Supplementary Material: Concurrent Subsidiary Supervision for Unsupervised Source-Free Domain Adaptation

Jogendra Nath Kundu[1]*, Suvaansh Bhambri[1]*, Akshay Kulkarni[1]*, Hiran Sarkar[1], Varun Jampani[2], and R. Venkatesh Babu[1]

[1] Indian Institute of Science
[2] Google Research

Supplementary Video

We provide a high-level summary video at https://youtu.be/ENJMz-Eg87k. We visually demonstrate the key insights of our work as well as illustrate the different subsidiary tasks and training algorithm used. We encourage the reader to go through the video for a better understanding of the key ideas.

Supplementary Document

In this document, we provide extensive implementation details, additional performance analysis and ablation studies. Towards reproducible research, we release our complete codebase and trained network weights at https://github.com/val-iisc/StickerDA. This supplementary is organized as follows:

1 Notations

We summarize the notations used in the paper in Table 1. The notations are listed under 5 groups: Models, Preliminaries, Datasets, Samples, and Spaces.

2 Approach

We summarize our approach in Algo. 1 and provide details of the target adaptation objectives that were omitted from the main paper due to space constraints.

Table 1: **Notation Table**

	Symbol	Description
Models	h	Shared backbone feature extractor
	f_g	Goal task classifier
	f_n	Subsidiary task classifier
Preliminaries	p_s	Source marginal distribution
	p_t	Target marginal distribution
	ϵ_s	Source goal task error
	ϵ_t	Target goal task error
	$\epsilon_{s,n}$	Source subsidiary task error
	$\epsilon_{s,n}$	Target subsidiary task error
	$d_{\mathcal{H}}$	$\mathcal{H}$-divergence
	$\mathcal{H}$	Backbone hypothesis space
	$\mathcal{H}_g^{(uns)}$	$\mathcal{H}$-space for unsup. goal task
	$\mathcal{H}_n^{(sup)}$	$\mathcal{H}$-space for sup. subsidiary task
Datasets	$\mathcal{D}_s$	Labeled source dataset
	$\mathcal{D}_t$	Unlabeled target dataset
	$\mathcal{D}_{s,n}$	Subsidiary source dataset
	$\mathcal{D}_{t,n}$	Subsidiary target dataset
	$\mathcal{D}_s^{(od)}$	Pseudo-OOS dataset
Samples	(x_s, y_s)	Labeled source sample
	$(x_{s,n}, y_s, y_n)$	Labeled subsidiary source sample
	$(x_s^{(od)}, y_s^{(od)})$	Labeled pseudo-OOS sample
	x_t	Unlabeled target sample
	$(x_{t,n}, y_n)$	Subsidiary target sample
Spaces	$\mathcal{X}$	Input space
	$\mathcal{Z}$	Backbone feature space
	$\mathcal{C}_g$	Label set for goal task
	$\mathcal{C}_n$	Label set for subsidiary task

2.1 Target adaptation

Self-training loss. We apply self-supervision in the target domain to cluster target samples based on their neighborhood [36]. Each target sample in the feature space is aligned with its neighbor. As a result, the model learns a discriminative metric that translates a point to a semantically similar match. This

is accomplished by reducing the entropy over point similarity. The model learns tightly clustered features as it moves neighboring points closer together, resulting in discriminative decision boundaries.

For each mini-batch of target features, we calculate the similarity to all target samples. Let $F_t^{(mb)} \in \mathbb{R}^{|\mathcal{D}_t| \times d}$ denote the memory bank which stores all target features and d denotes the dimensions for output features $f_g \circ h(x_t)$. Here, $|\mathcal{D}_t|$ denotes the number of samples in the target dataset. All stored features are L2-normalized. Specifically,

$$F_t^{(mb)} = [F_1, F_2, \ldots, F_{|\mathcal{D}_t|}] \tag{1}$$

where F_j denotes the j^{th} item in $F_t^{(mb)}$. Let $f_i = h(x_i)$ denote the features of the current i^{th} mini-batch, and B_t denote the set of indices of the mini-batch samples in $F_t^{(mb)}$. The probability that f_i is a neighbor of the feature F_j is,

$$p_{i,j} = \frac{\exp(F_j^T f_i / \mathcal{T})}{\sum_{j=1, j\neq i}^{|\mathcal{D}_t|} \exp(F_j^T f_i / \mathcal{T})} \tag{2}$$

where the temperature parameter $\mathcal{T}$ controls the number of neighbors. Then, the entropy *i.e.* the loss is defined as,

$$\mathcal{L}_{st} = -\frac{1}{|B_t|} \sum_{i \in B_t} \sum_{j=1, j\neq i}^{\mathcal{D}_t} p_{i,j} \log(p_{i,j}) \tag{3}$$

Diversity loss. We encourage the prediction to be balanced to avoid degenerate solutions, where the model predicts all data to a particular class (and does not predict other classes for any target sample). We employ the prediction diversity loss, which has been frequently used in clustering [9] and domain adaptation [20]. The diversity objective is,

$$\mathcal{L}_{div}(f_g \circ h(x)) = D_{KL}(\hat{p}, \frac{1}{|\mathcal{C}_g|} \mathbb{1}_{|\mathcal{C}_g|}) - \log |\mathcal{C}_g| \tag{4}$$

where $\mathbb{1}_{|\mathcal{C}_g|}$ represents a $|\mathcal{C}_g|$-dimensional vector of ones, $\hat{p} = \mathbb{E}_{x_t \in \mathcal{D}_t}[\sigma(f_g \circ h(x_t))]$ is average output embedding for entire target dataset, and σ denotes softmax.

