider the effects of three different loss functions on classification accuracy ($\mathcal{L}_{\mathcal{S}}$ and $\mathcal{L}_{\mathcal{A}}$ are required for adversarial learning). Different combinations of loss functions are reported in Tab. 4, in which T represents transfer loss, C, correlation loss, and DA, distribution alignment loss. "ACDA$-$T/C/DA" is implemented without transfer loss, correlation loss, and distribution alignment loss. It is a simple model, which only reduces the source risk without minimizing the domain discrepancy using adversarial learning. "ACDA$-$T/C" directly aligns the joint distribution of the two domains. "ACDA$-$DA" reports results without performing the additional domain distribution alignment. We observe that with the increasing of the number of loss functions, the robustness of our model keeps improving. The usefulness of loss functions is ordered as $\mathcal{L}_{\mathcal{C}} < \mathcal{L}_{\mathcal{DA}} < \mathcal{L}_{\mathcal{T}}$.



(a) Speaker (b) Glass (c) Desk lamp

Fig. 4: Example images to illustrate the importance of probability threshold $\mathcal{P}$. The task is A→W, (a) and (b) is from Clipart domain, and (c) is from Art domain in Office-Home dataset. When $\mathcal{P} = 0.9$, (a) and (b) are mistakenly treated as high-confidence examples in Desk lamp class, and these two examples will be excluded when $\mathcal{P} = 0.99$.

Therefore, the proposed continuous learning approach and correlation loss are effective in improving performance, and different loss functions are helpful and important in minimizing the target domain risk.

We further study the use of high-confidence transfer examples. To show the importance of selecting a high probability threshold $\mathcal{P}$, Fig. 4 shows two unusual examples, in which the task is from Art domain to Clipart domain. In our continuous learning setting, we bring high-confidence transfer examples into target domain (Clipart). Examples (a) and (b) are from the Clipart domain, and if $\mathcal{P} = 0.9$, these two will be treated as high-confidence transfer examples. However, these two are wrongly classified as desk lamps, while the original classes are speaker and glasses. When $\mathcal{P} = 0.9$, these two examples are included. Hence, $\mathcal{P} = 0.9999$ is a sufficient threshold to eliminate these negative examples.

We also show the performance of different $\mathcal{P}$ on classification error rate in Tab. 5 and Fig. 5. In Tab. 5, "−" lists the number of samples in the target domain (e.g., in C→A, the number of samples in A is reported). With the increasing of $\mathcal{P}$, the number of high confidence transfer examples is decreased, and error rate is first decreased and then increased. This phenomenon is induced by either the negative transfer examples are included in the training when $\mathcal{P}$ is small or fewer transfer examples are included in the training when $\mathcal{P}$ is too high. Therefore,

Table 4: Ablation experiments on Office-31 dataset

Task	A→W	A→D	W→A	W→D	D→A	D→W	**Ave.**
ACDA−T/C/DA	88.2	90.6	75.3	97.8	74.4	94.8	86.9
ACDA−T/DA	85.8	91.6	75.6	98.0	76.3	97.4	87.5
ACDA−T/C	91.9	88.5	78.3	98.1	77.8	97.3	88.7
ACDA−C/DA	91.4	91.6	78.3	98.8	78.4	96.9	89.2
ACDA−T	94.5	94.0	77.7	99.0	78.8	97.1	90.2
ACDA−DA	94.3	94.2	79.5	99.6	79.5	97.7	90.8
ACDA−C	95.4	95.1	79.1	99.0	79.8	97.9	91.1
ACDA	**95.5**	**96.2**	**80.2**	**99.2**	**81.1**	**98.4**	**91.8**

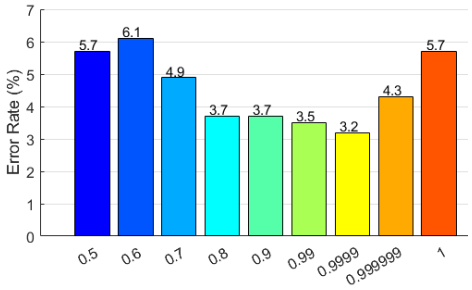


Fig. 5: Error rate of different probability thresholds $\mathcal{P}$ in Office + Caltech 10 dataset (x-axis is different $\mathcal{P}$).

