HVCLIP: High-dimensional Vector in CLIP for Unsupervised Domain Adaptation

Noranart Vesdapunt, Kah Kuen Fu, Yue Wu,
Xu Zhang, and Pradeep Natarajan

Amazon
{solves,kahkuen,wuayue,xzhnamz,natarap}@amazon.com

Abstract. Recent advancement in the large-scale image-text pre-training model (such as CLIP) has significantly improved unsupervised domain adaptation (UDA) by leveraging the pre-trained knowledge to bridge the source and target domain gap. However, Catastrophic forgetting still remains to be the main challenge, since traditional fine-tuning method to adjust CLIP model weights on a target domain can quickly override CLIP's pre-trained knowledge. To address the above issue, we propose to convert CLIP's features into high-dimensional vector (hypervector) space to utilize the robustness property of hypervector. We first study the feature dimension size in the hypervector space to empirically find the dimension threshold that allows enough feature patterns to be redundant to avoid excessive training (thus mitigating catastrophic forgetting). To further utilize the robustness of hypervector, we propose Discrepancy Reduction to reduce the domain shift between source and target domains, and Feature Augmentation to synthesize labeled target domain features from source domain features. We achieved the best results on four public UDA datasets, and showed the generalization of our method to other applications (few-shot learning, continual learning). The proposed method also shows model-agnostic property across vision-language and vision backbones.

1 Introduction

Unsupervised Domain Adaptation (UDA) proposes to mitigate the domain shift issue [65] by transferring the knowledge from a labeled source domain to an unlabeled target domain. Past UDA works either focus on reducing the domain shift between source and target domains (discrepancy-based method) [10] and/or synthesizing the labeled target domain from the source domain (synthetic method) [55]. Recent UDA works [2, 19, 20, 50, 58] leverage the large-scale pre-training as a new paradigm for UDA and significantly outperform the previous work. For example, SKD [58] shows that CLIP [41], even without fine-tuning (zero-shot prediction), can achieve +14% accuracy on DomainNet [39]) compared to SSRT [49] and CDTrans [60], which were pre-trained on ImageNet and fine-tuned on UDA dataset. The improvement of CLIP is due to the pre-training on millions or even billions image-text pairs [45]. However, such large-scale pre-training is prone to catastrophic forgetting where the traditional fine-tuning can

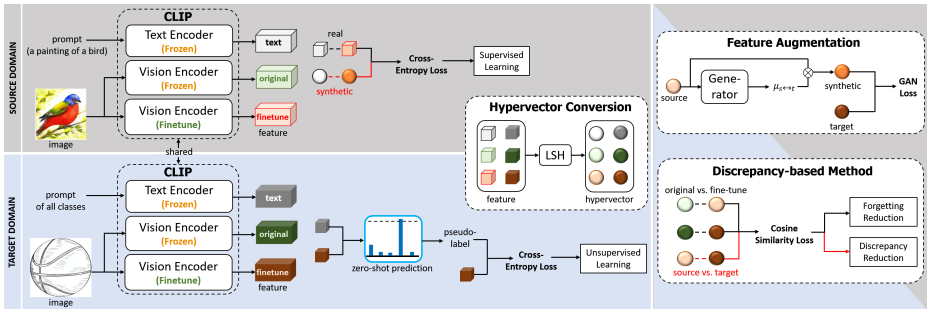

Fig. 1: Overview. For the source domain, we convert labels and domain names into prompts and train CLIP supervisedly to match the image feature to the text feature of the class label. For the target domain, we use zero-shot prediction to obtain pseudo-labels for unsupervised learning. We propose to utilize the robustness of hypervector to perform: Forgetting Reduction (constrain fine-tuned features on original CLIP's prediction features), Discrepancy Reduction (mitigate domain shift between source and target features), and Feature Augmentation (synthesize labeled target domain feature from the source domain).

quickly override CLIP's pre-trained knowledge and decrease the accuracy by half [20].

Mitigating catastrophic forgetting is a requirement for leveraging CLIP in UDA. Most methods [2, 19, 50] chose the simplest way of freezing the vision encoder in CLIP to avoid fine-tuning, and therefore avoiding catastrophic forgetting. However, by carefully adjusting the learning rate (PADCLIP) [20], or using self-knowledge distillation (SKD, use original pre-trained CLIP as a regularization) [58], both methods show +6% accuracy on VISDA-2017 [40] vs. frozen vision encoder. Mitigating catastrophic forgetting comes with a cost, for example, SKD has to re-fine-tune CLIP 5 times, resulting in 5x slower training speed (fine-tune and reset the weight, but keep the pseudo-label from the previous round of fine-tuning), in order to improve pseudo-label quality (higher pseudo-label quality requires less training iterations, and less training iterations leads to less catastrophic forgetting). PADCLIP proposed catastrophic forgetting measurement (CFM) in the image space (by measuring the prediction difference between weak and strong image augmentations) to adjust the learning rate without decreasing the training speed. However, PADCLIP's dependency on image space breaks the assumption of traditional UDA works that operate in the feature space (e.g., the discrepancy-based method minimizes the distance between source and target features, and the synthetic method synthesizes target features from source features).

One of the root causes of catastrophic forgetting is the lack of robustness of the features (a slight change of features after fine-tuning can lead to a different prediction). Instead of remediating the catastrophic forgetting symptom by

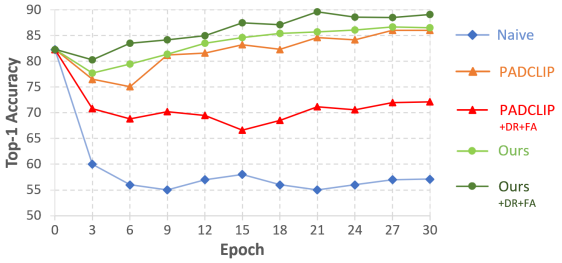

Fig. 2: Catastrophic Forgetting. We naively fine-tune CLIP (ResNet-101) on VisDA-2017 source and target domain training set, and test on the target domain validation set. CLIP forgot the pre-trained knowledge (accuracy -25%), but PADCLIP and our method can mitigate this (+3.7% and +4.2% respectively). However, Discrepancy Reduction (DR) and Feature Augmentation (FA) are not compatible with PADCLIP (-14% due to the image space dependency), while ours is compatible (+2.6%).

adjusting the learning rate or adding regularization, we seek to convert CLIP features into a more robust feature space in the beginning. We propose to fine-tune CLIP in the high-dimensional vector (hypervector) space [13]. Hypervector achieves robustness by introducing redundancy into the feature representation. During the fine-tuning, if CLIP matches the text and image features correctly or makes a small mistake, the hypervectors of these features are likely to be redundant which leads to a small loss (in other words, the loss can only be large when the prediction has a significant error). This is similar to CFM in PADCLIP where CFM aims to lower the learning rate from weak augmentation example (which PADCLIP is likely to predict correctly), and lowering the learning rate (or decreasing loss) will lead to less catastrophic forgetting. The key difference is that hypervector operates in the feature space, so it is compatible with discrepancy-based and synthetic methods, while PADCLIP is not compatible.