2.2 Subsidiary DA suitability criteria

2.2.1 Subsidiary-Domain Similarity Metric (DSM). As discussed in Sec. 3.1.3 of the main paper, we define *subsidiary-domain similarity metric*, γ_{DSM} as the inverse of the $\mathcal{H}$-divergence between the two domains. We follow [8] and use the $\mathcal{A}$-distance [4] between the goal task dataset $\mathcal{D}_s$ and the subsidiary task dataset $\mathcal{D}_{s,n}$ as a proxy for $\mathcal{H}$-divergence. We define the dataset labels as 1 for subsidiary source dataset $\mathcal{D}_{s,n}$ and 0 for original source dataset $\mathcal{D}_s$ and train a linear binary classifier on the features of a frozen ImageNet-pretrained

[26] ResNet-50 [10] with a subset of the mixed data, and obtain the classifier error on the other subset as ψ. The DSM is then computed as,

$$d_{\mathcal{A}}(\mathcal{D}_s, \mathcal{D}_{s,n}) = 2\psi(1-\psi) \tag{5}$$

$$\gamma_{DSM}(\mathcal{D}_s, \mathcal{D}_{s,n}) = 1 - \frac{1}{2} d_{\mathcal{A}}(\mathcal{D}_s, \mathcal{D}_{s,n}) \tag{6}$$

How to choose the threshold ζ_d? Insight 2 introduced a threshold ζ_d for DSM to select pretext tasks suitable for subsidiary supervised DA. To choose a threshold, we first consider the $\mathcal{A}$-distances between the actual source and target domains. These $\mathcal{A}$-distances are in the range of 1.5 to 2.0 [31] for Office-Home and indicate the range of $\mathcal{A}$-distances corresponding to realistic domain shifts. This range corresponds to the range of 0 to 0.25 in terms of DSM. In Fig. 7A of the main paper, we observed DSM in a range of 0 to 0.3 for the patch-location and image-rotation subsidiary task samples w.r.t. the original samples, indicating that these tasks induce a realistic domain shift. Contrary to this, our proposed sticker task produced DSM in the range of 0.6 to 0.9, indicating much better domain preservation. Thus, we choose the threshold $\zeta_d = 0.5$ which represents $\sim$70% reduced domain shift w.r.t. realistic domain shifts (*i.e.* w.r.t. 1.5 to 2.0).

2.2.2 Subsidiary-Task Similarity Metric (TSM). γ_{TSM} determines how similar a subsidiary task is to the goal task. TSM is calculated using the basic linear evaluation protocol [29] in self-supervised literature. It illustrates the degree of compatibility between the two tasks. For computing γ_{TSM}, we train a linear classifier f_n on the features $h_{s,g}$ for subsidiary task dataset $\mathcal{D}_{s,n}$ extracted using a frozen source-pretrained ResNet-50 [10] backbone. For the sticker classification task, we randomly select 4 classes to keep the number of classes uniform for the different subsidiary task candidates illustrated in Fig. 1C and Fig. 7B in the main paper. We thus obtain the error for the different subsidiary task classifiers as $\hat{\epsilon}_{s,n}$ and the subsidiary-task similarity metric is computed as:

$$\gamma_{TSM}(\mathcal{D}_s, \mathcal{D}_{s,n}) = 1 - \min_{f_n} \hat{\epsilon}_{s,n}(h_{s,g}) \tag{7}$$

How to choose the threshold ζ_n? Insight 3 introduced a threshold ζ_n for TSM to select pretext tasks suitable for subsidiary supervised DA. The task similarity of the subsidiary task is dependent on the goal task. For computing the threshold for TSM, we plot the γ_{TSM} for the candidate subsidiary tasks (Fig. 7B) and select the appropriate threshold ζ_n. Based on our observations in Fig. 7B of the main paper, we set ζ_n as 0.6.

Suitability criterion. Definition 1 in the main paper gives the overall suitability criterion for selecting the subsidiary task as:

$$\gamma_{DSM}(\mathcal{D}_s, \mathcal{D}_{s,n}) + \gamma_{TSM}(\mathcal{D}_s, \mathcal{D}_{s,n}) > \zeta \tag{8}$$

Therefore, we set the threshold ζ as a sum of ζ_d and ζ_n *i.e.* 1.1.