Table 5: The number of high-confidence transfer examples under different $\mathcal{P}$ on Office + Caltech-10 dataset

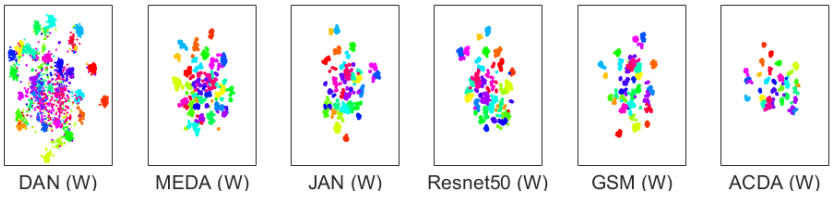
$\mathcal{P}$	C→A	C→W	C→D	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W
–	958	295	157	1123	295	157	1123	958	157	1123	958	295
0.5	956	295	157	1118	295	157	1112	950	157	1114	950	295
0.7	939	282	154	1076	278	150	969	894	157	1049	907	289
0.9	923	278	152	1056	272	147	804	675	156	900	685	273
0.9999	266	194	103	578	175	90	145	364	54	565	751	90
0.9999999	18	104	92	351	27	20	0	0	11	0	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0

$\mathcal{P} = 0.9999$ is the best hyper-parameter value for our model (hyperparameter $\mathcal{P}$ is tuned based on the top-3 correlated labels using the source domain data).

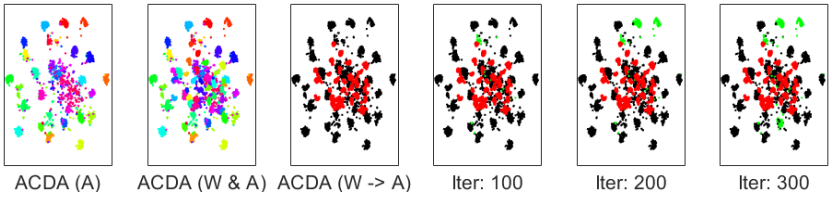
5 Discussion

There are two compelling advantages of the proposed ACDA model. First, we employed two-round classification and adversarially learned high-confidence transfer examples from the target domain. Secondly, our model surpasses the other models across all three datasets even using the same features, which reflects the value of our model on domain adaptation for image recognition. We also present the visualization of the W→A task in the Office-31 dataset to show the progress of our ACDA model. In Fig. 6(a), the feature representation of domain W of different methods is shown. For features of DAN, MEDA, JAN and Resnet50 model, the data distribution is not better aligned since data structures are mixing together. Although the features of the GSM model are slightly better than the first four methods, it is still worse than our features. Fig. 6(b) shows the evolution of selecting high-confidence transfer examples (green color dots) using our ACDA model. As the number of iterations increases, the more high confidence transfer examples are adversarially generated from target domain (A), which will be included in source domain (W). The accuracy in the target can be improved via such an evolution.

To validate the assumption that "two highly correlated examples should belong to the same class" in Sec. 3.5, we further compare the classes and show the matching class accuracy using the top-1 correlated labels in Tab. 6. This demonstrates that the correlated labels are useful in updating predicted labels.



(a) Visualization of different methods on domain W



(b) The evolution of our ACDA model in task W→A

Fig. 6: t-SNE view of task W→A in Office-31 dataset. Different colors in (a) and first two images in (b) represent different categories. In the last four images of (b), black dots represent domain A, red dots represent domain W and green dots are high-confidence examples, which are adversarially learned in ACDA (Iter: iteration).

Table 6: Top-1 correlated label accuracy (%) on three benchmarking datasets.