We formulate the discrepancy-based and synthetic methods in CLIP by proposing Forgetting Reduction (FR, we minimize the original and fine-tuned CLIP features), Discrepancy Reduction (DR, we minimize the cosine distance between source and target features), and Feature Augmentation (FA, we trained a generator to synthesize labeled target domain features from source domain features) in hypervector space. Fig. 2 shows that PADCLIP's CFM is not effective enough to reduce catastrophic forgetting beyond the traditional fine-tuning, and adding both methods leads to -14% accuracy drops, while ours is compatible (+2.6% accuracy). We achieved the best results on DomainNet [39], VisDA-2017 [40], Office-Home [54], and Office-31 [44]. Our method achieves the best results on CLIP with ResNet-50, ResNet-101, and ViT B/16 backbone. Our method is model-agnostic where it supports a vision-language model (CLIP) and vision models (BiT [17], DeiT [52]). Beyond UDA, we show the effectiveness of catastrophic forgetting mitigation by our method in the continual learn-

ing setting (accuracy +2.4% on CIFAR-100 and +1.3% on TinyImageNet), and few-shot learning setting (+1.5% vs. TIP-Adapter-F [64]).

- We proposed a Forgetting Reduction (FR) with hypervectors in CLIP to mitigate catastrophic forgetting by utilizing the robustness of high-dimensional vectors to regularize the fine-tuned vs. original prediction. We show the evidence by studying the hypervector dimension size vs. the accuracy drops due to catastrophic forgetting.
- We proposed Discrepancy Reduction (DR) and Feature Augmentation (FA) to formulate the traditional UDA methods (discrepancy-based and synthetic methods) in the CLIP setting. Both DR and FA are built on the foundation of our proposed hypervector.
- We achieved the best results on four UDA benchmarks with ResNet and ViT. Beyond UDA, our method supports few-shot learning and continual learning, and our method is model-agnostic (supports both vision-language and vision model).

2 Related Works

Unsupervised Domain Adaptation adapts labeled source domain to unlabeled target domain. Discrepancy-based method reduces the divergence between source and target domains by regularizing them into the same distribution [23, 31, 48], or applying adversarial loss to obtain domain-agnostic features [22, 30, 31, 42, 48, 53]. For example, CAN [14] pushes source and target domain features into the same distribution by a contrastive loss, and DM-ADA [59] mixes up source and target domain features during adversarial training to achieve continuous features regardless of the domain. Data augmentation method synthesizes labeled target domain images/features to augment training data [55,56]. For example, GFCA [11] trains a generator in a few-shot manner to synthesize cross-domain features. Our method enables discrepancy-based method and feature augmentation for CLIP [41] in UDA.

Vision Transformer (ViT) in UDA has an advantage of feature alignment as ViT [8] is more robust to noise than CNN [33]. Recent works [42, 49, 61] use ViT-B/16 to achieve better performance than CNN. CDTrans [60] proposes cross-attention to further align the source and target domain features, and PMTrans [68] proposes a lightweight version of CDTrans by mixing image patches from the source and the target domain. CLIP [41] adds a text encoder to ViT-B/16 to formulate a vision-language model and is the foundation of recent works [2, 19, 20, 50, 58]. Several improvements were proposed on top of CLIP, including, catastrophic forgetting measurement [20] or self-knowledge distillation [58] to mitigate catastrophic forgetting, Prompt Task-dependent Tuning [21] to learn prompt for each domain and class, memory-aware knowledge adaptor to encourage pseudo-label of the confidence classes, and incorporating LLM [2] to broaden the pre-training knowledge. We follow the previous methods above

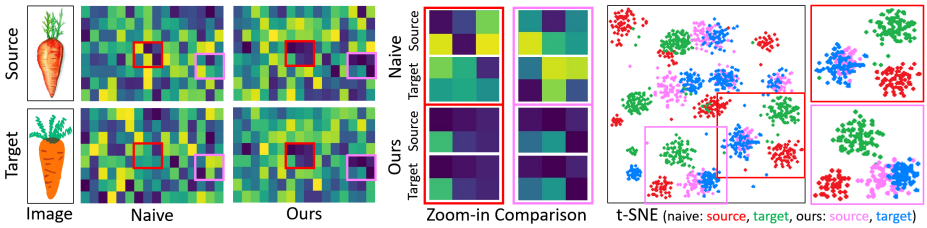

Fig. 3: Feature Visualization. We use Painting→ClipArt from DomainNet to visualize CLIP features from the naively fine-tuned model vs. our method. Our method's feature map has a higher similarity between the source and the target domain, thanks to Discrepancy Reduction and Feature Augmentation, which allows better utilization of labeled source domain knowledge.

to use ViT-B/16, and in theory, our improvement is orthogonal and compatible by converting CLIP features into hypervectors before applying each method.

Pseudo-label in UDA uses a trained model (on labeled source domain) to generate pseudo labels on unlabeled target domain [2,19,20,25,29,49,50,58,60,69]. Consistency regularization on pseudo-labels [16,47,63] ensures the consistency between the perturbed views of each training sample to promote the prediction consistency on the unlabeled training data. Inter-class bias in pseudo-labeling is caused by the prediction error from classes with similar appearance (e.g., dog vs. wolf), and DebiasPL [21] proposed to mitigate inter-class bias by causal inference. PADCLIP follows DebiasPL to introduce pseudo-label into CLIP for UDA. Our method also leverages pseudo-label.

Hypervector is robust to noise due to the high capacity of high-dimensional vector and hypervector is well-studied in Cognitive Science [13]. For Computer Vision, hypervector is used in object recognition [38], visual question answering [36], and scene transformation [15]. Recently, VSAIT [51] proposed to learn a feature mapping in hypervector space, and show the robustness such that hypervector can tolerate the different image contents (e.g., source image contains a tree, but target image contains sky) to perform synthetic image generation. Different from VSAIT, we proposed hypervector for UDA (image classification) to mitigate catastrophic forgetting and enable CLIP fine-tuning with our Discrepancy Reduction and Feature Augmentation.