Algorithm 1 Pseudo-code for the proposed approach

Source-side training

Input: source data $\mathcal{D}_s$, stickered source data $\mathcal{D}_{s,n}$, pseudo-OOS dataset $\mathcal{D}_s^{(od)}$, ImageNet pretrained backbone h (as per [20]), randomly initialized goal classifier f_g and randomly initialized sticker classifier f_n.

Goal task source pre-training

for $iter < MaxIter$ **do**:
  Sample batch from $\mathcal{D}_s \cup \mathcal{D}_{s,n}$
  Compute $\mathcal{L}_{s,g}$ using Eq. 7 (main paper)
  update θ_h, θ_{f_g} by minimizing $\mathcal{L}_{s,g}$
end for

Sticker task source pre-training

for $iter < MaxIter$ **do**:
  Sample batch from $\mathcal{D}_{s,n}$
  Sample batch from $\mathcal{D}_s^{(od)}$
  Compute $\mathcal{L}_{s,n}$ and $\mathcal{L}_s^{(od)}$ using Eq. 8 (main paper)
    ▷ using samples from $\mathcal{D}_{s,n}$ and $\mathcal{D}_s^{(od)}$ respectively
  update θ_{f_n} by minimizing $\mathcal{L}_{s,n}, \mathcal{L}_s^{(od)}$ using separate Adam optimizers
end for

Target-side training

Input: target data $\mathcal{D}_t$, stickered target data $\mathcal{D}_{t,n}$, source-side pretrained backbone h, goal classifier f_g and sticker classifier f_n.

Source-free target adaptation

for $iter < MaxIter$ **do**:
  Sample batch from $\mathcal{D}_t$
  Sample batch from $\mathcal{D}_{t,n}$
  Compute $\mathcal{L}_{st}$ and $\mathcal{L}_{div}$ using Eq. 3, 4 (suppl.)
    ▷ using samples from both $\mathcal{D}_t$ and $\mathcal{D}_{t,n}$
  Compute $\mathcal{L}_{t,n}$ using Eq. 9 (main paper)
    ▷ using samples from only $\mathcal{D}_{t,n}$
  update θ_h, θ_{f_n} by minimizing $\mathcal{L}_{t,n}$
  update θ_h by minimizing $\mathcal{L}_{st}, \mathcal{L}_{div}$ using separate Adam optimizers
end for

3 Implementation details

3.1 Sticker intervention

We define a sticker as a printed alphabet with a random color and random texture [5] within the alphabet. We scale the sticker randomly and paste it at a random location within a black image (all zeros) with the same size as goal task sample $x_s \in \mathbb{R}^{H \times W}$, yielding $x_n \in \mathbb{R}^{H \times W}$ (see Fig. 1A). The corresponding sticker-task labels y_n, along with x_n, form the sticker dataset $\mathcal{D}_n$. We also define a pixel-wise mask to perform mixup [37] only at the sticker pixels to avoid the effects of the black background on the rest of the goal task image.

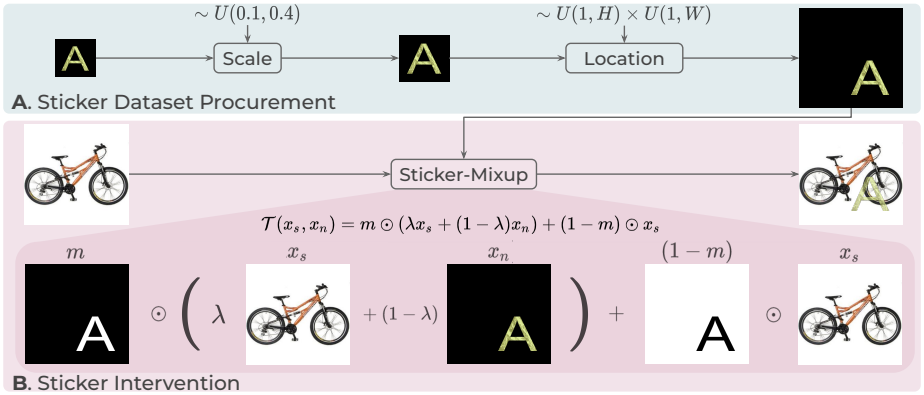

Fig. 1: Illustration of **A.** sticker dataset procurement and **B.** sticker intervention $\mathcal{T}$ (see Sec. 3.1). *Best viewed in color.*

Specifically, $m(u) = \mathbb{1}(x_n(u) \neq 0)$ where $u : [u_x, u_y]$ denotes the spatial index in an $H \times W$ lattice. As shown in Fig. 1B, a goal task sample x, *i.e.* either x_s, $x_s^{(od)}$ or x_t, and a sticker x_n are combined using mixup [37] as,

$$\mathcal{T}(x, x_n) = m \odot (\lambda x + (1-\lambda)x_n) + (1-m) \odot x \tag{9}$$

where λ denotes the mixup ratio, $\odot$ represents element-wise multiplication and $\mathcal{T}$ is the sticker intervention (as defined in Insight 4 of main paper).