Domain	Office + Caltech-10				Office-31			Office-Home			
	A	C	D	W	A	D	W	Ar	Cl	Pr	Rw
Correlated	95.7	96.6	100	100	88.4	99.0	99.2	76.7	76.0	93.3	85.0

Unfortunately, performance is not always better than other methods, which is a function of the differences across specific domain adaptation tasks. Furthermore, we use the top-K highly correlated instances to construct the adjacency matrix (top-3 in our experiments); other K could be further explored.

6 Conclusion

We propose novel adversarial continuous learning in unsupervised domain adaptation (termed ACDA) to overcome limitations in generating proper transfer examples and aligning the joint distributions of two domains by minimizing five loss functions. The generated transfer examples can help to further learn the domain invariant of the two domains. As a component of our ACDA model, explicit domain-invariant features are learned through such a cross-domain training scheme. Experiments on three benchmark datasets show the robustness of our proposed ACDA model.

References

1. S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175,

2010.
2. Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.
3. M. Ghifary, W. B. Kleijn, and M. Zhang. Domain adaptive neural networks for object recognition. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pages 898–904. Springer, 2014.
4. B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073. IEEE, 2012.
5. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.
6. R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 999–1006. IEEE, 2011.
7. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
8. G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019.
9. H. Liu, M. Long, J. Wang, and M. Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In *International Conference on Machine Learning*, pages 4013–4022, 2019.
10. J. Long, M.and Wang, G. Ding, S. J. Pan, and S. Y. Philip. Adaptation regularization: A general framework for transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(5):1076–1089, 2013.
11. M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.
12. M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1647–1657, 2018.
13. M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu. Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1410–1417, 2014.
14. M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016.
15. M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 2208–2217. JMLR.org, 2017.
16. T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.
17. S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.
18. M. M. Rahman, C. Fookes, M. Baktashmotlagh, and S. Sridharan. On minimum discrepancy estimation for deep domain adaptation. In *Domain Adaptation for Visual Understanding*, pages 81–94. Springer, 2020.

19. K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proceedings of the European Conference on Computer Vision*, pages 213–226. Springer, 2010.
20. R. Shu, H. H. Bui, H. Narui, and S. Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018.
21. B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Proc. of European Conference on Computer Vision*, pages 443–450. Springer, 2016.
22. E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.
23. E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.
24. H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.
25. J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu. Visual domain adaptation with manifold embedded distribution alignment. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM '18, pages 402–410, 2018.
26. X. Wang, L. Li, W. Ye, M. Long, and J. Wang. Transferable attention for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5345–5352, 2019.
27. J. Zhang, W. Li, and P. Ogunbona. Joint geometrical and statistical alignment for visual domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1859–1867, 2017.
28. W. Zhang, W. Ouyang, W. Li, and D. Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3801–3809, 2018.
29. Y. Zhang, J. P. Allem, J. B. Unger, and T. B. Cruz. Automated identification of hookahs (waterpipes) on instagram: an application in feature extraction using convolutional neural network and support vector machine classification. *Journal of Medical Internet Research*, 20(11):e10513, 2018.
30. Y. Zhang and B. D. Davison. Modified distribution alignment for domain adaptation with pre-trained Inception ResNet. *arXiv preprint arXiv:1904.02322*, 2019.
31. Y. Zhang and B. D. Davison. Adversarial consistent learning on partial domain adaptation of PlantCLEF 2020 challenge. In *CLEF working notes 2020, CLEF: Conference and Labs of the Evaluation Forum*, 2020.
32. Y. Zhang and B. D. Davison. Domain adaptation for object recognition using subspace sampling demons. *Multimedia Tools and Applications*, pages 1–20, 2020.
33. Y. Zhang and B. D. Davison. Impact of ImageNet model selection on domain adaptation. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision Workshops*, pages 173–182, 2020.
34. Y. Zhang, H. Tang, K. Jia, and M. Tan. Domain-symmetric networks for adversarial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5031–5040, 2019.
35. Y. Zhang, S. Xie, and B. D. Davison. Transductive learning via improved geodesic sampling. In *Proceedings of the 30th British Machine Vision Conference*, 2019.