3 Methodology

3.1 Formulation of CLIP in UDA

Given N_s labeled source domain $D_s = \{x_i^s, y_i^s\}_{i=1}^{N_s}$ and N_t unlabeled target domain $D_t = \{x_i^t\}_{i=1}^{N_t}$, we formulate image-text pairs for CLIP by converting a

label into a prompt in the format of "a [DOMAIN] of a [CLASS]" (e.g., a real photo of a car). The probability of an image belonging to class k is calculated by forwarding image and prompt (τ) from all K classes to CLIP's vision (f) and text encoder (g) and applying softmax:

$$\ddot{p}(\hat{y_i} = k|x_i) = \frac{\exp(\cos(g(\tau_k), f(x_i)/T)}{\sum_{z=1}^{K} \exp(\cos(g(\tau_z), f(x_i)/T)} \tag{1}$$

where T is the temperature parameter learned by CLIP, cos is a cosine similarity [41], and we denote a vector of $\ddot{p}_k$ as p (probability of a sample in a minibatch B). The supervised training loss for a labeled source domain (L_s) is the cross-entropy($\mathbf{H}$) between model prediction and label:

$$L_s = \frac{1}{B} \sum_{i=1}^{B} \mathbf{H}(y_i^s | p(x_i^s)) \tag{2}$$

The unsupervised training loss (L_u) is formulated by using the model's prediction from an unlabeled target domain ($q = p(x^t)$) as a soft pseudo-label to convert into a hard pseudo-label by a one-hot encoder ($\mathbb{1}$), and use a fixed threshold ($\tau = 0.4$) to select high confident pseudo-labels.

$$L_u = \frac{1}{B} \sum_{i=1}^{B} \mathbb{1}[\max(q_i) \geq \tau] \cdot \mathbf{H}(argmax(q_i)|q_i) \tag{3}$$

3.2 Hypervector Space Conversion

We convert CLIP's features into hypervector space with Random Projection (RP) like Local Sensitivity Hashing (LSH) [38]. RP projects and normalizes the features using a random matrix where each row is sampled from a standard normal distribution and normalized to unit length to achieve the output vectors in the range of [-1, 1]. Our hypervector has a dimensionality of 4096 on each CLIP's feature (CLIP's feature size is 512, so the total hypervector size is 512*4096=2.1M). The [-1, 1] vector range and the high dimensionality of our hypervector allow us to use Vector Symbolic Architecture (VSA) to map a hypervector into another hypervector. VSAIT [51] proposed a hypervector mapping by assuming that each image is represented with VSA operations: binding/unbinding ($\otimes$, element-wise multiplication) and bundling (element-wise addition):

$$v^s = c \otimes c^s + d \otimes d^s, \tag{4}$$

where v^s is a source domain hypervector, c and d are hypervectors representing contents of the image (e.g., a car and a dog, (we use 2 terms as an example)) bound with source-specific hypervectors: c^s and d^s. A hypervector mapping ($u^{s \leftrightarrow t}$ is achieved by unbinding source-specific hypervectors and binding target-specific hypervectors (c^t and d^t).

$$u^{s \leftrightarrow tgt} = c^s \otimes c^t + d^s \otimes d^t \tag{5}$$

Since the binding/unbinding operation is invertible ($c \otimes c^s \otimes c^s = c$), and the binding of irrelevant attributes is almost orthogonal ($c^s \otimes d^s \approx 0$, this is shown in the empirical study [38] that hypervector with dimension >=700 (we use 4,096) has this property, and we confirm in our ablation study section), the target domain hypervector (v^t) is defined as:

$$\begin{aligned} v^s \otimes u^{s \leftrightarrow t} &= c \otimes c^s \otimes (c^s \otimes c^t + d^s \otimes d^t) \\ &+ d \otimes d^s \otimes (c^s \otimes c^t + d^s \otimes d^t) \\ &\approx c \otimes c^t + d \otimes d^t \approx v^t \end{aligned} \tag{6}$$

3.3 Feature Augmentation

We augment CLIP's features by converting them into hypervector space with RP (ϕ), and use a generator (G) to synthesize target feature ($\hat{v}_i^t = \phi(f(\hat{x}_i^t))$) with the same label as the input source feature ($v_i^s = \phi(f(x_i^s))$) by learning a hypervector mapping ($u^{s \leftrightarrow t} = G(v_i^s)$):

$$\hat{v}_i^t = v_i^s \otimes G(v_i^s) \tag{7}$$

Our G is a modified ResNet-18 that takes a reshaped hypervector from 4096x512 to 4096x32x16 (WxHxC) as input, and we remove all pooling layers, set the convolutional stride to 1, and add a convolutional layer with kernel size 1x1 at the end to map the output back to the input shape. We chose ResNet-18 for simplicity, and we also use our modified ResNet-18 as a discriminator (D) by changing the last 1x1 convolutional layer to output 2 classes (Fake vs. Real). We train G and D in an adversarial training manner. During the discriminator training, we freeze CLIP and G to train D to classify the synthesized target feature ($\hat{v}_i^t$) as 0 and classify the real target feature (v_i^t) as 1:

$$L_d = max[log(D(v_i^t)) + log(1 - D(\hat{v}_i^t))] \tag{8}$$

During the generator training, we freeze CLIP and D to train G with a discriminative loss and a cycle loss (convert synthesized target features back to source features by $u^{s \leftrightarrow t}$). Since $u^{s \leftrightarrow t}$ can map source to target and target to source, we only need a single generator (unlike direct feature transformation that needs 2 generators to map target to source, and source to target):

$$\begin{aligned} L_g = &min[log(D(v_i^t)) + log(1 - D(\hat{v}_i^t))] \\ &+ min[1 - cos(\hat{v}_i^t \otimes u^{s \leftrightarrow t}, v_i^s)] \end{aligned} \tag{9}$$

During CLIP training, we freeze D and G and use the synthesized feature to augment the sample in Eqn. 2:

$$L_f = \frac{1}{B} \sum_{i=1}^{B} \mathbf{H}(y_i^s | p(\hat{v}_i^t)) \tag{10}$$

where $p(\hat{v}_i^t)$ is achieved by performing Eqn. 1 in the hypervector space (replace $g(\tau)$ with $\phi(g(\tau))$ and $f(x_i)$ with $\hat{v}_i^t$).