3.1.1 Hyperparameters

a) Sticker shape is decided by randomly selected alphabets.
b) Sticker size is determined by randomly sampling the size ratio between sticker and goal task images from a uniform distribution over the range $[0.1, 0.4]$.
c) Sticker location for pasting the sticker in the goal task image is sampled from a uniform distribution over the ranges $[1, H]$ and $[1, W]$. The sampled coordinates are rounded down to the nearest integer for pasting the sticker.
d) Number of sticker classes determines the difficulty level of the subsidiary supervised DA problem.
e) Mixup ratio determines the visibility of the sticker w.r.t. the goal task image. We use a constant mixup ratio of 0.4.
We provide ablations for these hyperparameters in Sec. 4.2.

3.1.2 Usage. The intervention is applied in the same manner to both source samples x_s as well as target samples x_t, yielding sticker labels y_n for the sticker classifier. Mitsuzumi *et al.* [23] show that, beyond a certain grid size (4x4), shuffling the grid patches makes the domain unrecognizable. Inspired by this, we generate the pseudo-OOS dataset by randomly shuffling the grid patches with

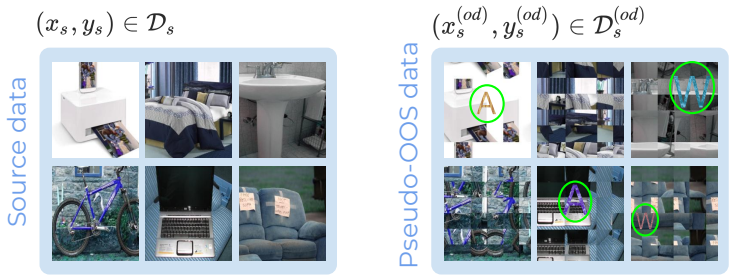

Fig. 2: The pseudo-OOS data $\mathcal{D}_s^{(od)}$ contains patch-shuffled versions of source data $\mathcal{D}_s$. Green circles only highlight the stickers and are not part of the samples.

a grid size of (6x6) as shown in Fig. 2. The sticker intervention is also applied to the pseudo-OOS samples in order to emphasize the difference between source and pseudo-OOS samples even when stickers are present. However, for pseudo-OOS samples, the sticker label is treated as $y_s^{(od)}$, for the OOS node to act as an implicit domain discriminator, leading to improved source-target alignment.

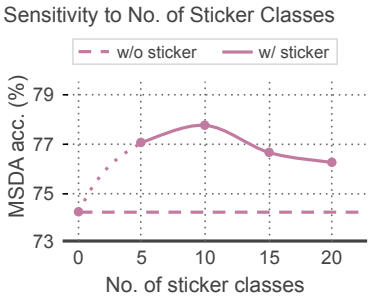

Fig. 3: Sensitivity to no. of sticker classes $|\mathcal{C}_n|$ for Office-Home MSDA.

Method	SF	Office-31			
		→A	→W	→D	Avg.
PFSA [7]	✗	57.0	97.4	99.7	84.7
SImpAl [31]	✗	70.6	97.4	99.2	89.0
WAMDA [1]	✗	72.0	98.6	99.6	90.0
MIAN [25]	✗	76.2	98.4	99.2	91.3
MLAN [35]	✗	75.7	98.8	99.6	91.4
Source-combine	✗	65.2	94.6	98.4	86.1
SHOT[20]-Ens	✓	75.0	94.9	97.8	89.3
DECISION [2]	✓	75.4	98.4	99.6	91.1
CAiDA [6]	✓	75.8	98.9	99.8	91.6
Ours	✓	**78.3**	**99.1**	**99.7**	**92.4**

Table 2: Multi-Source DA (MSDA) comparisons on Office-31.

Enabling source-free DA. The proposed sticker intervention can be used within source-free constraints. This is because, the alphabet font can be shared between source-side and target-side while the texture dataset [5] is open-source.

3.2 Experimental settings

Architecture details. We use a ResNet-50 [10] backbone for Office-Home, Office-31 and DomainNet, and ResNet-101 for VisDA, for a fair comparison with prior works. We employ the same network design as SHOT [20], *i.e.* replacing the classifier with a fully connected layer with batch norm [13] and another fully connected layer with weight normalization [28]. For the subsidiary classifier, we use the same architecture after ResLayer-3.

Optimization details. We employ multiple Adam optimizers during training to avoid loss weighting hyperparameters. Specifically, we use a distinct optimizer for each loss term. In each training iteration, we optimize only one of the losses (round robin method). Each optimizer uses a learning rate of 1e-3. Intuitively, each Adam optimizer's moment parameters adaptively scale the associated gradients, eliminating the requirement for loss-scaling hyperparameter tuning. For source model training, following [20], we set the maximum number of epochs to 100 and 30 for Office-31 and Office-Home, whereas it is set to 10 and 15 for VisDA and DomainNet respectively. For adaptation, the maximum number of epochs is set to 15 for all datasets, following [20].

4 Analysis

We provide more comparisons with prior state-of-the-art methods and report hyperparameter sensitivity analyses.

4.1 Extended comparisons and ablations

a) Single-Source DA for Office-31 and VisDA. Our approach outperforms source-free NRC [36] and SHOT++ [21] by 1.5% and 1.7% respectively on Office-31 (Table 3), and gives comparable performance to non-source-free works. On the larger and more challenging VisDA dataset, our approach surpasses NRC by 1.6% and SHOT++ by 1% (Table 3).
b) Multi-Source DA for Office-31. To analyze our performance on closed-set MSDA, we compare our approach with source-free and non-source-free prior arts in Table 2. Even without domain labels, our approach achieves *state-of-the-art* results on the Office-31 benchmark, even for the non-source-free setting.
c) Variance across random seeds. We highlight the significance of our results by reporting the mean and standard deviation of accuracy for 5 runs with different random seeds (2nd last row of Table 3) for SSDA. We observe low variance even w.r.t. prior non-source-free works.
d) Ablations for target adaptation. We present ablations on the goal task objectives for the target-side training ($\mathcal{L}_{st}$ and $\mathcal{L}_{div}$) in Table 4. First, we compare the baseline *i.e.* source-trained model (#1) with the $\mathcal{L}_{div}$ based DA model (#2). It is interesting to note that only using the diversity objective with subsidiary supervision improves SSDA and MSDA by 2.4% and 5.5% respectively over the baseline (#2 vs. #1), highlighting the relevance of diversity promotion.

The neighborhood clustering based self-training loss $\mathcal{L}_{st}$ improves target clustering in the latent $\mathcal{Z}$ space by bringing the backbone features $h(x)$ closer to their respective nearest neighbors. Using $\mathcal{L}_{st}$ in conjunction with the subsidiary DA loss $\mathcal{L}_{t,n}$ enhances the goal task adaptation by 10.5% and 5.2% for SSDA and MSDA respectively, compared to not using $\mathcal{L}_{st}$ (#4 vs. #2). We observe that employing both $\mathcal{L}_{div}$ and $\mathcal{L}_{st}$ further improves the performance by 3.8% and 1.9% for SSDA and MSDA respectively (#4 vs. #3), demonstrating that the two losses are complementary for goal task DA.

Table 3: Single-Source Domain Adaptation (SSDA) on Office-31 and VisDA benchmarks with mean and standard deviation over 5 runs. The last row indicates the variance over different sets of sticker shapes while others indicate variance over different random seeds. SF indicates *source-free* DA.

Method	SF	Office-31							VisDA
		A→D	A→W	D→W	W→D	D→A	W→A	Avg	S → R
FAA [12]	✗	94.4	92.3	99.2	99.7	80.5	78.7	90.8	-
RFA [3]	✗	93.0	92.8	99.1	100.0	78.0	77.7	90.2	79.4
SCDA [19]	✗	95.4	95.3	99.0	100.0	77.2	75.9	90.5	-
DMRL [32]	✗	93.4±0.5	90.8±0.3	99.0±0.2	100.0±0.0	73.0±0.3	71.2±0.3	87.9	-
MCC [15]	✗	98.6±0.1	95.5±0.2	98.6±0.1	100.0±0.0	72.8±0.3	74.9±0.3	89.4	-
CAN [16]	✗	95.0±0.3	94.5±0.3	99.1±0.2	99.8±0.2	78.0±0.3	77.0±0.3	90.6	87.2
RWOT [34]	✗	94.5±0.2	95.1±0.2	99.5±0.2	100.0±0.0	77.5±0.1	77.9±0.3	90.8	-
FixBi [24]	✗	95.0±0.4	96.1± 0.2	99.3±0.2	100.0±0.0	78.7±0.5	79.4± 0.3	91.4	87.2
CDAN+RADA [14]	✗	96.1±0.4	96.2±0.4	99.3±0.1	100.0±0.0	77.5±0.1	77.4±0.3	91.1	76.3
SHOT [20]	✓	94.0	90.1	98.4	99.9	74.7	74.3	88.6	82.9
CPGA [27]	✓	94.4	94.1	98.4	99.8	76.0	76.6	89.9	84.1
HCL [11]	✓	90.8	91.3	98.2	100.0	72.7	72.7	87.6	83.5
VDM-DA [30]	✓	93.2	94.1	98.0	100.0	75.8	77.1	89.7	85.1
A^2Net [33]	✓	94.5	94.0	99.2	100.0	76.7	76.1	90.1	84.3
NRC [36]	✓	96.0	90.8	99.0	100.0	75.3	75.0	89.4	85.9
SHOT++ [21]	✓	94.3	90.4	98.7	99.9	76.2	75.8	89.2	87.3
3C-GAN [18]	✓	92.7±0.4	93.7±0.2	98.5±0.1	99.8±0.2	75.3±0.5	**77.8**±0.1	89.6	-
SFDA [17]	✓	92.2±0.2	91.1±0.3	98.2±0.3	99.5±0.2	71.0±0.2	71.2±0.2	87.2	-
Ours (random seed)	✓	95.6±0.2	**94.6**±0.2	**99.2**±0.1	99.8±0.2	77.0±0.3	77.7±0.3	**90.7**	**88.2**±0.4
Ours (random sticker)	✓	95.5±0.1	94.2±0.2	98.9±0.2	99.9± 0.1	**77.2**±0.1	76.3±0.2	90.3	88.0±0.3