Algorithm 1 HVCLIP for UDA

Input: Labeled source data $D_s = \{x_i^s, y_i^s\}_{i=1}^{N_s}$ and unlabeled target data $D_t = \{x_i^t\}_{i=1}^{N_t}$, number of iteration T.
Output: A fine-tuned CLIP vision encoder (f)

```
Cache original CLIP's feature (v_o^s, v_o^t) for Eqn. 12;
Initialize k = 0;
while k < T do
    k ← k + 1;
    Select a subset of D_s, D_t sample for batch size B;
    Extract and convert features to hypervector (v_i^s, v_i^t);
    Fine-tune generator (G) with (Eqn. 9).
    Fine-tune discriminator (D) with (Eqn. 8).
    Augment CLIP features (Eqn. 7);
    Fine-tune CLIP with Eqn. 2-3, 10-12;
end while
```

3.4 Discrepancy-based Method

Discrepancy Reduction. We reduce the domain discrepancy between source and target domain features in the hypervector space by introducing a cosine similarity loss. As the binding of irrelevant attributes is almost orthogonal, the source-target features pair can be randomly sampled during the training (we do not need source-target pairs that have the same class, background, number of objects, etc.).

$$L_r = \frac{1}{B}\sum_{i=1}^{B}\left[1 - cos(v_i^s, v_i^t))\right] \tag{11}$$

Forgetting Reduction. We reduce the catastrophic forgetting issue by minimizing a cosine similarity loss between the original CLIP's features (before fine-tuning) and the features during the fine-tuning in the hypervector space. Since hypervector is robust, we do not augment images for the original CLIP's features in hypervector space (v_o^s, v_o^t) which allows us to cache the original features once to avoid training overhead:

$$L_c = \frac{1}{B}\sum_{i=1}^{B}\left[2 - cos(v_o^s, v_i^s) - cos(v_o^t, v_i^t)\right] \tag{12}$$

Our final loss is the summation of Eqn. 2-3 (supervised learning and pseudo-label), Eqn. 8-10 (Feature Augmentation) and Eqn. 11-12 (Discrepancy-based Method). Pseudo-code can be found in Algorithm 1. Feature augmentation and discrepancy reduction are complementary because both bridge the domain gap.

4 Experimental Setup

Dataset. We strictly follow the protocol of previous works [2, 19, 20, 49, 50, 60] to evaluate our methods on four UDA datasets with a single model for each ex-

periment. VisDA-2017 [40] has 12 classes of 152k synthetic and 55k real images from COCO [28]. DomainNet [39] has 345 classes of 569k images from 6 domains: Clipart (clp), Infograph (inf), Painting (pnt), Quickdraw (qdr), Real-world (rel), and Sketch (skt). Office-Home [54] has 65 classes of 15.5k images from 4 domains: Art (Ar), Clipart (Cl), Product (Pr), and Real-world (Rw). Office-31 [44] has 31 classes of 4.6k images from 3 domains: Amazon (A), DSLR (D), and Webcam (W). Similar to past works, we use synthetic as the source and real object as the target domain for VisDA-2017, and for the other three datasets, we select one domain as the source and another domain as the target to have 30, 12, and 6 source-target domain combinations of DomainNet, Office-Home, and Office-31.

Training Configuration. We experiment on ResNet-50/101 [12] and ViT-B/16 [8] with a patch size of 16×16 as the vision encoder in CLIP [41]. We freeze the text encoder and only train the vision encoder in CLIP because the training size of text is small (e.g., VisDA-2017 has 12 classes from 2 domains, so the text training size is only 24 samples). We use a learning rate of $1e^{-6}$ and batch size 32 to train all models for 30 epochs. We follow the training configuration in CLIP by using Adam optimizer with decoupled weight decay regularization [34] to decay weights, and cosine scheduler [32] to decay the learning rate. For image augmentation, we follow previous works [20, 50, 66] by using translation, flipping, CTAugment [1], and RandAugment [4]). For our Feature Augmentation, we iteratively train the generator, and discriminator for 100 iterations each, both with learning rate $1e^{-3}$ by cosine scheduler, and we use generated features to fine-tune CLIP on every iteration.

5 Results

5.1 Ablation studies

We show the ablation studies by fine-tuning CLIP with the ResNet-101 backbone on ViSDA-2017. We use both supervised learning and pseudo-label.

Importance of Forgetting/Discrepancy Reduction. Since CLIP suffers from catastrophic forgetting, we first study the importance of our Forgetting Reduction. Tab. 1 shows that CLIP without fine-tuning achieves 82.3% and fine-tuning leads to -25.2% accuracy drops (row: 1,3). Our Forgetting Reduction mitigates the catastrophic forgetting to increase the accuracy by +4.2% and ours is more effective than PADCLIP's CFM (only increases accuracy by +3.7%, row: 1,8,13). We further use our Discrepancy Reduction to mitigate the feature discrepancy between the source and the target domain to improve accuracy by +1.1% (row: 8,9).

Importance of Feature Augmentation. Our Feature Augmentation expands the labeled domain knowledge by synthesizing the labeled target domain features to further improve the accuracy by +1.5% (row: 9,12). We also show that our cycle loss (Eq.9) provides +1% improvement (row: 10, 12). We visualize the effect of our Discrepancy Reduction and our Feature Augmentation in Fig. 3) by

Table 1: Ablation study. We fine-tune CLIP (ResNet-101) on VisDA-2017 source and target domain training set, and test on VisDA-2017 target domain validation set. Row 1,2 shows that hypervector does not degrade performance. Row 1,3 shows that fine-tuning leads to catastrophic forgetting. Row 5-8 shows that hypervector is effective at dimension 1024 onwards (similar observation with past work [38], and we use feature dimension = 512 * hypervector dimension). Row 9,10,12 show the effectiveness of our methods. Row 11,12 show that removing hypervector breaks discrepancy-based method and feature augmentation. Row 8,13 shows that ours is more effective at mitigating catastrophic forgetting than PADCLIP. Row 13-15 shows that PADCLIP is not compatible with discrepancy-based method and feature augmentation.

#	Method	Forgetting Reduction	Discrepancy Reduction	Feature Augment	Cycle Loss	Hypervector Dimension	Accuracy
1	Zero-shot	✗	✗	✗	✗	✗	82.3
2		✗	✗	✗	✗	4096	82.0
3	Fine-tune	✗	✗	✗	✗	✗	57.1
4		✗	✗	✗	✗	4096	61.1
5	HVCLIP (Ours)	✓	✗	✗	✗	64	68.4
6		✓	✗	✗	✗	256	77.5
7		✓	✗	✗	✗	1024	84.7
8		✓	✗	✗	✗	4096	86.5
9		✓	✓	✗	✗	4096	87.6
10		✓	✓	✓	✗	4096	88.1
11		✓	✓	✓	✓	✗	68.5
12		✓	✓	✓	✓	4096	89.1
13	PADCLIP (CFM) [20]	✗	✗	✗	✗	✗	86.0
14		✓	✓	✗	✗	✗	77.8
15		✓	✓	✓	✓	✗	72.1

reshaping the hypervector feature from 2.1M to 2048x1024 and downsample to 16x8 to visualize each feature, and we use t-SNE [7] to visualize the feature distribution. Our source and target domain features of the same class but different domain images have a higher similarity than a traditional fine-tuning method.