4.2 Hyperparameter sensitivity analysis

a) Sticker shape. We randomly selected 10 alphabets and used them consistently to report all the results in the main paper. However, to test the variance of our approach w.r.t. sticker shape, we report the mean and standard deviation over 5 runs of SSDA experiments on Office-31 (last row of Table 3), randomly sampling the 10 alphabets (*i.e.* sticker shapes) for each run. We observe a low standard deviation indicating low sensitivity to the sticker shapes.

b) Sticker size. We select this scale range based on empirical evidence (Table 5). We observe that adaptation performance suffers with sticker scale less than 0.1, since the sticker is hardly visible, making it difficult for the sticker classifier to receive meaningful supervision. The performance with larger sized stickers (more than 0.7) also drops as the sticker may occlude goal task content significantly.

c) Sticker location. We observe that our approach is only mildly sensitive to this hyperparameter (Table 5). We restrict the sticker location to regions far from the image centre and observe slightly lower accuracy. On the other hand, pasting the sticker near the image centre area further decreases performance as the sticker may occlude a larger part of the goal task content. Allowing the sticker to be pasted uniformly across the image yields the best performance.

d) Number of sticker classes. We perform a sensitivity analysis for the number of sticker categories $|\mathcal{C}_n|$ for MSDA on Office-Home (Fig. 3). We observe that

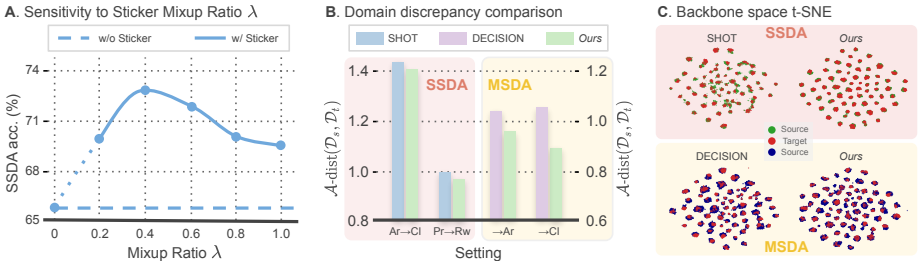

Fig. 4: **A.** Sensitivity to sticker mixup ratio λ for SSDA on Office-Home. **B.** $\mathcal{A}$-distance between source and target data on Office-Home. **C.** Backbone feature space t-SNE comparisons with SHOT [20] on Rw→Pr (SSDA), DECISION [2] on →Ar (MSDA) from Office-Home.

Table 4: Ablation study on Office-Home. SF, SSDA and MSDA indicate source-free, single-source DA and multi-source DA.

#	Target-side			SF	**Office-Home**	
	$\mathcal{L}_{st}$	$\mathcal{L}_{div}$	$\mathcal{L}_{t,n}$		SSDA	MSDA
1	✗	✗	✗	-	60.2	66.9
2	✗	✓	✓	✓	62.6	72.4
3	✓	✗	✓	✓	69.3	75.7
4	✓	✓	✓	✓	**73.1**	**77.6**

Table 5: Sensitivity analysis for sticker scale and location on the single-source DA (SSDA) benchmark of Office-Home dataset.

Sticker scale	Acc.
$0.05 - 0.1$	71.8
$0.1 - 0.4$	72.2
$0.4 - 0.7$	**73.1**
$0.7 - 1.0$	72.0

Sticker location	Acc.
Central region	71.5
Except central region	72.0
Entire image	**73.1**

performance improves with increasing number of classes upto 10 and reduces slightly for higher $|\mathcal{C}_n|$. Overall, we observe consistent gains over the baseline.
e) Mixup ratio $\boldsymbol{\lambda}$. In Fig. 4A, we observe consistent gains over the baseline (mixup ratio $\lambda = 0$ *i.e.* sticker classifier and losses not used) for a wide range of λ values. The best performance is observed for $\lambda = 0.4$. Intuitively, higher mixup ratios imply very low sticker visibility while lower mixup ratios imply more occlusion of goal task content, both yielding slightly lower performance.

4.3 Domain discrepancy analysis

In Fig. 4B, we report $\mathcal{A}$-distance as a measure of the domain discrepancy $d_{\mathcal{H}}(p_s, p_t)$ across different source-target pairings in the backbone feature space $\mathcal{Z}$ for our approach and prior source-free state-of-the-art SSDA [20] and MSDA [2] works. A lower value for $\mathcal{A}$-distance indicates lower domain discrepancy. In comparison to prior works, our technique clearly achieves lower $\mathcal{A}$-distance between source and target for both settings. This implies that our backbone learns domain-agnostic features that are more generalized to the target domain. This corresponds to an increase in target performance and demonstrates that subsidiary supervised adaptation efficiently minimizes the latent space distribution shift, $d_{\mathcal{H}}(p_s, p_t)$, consistent with Insight 1 of the main paper.

4.4 Domain alignment analysis

In Fig. 4C, we present t-SNE [22] visualizations of backbone features learned by SHOT [20] and our approach for SSDA, and DECISION [2] and our approach for MSDA. As expected, all three approaches aid the formation of target clusters but source-target alignment for prior arts is weaker compared to our approach. We also observe that our method better preserves the source clusters (*green* in SSDA and *blue* in MSDA) while producing dense clusters for the target features (*red* in both settings) that are better aligned with the source clusters. This improved source-target alignment can be attributed to the OOS node in the sticker classifier, consistent with Insight 6 presented in the main paper.

Table 6: Training and inference time comparison w.r.t. NRC [36] and SHOT++ [21]. All timings are obtained using a single 1080Ti GPU.

Method	Training time (in sec), Ar→Cl				Inference time (in millisec)	Office-Home	
	Source pretrain	Sticker pretrain	Target adapt	Total		SSDA Avg.	MSDA Avg.
NRC	282	-	1060	1342	1.9	72.2	74.7
SHOT++	306	-	10043	10349	1.9	73.0	75.7
Ours	282	643	**284**	**1209**	1.9	**74.0**	**77.6**

4.5 Efficiency analysis

We provide detailed training time comparisons of our work w.r.t. NRC [36] and SHOT++ [21] in Table 6. We make certain observations: **1)** We achieve superior target adaptation efficiency with the fastest training (4th column) and the best performance (last 2 columns). Note that we use same learning rate and scheduler as in NRC and SHOT++. **2)** Inference complexity (6th column) is same for all as we do not require the subsidiary classifier during inference.

Table 7: Combining multiple subsidiary tasks.

Office-Home	SSDA
Baseline (B)	66.2
B + patch-loc	67.6
B + rotation	67.9
B + rotation + patch-loc	68.0
B + sticker-rot	69.0
B + sticker-clsf	69.7
B + sticker-rot + sticker-clsf	69.5

4.6 Combining subsidiary tasks

Introducing multiple subsidiary tasks in the same framework brings up additional challenges like multi-task balancing. For instance, consider a combination of rotation (`Rot`) and patch-location (`PL`). From Fig. 1C in the main paper, `Rot` has high TSM while `PL` has high DSM. This does not imply that combining `Rot` and `PL` would yield a better overall TSM+DSM, and may rather have a detrimental impact. Thus, one should aim for a subsidiary task having both TSM and DSM greater than those of `Rot` and `PL`. Empirically, we do not find any conclusive result. In Table 7, we observe that while `Rot`+`PL` shows marginal gains, combining Sticker-rot and Sticker-clsf shows degraded performance.

Table 8: **Comparisons w.r.t. pretext task based DA works.**

Method	Pretext Task	High DSM+TSM	Additional regularization
SS-DA [17]	Rotation, Rot. Patch Jigsaw	✗	Adv. alignment, AdaBN
JiGen [5]	Jigsaw	✗	Augmentations
PAC [34]	Rotation	✗	Aug. consistency
Ours	**Sticker**	✓	**None**

Table 9: **Comparisons w.r.t. prior source-free DA works.**

Method	Key insights (differences)	Common
SHOT	Info-max. for implicit feature alignment	$\mathcal{L}_{div}$
SHOT++	Easy-hard target split for better adaptation	$\mathcal{L}_{div}$
CPGA	Contrastive prototypes for better pseudo-labels	$\mathcal{L}_{st}$
GSFDA	Local struct. clustering for better repr. learning	$\mathcal{L}_{st}, \mathcal{L}_{div}$
NRC	Cluster assumption for better pseudo-labels	$\mathcal{L}_{st}, \mathcal{L}_{div}$
A^2Net	Dual classifiers to find src-similar tgt samples and contrastive matching for category-wise alignm.	-
Ours	**1.** How and when subsidiary task is DA-assistive? **2.** Criteria for DA-assistive subsidiary tasks **3.** Process of sticker intervention	$\mathcal{L}_{st}, \mathcal{L}_{div}$

4.7 Differences and relationships with prior-arts

These are discussed in Table 8 and 9. Our method is free from additional regularization unlike prior works (Table 8). While our key contributions are unique, the common loss terms are widely used (*e.g.* GSFDA, NRC in Table 9).