Importance of Hypervector. The robustness of the hypervector allows our Forgetting Reduction to constrain the augmented image (during fine-tuning) and the unaugmented original CLIP's prediction (so that the original CLIP's prediction can be cached to speed up the training), allows our Discrepancy Reduction to perform on an unpaired setting (source and target domain images are unaligned), and allows our Feature Augmentation to perform on an unpaired feature translation setting (source and target domain features are also unaligned). Tab. 1 (row: 11,12) shows that removing hypervector collapses our formulation (accuracy decreases by -20.6%). Similarly for PADCLIP, CFM operates in the image space, but our Disprecancy Reduction and Feature Augmentation operates in the feature space, so PADCLIP is not compatible with them (accuracy decreases by -13.9%, row 13-15). We also studied the dimension of hypervector and observed the effective dimension to be 1024 onwards (row: 5-8).

Table 2: Accuracies (%) on VisDA-2017. Full table in Appendix.

Method		plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Avg.
ResNet-101 [12]	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
FixBi [37]		96.1	87.8	90.5	90.3	96.8	95.3	92.8	88.7	97.2	94.2	90.9	25.7	87.2
AaD [62]		97.4	90.5	80.8	76.2	97.3	96.1	89.8	82.9	95.5	93.0	92.0	64.7	88.0
PADCLIP [20]		96.7	88.8	87.0	82.8	97.1	93.0	91.3	83.0	95.5	91.8	91.5	63.0	88.5
SKD [58]		98.8	87.8	92.0	72.0	98.7	93.4	94.8	75.1	92.5	96.0	95.9	72.0	89.1
VFR [19]		97.2	89.3	87.6	83.1	98.4	95.4	92.2	82.5	94.9	93.2	91.3	64.7	89.2
Ours		98.8	90.1	90.8	82.2	97.3	95.5	91.8	82.9	94.9	92.8	92.2	70.8	**90.0**
ViT-B [8]	ViT-B/16	99.1	60.7	70.6	82.7	96.5	73.1	97.1	19.7	64.5	94.7	97.2	15.4	72.6
PMTrans [68]		98.9	93.7	84.5	73.3	99.0	98.0	96.2	67.8	94.2	98.4	96.6	49.0	87.5
SSRT [49]		98.9	87.6	89.1	84.8	98.3	98.7	96.3	81.1	94.9	97.9	94.5	43.1	88.8
PADCLIP [20]		98.1	93.8	87.1	85.5	98.0	96.0	94.4	86.0	94.9	93.3	93.5	70.2	90.9
DIFO [50]		97.5	89.0	90.8	83.5	97.8	97.3	93.2	83.5	95.2	96.8	93.7	65.9	90.3
ADCLIP [46]		99.6	92.8	94.0	78.6	98.8	95.4	96.8	83.9	91.5	95.8	95.5	65.7	90.7
SKD [58]		99.6	95.2	94.3	74.5	99.6	98.0	95.9	79.8	89.8	99.2	96.6	71.8	91.2
VFR [19]		98.4	94.3	89.0	85.4	98.5	98.3	96.1	86.3	95.1	95.2	92.5	70.9	91.7
Ours		99.0	93.7	92.1	84.5	98.8	96.2	94.2	88.6	96.9	96.7	94.5	74.4	**92.5**

Table 3: Accuracies (%) on Office-Home. *Only LLaVO [2] uses FLAN-T5 LLM.

Method		Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg.
ResNet-50 [12]	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
SHOT [27]		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
FixBi [37]		58.1	77.3	80.4	67.7	79.5	78.1	65.8	57.9	81.7	76.4	62.9	86.7	72.7
VFR [19]		58.1	85.0	84.5	77.4	85.0	84.7	76.5	58.8	85.7	75.9	60.4	86.4	76.5
PADCLIP [20]		57.5	84.0	83.8	77.8	85.5	84.7	76.3	59.2	85.4	78.1	60.2	86.7	76.6
SKD [58]		61.6	86.8	86.7	78.0	87.4	86.8	77.3	61.0	87.1	79.6	64.1	88.9	78.8
Ours		62.0	85.8	86.2	77.8	84.3	86.8	80.7	66.5	87.8	80.3	64.9	90.4	**79.5**
ViT-B [8]	ViT-B/16	54.7	83.0	87.2	77.3	83.4	85.5	74.4	50.9	87.2	79.6	53.8	88.8	75.5
SSRT [49]		75.2	89.0	91.1	85.1	88.3	89.9	85.0	74.2	91.3	85.7	78.6	91.8	85.4
PADCLIP [20]		76.4	90.6	90.8	86.7	92.3	92.0	86.0	74.5	91.5	86.9	79.1	93.1	86.7
VFR [19]		78.2	90.4	91.0	87.5	91.9	92.3	86.7	79.7	90.9	86.4	79.4	93.5	87.3
PMTrans [68]		81.2	91.6	92.4	88.9	91.6	93.0	88.5	80.0	93.4	89.5	82.4	94.5	88.9
SKD [58]		79.6	93.7	92.7	89.5	93.7	92.9	89.1	81.1	92.6	90.2	81.6	94.2	89.2
LLaVO* [2]		85.4	96.6	94.1	90.3	97.1	94.4	87.9	85.7	94.5	90.1	85.5	97.3	91.6
Ours		86.3	96.4	94.0	91.6	97.9	94.6	87.5	85.3	94.8	89.9	88.1	97.0	**92.0**

Table 4: Accuracies (%) on Office-31. Full table in Appendix.

Method		A→W	D→W	W→D	A→D	D→A	W→A	Avg.
ResNet-50 [12]	ResNet-50	68.4	96.7	99.3	68.9	62.5	60.7	76.1
SHOT [27]		90.1	98.4	99.9	94.0	74.7	74.3	88.6
CDAN+TN [57]		95.7	98.7	100	94.0	73.4	74.2	89.3
MDD+SCDA [24]		95.3	99.0	100	95.4	77.2	75.9	90.5
FixBi [37]		96.1	99.3	100	95.0	78.7	79.4	91.4
Ours		96.2	99.4	100	96.0	80.1	80.6	**92.1**
ViT-B [8]	ViT-B/16	91.2	99.2	100	90.4	81.1	80.6	90.4
SSRT [49]		97.7	99.2	100	98.6	83.5	82.2	93.5
PADCLIP [20]		97.9	99.2	100	98.5	84.6	85.3	94.3
VFR [66]		98.1	99.4	100.0	98.7	84.4	85.5	94.4
PMTrans [68]		99.1	99.6	100	99.4	85.7	86.3	95.0
Ours		99.3	100	100	99.4	87.3	86.8	**95.5**

Table 5: Accuracies (%) on DomainNet. Column labels are source domains and the row labels are target domains. Full table in Appendix.