References

1. Aggarwal, S., Kundu, J.N., Babu, R.V., Chakraborty, A.: WAMDA: Weighted alignment of sources for multi-source domain adaptation. In: BMVC (2020) 7
2. Ahmed, S.M., Raychaudhuri, D.S., Paul, S., Oymak, S., Roy-Chowdhury, A.K.: Unsupervised multi-source domain adaptation without access to source data. In: CVPR (2021) 7, 10, 11
3. Awais, M., Zhou, F., Xu, H., Hong, L., Luo, P., Bae, S.H., Li, Z.: Adversarial robustness for unsupervised domain adaptation. In: ICCV (2021) 9
4. Ben-David, S., Blitzer, J., Crammer, K., Pereira, F.: Analysis of representations for domain adaptation. In: NeurIPS (2006) 3
5. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , Vedaldi, A.: Describing textures in the wild. In: CVPR (2014) 5, 7
6. Dong, J., Fang, Z., Liu, A., Sun, G., Liu, T.: Confident anchor-induced multi-source free domain adaptation. In: NeurIPS (2021) 7
7. Fu, Y., Zhang, M., Xu, X., Cao, Z., Ma, C., Ji, Y., Zuo, K., Lu, H.: Partial feature selection and alignment for multi-source domain adaptation. In: CVPR (2021) 7
8. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. The Journal of Machine Learning Research **17**(1), 2096–2030 (2016) 3
9. Gomes, R., Krause, A., Perona, P.: Discriminative clustering by regularized information maximization. In: NeurIPS (2010) 3
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016) 4, 7
11. Huang, J., Guan, D., Xiao, A., Lu, S.: Model adaptation: Historical contrastive learning for unsupervised domain adaptation without source data. In: NeurIPS (2021) 9
12. Huang, J., Guan, D., Xiao, A., Lu, S.: RDA: Robust domain adaptation via fourier adversarial attacking. In: ICCV (2021) 9
13. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015) 7
14. Jin, X., Lan, C., Zeng, W., Chen, Z.: Re-energizing domain discriminator with sample relabeling for adversarial domain adaptation. In: ICCV (2021) 9
15. Jin, Y., Wang, X., Long, M., Wang, J.: Minimum class confusion for versatile domain adaptation. In: ECCV (2020) 9
16. Kang, G., Jiang, L., Yang, Y., Hauptmann, A.G.: Contrastive adaptation network for unsupervised domain adaptation. In: CVPR (2019) 9
17. Kim, Y., Cho, D., Han, K., Panda, P., Hong, S.: Domain adaptation without source data. IEEE Transactions on Artificial Intelligence **2**(6), 508–518 (2021) 9
18. Li, R., Jiao, Q., Cao, W., Wong, H.S., Wu, S.: Model adaptation: Unsupervised domain adaptation without source data. In: CVPR (2020) 9
19. Li, S., Xie, M., Lv, F., Liu, C.H., Liang, J., Qin, C., Li, W.: Semantic concentration for domain adaptation. In: ICCV (2021) 9

20. Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: ICML (2020) 3, 5, 7, 8, 9, 10, 11
21. Liang, J., Hu, D., Wang, Y., He, R., Feng, J.: Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021) 8, 9, 11
22. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008) 11
23. Mitsuzumi, Y., Irie, G., Ikami, D., Shibata, T.: Generalized domain adaptation. In: CVPR (2021) 6
24. Na, J., Jung, H., Chang, H.J., Hwang, W.: FixBi: Bridging domain spaces for unsupervised domain adaptation. In: CVPR (2021) 9
25. Park, G.Y., Lee, S.W.: Information-theoretic regularization for multi-source domain adaptation. In: ICCV (2021) 7
26. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019) 4
27. Qiu, Z., Zhang, Y., Lin, H., Niu, S., Liu, Y., Du, Q., Tan, M.: Source-free domain adaptation via avatar prototype generation and adaptation. In: IJCAI (2021) 9
28. Salimans, T., Kingma, D.P.: Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In: NeurIPS (2016) 7
29. Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., Madry, A.: Do adversarially robust imagenet models transfer better? In: NeurIPS (2020) 4
30. Tian, J., Zhang, J., Li, W., Xu, D.: Vdm-da: Virtual domain modeling for source data-free domain adaptation. IEEE Transactions on Circuits and Systems for Video Technology (2021) 9
31. Venkat, N., Kundu, J.N., Singh, D.K., Revanur, A., Babu, R.V.: Your classifier can secretly suffice multi-source domain adaptation. In: NeurIPS (2020) 4, 7
32. Wu, Y., Inkpen, D., El-Roby, A.: Dual mixup regularized learning for adversarial domain adaptation. In: ECCV (2020) 9
33. Xia, H., Zhao, H., Ding, Z.: Adaptive adversarial network for source-free domain adaptation. In: ICCV (2021) 9
34. Xu, R., Liu, P., Wang, L., Chen, C., Wang, J.: Reliable weighted optimal transport for unsupervised domain adaptation. In: CVPR (2020) 9
35. Xu, Y., Kan, M., Shan, S., Chen, X.: Mutual learning of joint and separate domain alignments for multi-source domain adaptation. In: WACV (2022) 7
36. Yang, S., Wang, Y., van de Weijer, J., Herranz, L., Jui, S.: Exploiting the intrinsic neighborhood structure for source-free domain adaptation. In: NeurIPS (2021) 2, 8, 9, 11
37. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: ICLR (2018) 5, 6