ResNet -101 [12]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	19.3	37.5	11.1	52.2	41.0	32.2
inf	30.2	-	31.2	3.6	44.0	27.9	27.4
pnt	39.6	18.7	-	4.9	54.5	36.3	30.8
qdr	7.0	0.9	1.4	-	4.1	8.3	4.3
rel	48.4	22.2	49.4	6.4	-	38.8	33.0
skt	46.9	15.4	37.0	10.9	47.0	-	31.4
Avg.	34.4	15.3	31.3	7.4	40.4	30.5	26.6

ViT -B/16 [8]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	27.2	53.1	13.2	71.2	53.3	43.6
inf	51.4	-	49.3	4.0	66.3	41.1	42.4
pnt	53.1	25.6	-	4.8	70.0	41.8	39.1
qdr	30.5	4.5	16.0	-	27.0	19.3	19.5
rel	58.4	29.0	60.0	6.0	-	45.8	39.9
skt	63.9	23.8	52.3	14.4	67.4	-	44.4
Avg.	51.5	22.0	46.1	8.5	60.4	40.3	38.1

SSRT [49]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	29.4	57.2	26.0	72.6	58.1	48.7
inf	57.0	-	54.4	12.8	69.5	48.4	48.4
pnt	62.9	27.4	-	15.8	72.1	53.9	46.4
qdr	44.6	8.9	29.0	-	42.6	28.5	30.7
rel	66.2	31.0	61.5	16.2	-	52.9	45.6
skt	69.0	29.6	59.0	27.2	72.5	-	51.5
Avg.	59.9	25.3	52.2	19.6	65.9	48.4	45.2

PAD CLIP [20]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	73.6	75.4	74.6	76.4	76.3	75.3
inf	55.1	-	54.3	53.6	54.9	54.9	54.6
pnt	71.1	70.6	-	70.0	72.7	71.7	71.2
qdr	36.8	18.0	32.0	-	31.7	34.9	30.7
rel	84.2	83.5	83.5	83.1	-	83.6	83.6
skt	68.1	66.6	67.2	66.1	67.5	-	67.1
Avg.	63.1	62.5	62.5	69.5	60.6	64.3	63.7

LLaVO [2]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	56.0	71.5	19.9	87.1	74.2	61.8
inf	82.0	-	72.7	21.5	86.9	72.9	67.2
pnt	82.3	55.3	-	17.5	86.8	72.8	63.0
qdr	79.2	52.5	66.3	-	84.8	70.7	70.7
rel	83.7	55.7	74.2	18.8	-	73.6	61.2
skt	83.1	55.3	75.4	21.4	87.1	-	64.4
Avg.	82.1	55.0	72.0	19.8	84.2	72.8	64.7

Ours	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	75.2	77.1	73.8	76.2	77.1	75.9
inf	65.4	-	54.7	51.4	70.2	57.8	59.9
pnt	69.6	72.2	-	70.1	71.6	72.4	71.2
qdr	52.0	18.2	38.5	-	62.4	41.8	42.6
rel	82.0	82.5	80.4	80.5	-	86.8	82.4
skt	76.3	65.0	63.6	67.2	75.5	-	69.5
Avg.	69.1	62.6	62.9	68.6	71.2	67.2	**66.9**

5.2 External Comparison

We achieved state-of-the-art results on four public datasets using the ViT-B/16 backbone, and we tested convolutional backbones (ResNet-50 and ResNet-101) for a fair comparison.

VisDA-2017. We use ResNet-101 as CLIP's backbone to perform a fair comparison against past works. Tab. 2 shows that for ResNet-101, our method improves 8 out of 12 classes against the previous best method (VFR [19]), and our method with ResNet-101 backbone outperforms the recent works with ViT-B backbone (PMTrans [68], SSRT [49]). For the ViT-B/16 backbone, our method also outperforms VFR [19] for 8 out of 12 classes to achieve the new state-of-the-art.

Office-Home/31. We tested ResNet-50 as CLIP's backbone for a fair comparison. On Office-Home, Tab. 3 shows that our method improves +0.7% accuracy over SKD [58] on the ResNet-50 backbone. On ViT-B/16 backbone, LLaVO [2] integrated an extra LLM model (FLAN-T5 [3]) to achieve the previous best result, but we outperform LLaVO without adding any LLM by +0.4% accuracy. For Office-31, the dataset size is small such that the average number of images per class on a single domain is only 49 images (shared by both training and testing), so the accuracy is close to saturation. We achieve +0.7% and +0.5% accuracy improvement over the previous best models (FixBi [37] and PMTrans [68]) on ResNet-50 and ViT-B/16 respectively. Note that the test set is saturated (e.g., SKD [58] uses 5x slower training to achieve +0.3% accuracy vs. PADCLIP [20])

Table 6: Classification accuracy for few-shot learning with ResNet-101 for ImageNet and ViT-B for DomainNet. Oracle CLIP was fine-tuned on the whole training set.

Methods		Shot	Accuracy
CLIP [41]	ImageNet	0	62.5
Tip-Adapter [64]		16	64.8
CLIP-Adapter [9]		16	65.4
CLIP + CoOp [67]		16	66.6
Tip-Adapter-F [64]		16	68.6
PADCLIP [20]		16	69.0
VFR [19]		16	69.2
Ours		16	70.1

Methods		Shot	Accuracy
CLIP [41]	DomainNet	0	56.6
Oracle CLIP [41]		-	63.7
PADCLIP [20]		16	59.2
Ours		16	60.1
PADCLIP [20]		32	60.8
Ours		32	61.4

Table 7: Model-agnostic property of our method on vision-language (CLIP) and vision backbone (BIT, DeiT). We tested PADCLIP and our method on VISDA-2017. We use ViT-B/16 backbone for CLIP.

Model	Accuracy	
	PADCLIP [20]	Ours
CLIP [41]	90.9	91.7
BIT [17]	88.1	89.0
DeiT [52]	88.4	89.9

DomainNet. We use the ViT-B/16 backbone to compare against previous works. Similar to LLaVO [2], VFR [19], SKD [58], PADCLIP [20], our method has a significant improvement over non-CLIP methods (e.g., accuracy +21.7% against SSRT [49]). Our method improves QuickDraw domain over LLaVO [2] (e.g., +27% on ClipArt→QuickDraw and +52.6% on QuickDraw→Painting) because QuickDraw rarely existed in CLIP's pre-training data and does not exist in LLM's training data (so LLaVO struggles with QuickDraw), but our model learned to reduce the discrepancy between source and target domain to achieve +2.2% accuracy over LLaVO (the previous best result).

5.3 Applications

Few-shot Learning. Our method preserves CLIP's few-shot learning ability because we did not modify CLIP's backbone (no extra layers on top of any features). Tab. 6 shows the few-shot learning results on ImageNet [5] and DomainNet [39]. For ImageNet, we use CLIP with ResNet-101 backbone on ImageNet 16-shot for a fair comparison with past works (+3.8% vs. CLIP-Adapter, +2.6% [9] vs. CoOP [67], +0.6% vs. Tip-Adapter-F [64], and +0.9% vs. VFR [19]). For DomainNet, we outperform PADCLIP by 0.9% and 0.6% for 16-shot and 32-shot respectively.

Continual Learning. We show the generalization of our Forgetting Reduction to the Continual Learning setting. We follow ZSCL [66] to evaluate the class-incremental setting, and we outperform ZSCL on 20 steps/last accuracy

Table 8: Continual learning accuracy on class-incremental setting.

Model	CIFAR-100 [18]				TinyImageNet [35]			
	10 steps		20 steps		10 steps		20 steps	
	Average	Last	Average	Last	Average	Last	Average	Last
CLIP [41]	74.5	65.9	75.2	65.7	69.6	65.6	69.5	65.3
Fine-tuned CLIP [41]	65.5	53.2	59.7	43.1	57.1	41.5	54.6	44.6
LwF [26]	65.9	48.0	60.6	40.6	57.6	44.0	54.8	42.3
iCaRL [43]	79.4	71.0	73.3	64.6	73.5	66.0	69.7	64.7
LwF-VR [6]	78.8	70.8	74.5	63.5	74.1	67.1	70.0	63.9
ZSCL [66]	82.2	73.7	80.4	69.6	78.6	71.6	77.2	68.3
Ours	82.6	75.4	81.7	72.0	79.2	73.5	77.4	69.6

by +2.4% and +1.3% on CIFAR-100 [18] and TinyImageNet [35] respectively.

Model-Agnostic. We show the model-agnostic property of our method across different backbones. Tab. 7 shows that we outperform PADCLIP by +0.8%, 0.9%, +1.5% for CLIP (ViT-B/16), BiT ([17]) and DeiT-B [52]. Our method supports both vision-language model (CLIP) and vision models (BiT and DeiT).

5.4 Computational Complexity.

On a single Nvidia Tesla V100 GPU, the baseline of fine-tuning CLIP on the labeled source domain from VisDA-2017 requires 16.5 hours, and adding pseudo-labels increases the training time to 23 hours. We first cache the original CLIP prediction on the entire training set for Forgetting Reduction (only needs once). We then convert CLIP's features into hypervector space and perform Forgetting Reduction (+0.5 hour), Discrepancy Reduction (+0.5 hour), and Feature Augmentation (+6 hours), so the total training time is 30 hours.

6 Conclusion

We propose a High-dimensional Vector in CLIP (HVCLIP) by converting CLIP's features into hypervector space to utilize the robustness of the high-dimensional vector. Hypervector allows us to propose: Forgetting Reduction (to mitigate the catastrophic forgetting), Discrepancy Reduction (to mitigate domain shift between source and target domains), and Feature Augmentation (to synthesize labeled target domain). We achieved the best results on four public datasets and we show the generalization of our method to another application (few-shot learning, continual learning), and across vision-language (CLIP) and vision backbones (BiT, DeiT).

References

1. Berthelot, D., Carlini, N., Cubuk, E.D., Kurakin, A., Sohn, K., Zhang, H., Raffel, C.: Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In: ICLR (2020)

2. Chen, S., Zhang, Y., Jiang, W., Lu, J., Zhang, Y.: Large language models as visual cross-domain learners (2024)
3. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S.S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E.H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q.V., Wei, J.: Scaling instruction-finetuned language models (2022). https://doi.org/10.48550/ARXIV.2210.11416, https://arxiv.org/abs/2210.11416
4. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: CVPRW. pp. 702–703 (2020)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
6. Ding, Y., Liu, L., Tian, C., Yang, J., Ding, H.: Don't stop learning: Towards continual learning for the clip model. arXiv preprint arXiv:2207.09248 (2022)
7. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 647–655. PMLR, Bejing, China (22–24 Jun 2014), https://proceedings.mlr.press/v32/donahue14.html
8. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. ICLR (2021)
9. Gao, P., Geng, S., Zhang, R., Ma, T., Fang, R., Zhang, Y., Li, H., Qiao, Y.: Clip-adapter: Better vision-language models with feature adapters. arXiv preprint arXiv:2110.04544 (2021)
10. Gao, Z., Zhang, S., Huang, K., Wang, Q., Zhong, C.: Gradient distribution alignment certificates better adversarial domain adaptation. In: ICCV. pp. 8937–8946 (2021)
11. Hassan, E.T., Chen, X., Crandall, D.J.: Unsupervised domain adaptation using generative models and self-ensembling. CoRR **abs/1812.00479** (2018), http://arxiv.org/abs/1812.00479
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
13. Kanerva, P.: Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. Cognitive computation **1**(2), 139–159 (2009)
14. Kang, G., Jiang, L., Yang, Y., Hauptmann, A.G.: Contrastive adaptation network for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4893–4902 (2019)
15. Kent, S., Olshausen, B.: A vector symbolic approach to scene transformation. Cognitive computational neuroscience (ccn'17)(extended abstract)[link] (2017)
16. Kim, J., Ryoo, K., Seo, J., Lee, G., Kim, D., Cho, H., Kim, S.: Semi-supervised learning of semantic correspondence with pseudo-labels. In: CVPR. pp. 19699–19709 (2022)
17. Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., Houlsby, N.: Big transfer (bit): General visual representation learning (2019). https://doi.org/10.48550/ARXIV.1912.11370, https://arxiv.org/abs/1912.11370
18. Krizhevsky, A., Nair, V., Hinton, G.: Cifar-100 (canadian institute for advanced research) http://www.cs.toronto.edu/~kriz/cifar.html

19. Lai, Z., Bai, H., Zhang, H., Du, X., Shan, J., Yang, Y., Chuah, C.N., Cao, M.: Empowering unsupervised domain adaptation with large-scale pre-trained vision-language models. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 2691–2701 (January 2024)
20. Lai, Z., Vesdapunt, N., Zhou, N., Wu, J., Huynh, P.C., Li, X., Fu, K.K., Chuah, C.N.: Padclip: Pseudo-labeling with adaptive debiasing in clip for unsupervised domain adaptation. In: ICCV (2023)
21. Lai, Z., Wang, C., Gunawan, H., Cheung, S.C.S., Chuah, C.N.: Smoothed adaptive weighting for imbalanced semi-supervised learning: Improve reliability against unknown distribution data. In: ICML. pp. 11828–11843 (2022)
22. Lee, S., Kim, D., Kim, N., Jeong, S.G.: Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 91–100 (2019)
23. Li, J., Chen, E., Ding, Z., Zhu, L., Lu, K., Shen, H.T.: Maximum density divergence for domain adaptation. IEEE Transactions on Pattern Analysis and Machine Intelligence **43**(11), 3918–3930 (2021). https://doi.org/10.1109/TPAMI.2020.2991050
24. Li, S., Xie, M., Lv, F., Liu, C.H., Liang, J., Qin, C., Li, W.: Semantic concentration for domain adaptation. In: ICCV. pp. 9102–9111 (2021)
25. Li, Y.J., Dai, X., Ma, C.Y., Liu, Y.C., Chen, K., Wu, B., He, Z., Kitani, K., Vajda, P.: Cross-domain adaptive teacher for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7581–7590 (2022)
26. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017)
27. Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: ICML. pp. 6028–6039 (2020)
28. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Doll'a r, P., Zitnick, C.L.: Microsoft COCO: common objects in context. CoRR **abs/1405.0312** (2014), http://arxiv.org/abs/1405.0312
29. Liu, H., Wang, J., Long, M.: Cycle self-training for domain adaptation. Advances in Neural Information Processing Systems **34**, 22968–22981 (2021)
30. Long, M., Cao, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: NeurIPS. pp. 1645–1655 (2018)
31. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. In: ICML. pp. 2208–2217 (2017)
32. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
33. Maurício, J., Domingues, I., Bernardino, J.: Comparing vision transformers and convolutional neural networks for image classification: A literature review. Applied Sciences **13**(9), 5521 (Apr 2023). https://doi.org/10.3390/app13095521, http://dx.doi.org/10.3390/app13095521
34. Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al.: Mixed precision training. In: ICLR (2018)
35. mnmoustafa, M.A.: Tiny imagenet (2017), https://kaggle.com/competitions/tiny-imagenet
36. Montone, G., O'Regan, J.K., Terekhov, A.V.: Hyper-dimensional computing for a visual question-answering system that is trainable end-to-end. arXiv preprint arXiv:1711.10185 (2017)

37. Na, J., Jung, H., Chang, H.J., Hwang, W.: Fixbi: Bridging domain spaces for unsupervised domain adaptation (2021)
38. Neubert, P., Schubert, S., Protzel, P.: An introduction to hyperdimensional computing for robotics. KI-Künstliche Intelligenz **33**(4), 319–330 (2019)
39. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: ICCV. pp. 1406–1415 (2019)
40. Peng, X., Usman, B., Kaushik, N., Hoffman, J., Wang, D., Saenko, K.: Visda: The visual domain adaptation challenge. arXiv preprint arXiv:1710.06924 (2017)
41. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML. pp. 8748–8763 (2021)
42. Rangwani, H., Aithal, S.K., Mishra, M., Jain, A., Radhakrishnan, V.B.: A closer look at smoothness in domain adversarial training. In: ICML. pp. 18378–18399 (2022)
43. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 2001–2010 (2017)
44. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: ECCV. pp. 213–226 (2010)
45. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., Schramowski, P., Kundurthy, S., Crowson, K., Schmidt, L., Kaczmarczyk, R., Jitsev, J.: Laion-5b: An open large-scale dataset for training next generation image-text models. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems. vol. 35, pp. 25278–25294. Curran Associates, Inc. (2022), https://proceedings.neurips.cc/paper_files/paper/2022/file/a1859debfb3b59d094f3504d5ebb6c25-Paper-Datasets_and_Benchmarks.pdf
46. Singha, M., Pal, H., Jha, A., Banerjee, B.: Ad-clip: Adapting domains in prompt space using clip. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops. pp. 4355–4364 (October 2023)
47. Sohn, K., Berthelot, D., Li, C.L., Zhang, Z., Carlini, N., Cubuk, E.D., Kurakin, A., Zhang, H., Raffel, C.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In: NeurIPS (2020)
48. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. In: ECCV. pp. 443–450 (2016)
49. Sun, T., Lu, C., Zhang, T., Ling, H.: Safe self-refinement for transformer-based domain adaptation. In: CVPR. pp. 7191–7200 (2022)
50. Tang, S., Su, W., Ye, M., Zhu, X.: Source-free domain adaptation with frozen multimodal foundation model (2023)
51. Theiss, J., Leverett, J., Kim, D., Prakash, A.: Unpaired image translation via vector symbolic architectures. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 17–32. Springer Nature Switzerland, Cham (2022)
52. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers & distillation through attention. In: ICML. vol. 139, pp. 10347–10357 (July 2021)
53. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014)
54. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: CVPR. pp. 5018–5027 (2017)

55. Wang, Q., Meng, F., Breckon, T.P.: Data augmentation with norm-ae and selective pseudo-labelling for unsupervised domain adaptation. Neural Networks **161**, 614–625 (2023). https://doi.org/https://doi.org/10.1016/j.neunet.2023.02.006, https://www.sciencedirect.com/science/article/pii/S0893608023000692
56. Wang, T., Ding, Z., Shao, W., Tang, H., Huang, K.: Towards fair cross-domain adaptation via generative learning. In: WACV. pp. 454–463 (2021)
57. Wang, X., Jin, Y., Long, M., Wang, J., Jordan, M.: Transferable normalization: Towards improving transferability of deep neural networks. In: NeurIPS (2019)
58. Westfechtel, T., Zhang, D., Harada, T.: Combining inherent knowledge of vision-language models with unsupervised domain adaptation through self-knowledge distillation (2023)
59. Xu, M., Zhang, J., Ni, B., Li, T., Wang, C., Tian, Q., Zhang, W.: Adversarial domain adaptation with domain mixup. In: AAAI. pp. 6502–6509 (2020)
60. Xu, T., Chen, W., Wang, P., Wang, F., Li, H., Jin, R.: Cdtrans: Cross-domain transformer for unsupervised domain adaptation. In: ICLR (2022)
61. Yang, J., Liu, J., Xu, N., Huang, J.: Tvt: Transferable vision transformer for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 520–530 (2023)
62. Yang, S., Wang, Y., Wang, K., Jui, S., et al.: Attracting and dispersing: A simple approach for source-free domain adaptation. In: Advances in Neural Information Processing Systems (2022)
63. Zhang, B., Wang, Y., Hou, W., Wu, H., Wang, J., Okumura, M., Shinozaki, T.: Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. NeurIPS **34**, 18408–18419 (2021)
64. Zhang, R., Zhang, W., Fang, R., Gao, P., Li, K., Dai, J., Qiao, Y., Li, H.: Tip-adapter: Training-free adaption of clip for few-shot classification. In: ECCV. pp. 493–510. Springer (2022)
65. Zhang, Y., Deng, B., Tang, H., Zhang, L., Jia, K.: Unsupervised multi-class domain adaptation: Theory, algorithms, and practice. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
66. Zheng, Z., Ma, M., Wang, K., Qin, Z., Yue, X., You, Y.: Preventing zero-shot transfer degradation in continual learning of vision-language models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 19125–19136 (October 2023)
67. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. Int. J. Comput. Vis. pp. 1–12 (2022)
68. Zhu, J., Bai, H., Wang, L.: Patch-mix transformer for unsupervised domain adaptation: A game perspective. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3561–3571 (2023)
69. Zou, Y., Yu, Z., Liu, X., Kumar, B., Wang, J.: Confidence regularized self-training. In: ICCV. pp. 5982–5991 (2019)