Blending-target Domain Adaptation by Adversarial Meta-Adaptation Networks

Ziliang Chen[1], Jingyu Zhuang[1], Xiaodan Liang[1,2], Liang Lin[1,2*]
[1]Sun Yat-sen University [2]DarkMatter AI Research
c.ziliang@yahoo.com, zhuangjy6@mail2.sysu.edu.cn, xdliang328@gmail.com, linliang@ieee.org

1. Appendix.A

1.1. Unsupervised Meta-learner

Our meta-learner U is trained as deep embedding clustering ([15]) by receiving data and its feature-level feedbacks (the concatenation of the feature input to the discriminators and its classification result, for brevity, we mark $F(\boldsymbol{x}^{(t)})$ in our paper). It is very important to note that, distinguished from the original version solely using a DAE to initiate data embeddings, our meta-learner leverage the improved DEC [5] as our implementation, where the clustering embeddings are updated by reconstruction loss as well as the clustering objective *w.r.t.* centroids $\{\mu_j\}_{j=1}^{k}$. Therefore, U_1, U_2 and $\{\mu_j\}_{j=1}^{k}$ are alternatively updated by

$$
\begin{aligned}
U_2 =& U_2 - \frac{\alpha}{m}\sum_{i=1}^{m}\frac{\partial L_{\mathrm{rec}}(\boldsymbol{x}_i;F)}{\partial U_2} \\
U_1 =& U_1 - \frac{\alpha}{m}\sum_{i=1}^{m}[\frac{\partial L_{\mathrm{rec}}(\boldsymbol{x}_i;F) - \sum_{j=1}^{k} p_{i,j}\log\frac{p_{i,j}}{q_{i,j}}}{\partial U_1}] \\
\mu_j =& \mu_j - 2\frac{\alpha}{m}\sum_{i=1}^{m}\sum_{j=1}^{K}(1+||U_1(\boldsymbol{x}_i^{(t)})-\mu_j||)^{-1} \\
& \qquad (p_{ij}-q_{ij})(U_1(\boldsymbol{x}_i^{(t)})-\mu_j)
\end{aligned}
\tag{1}
$$

where α and m denote the learning rate and mini-batch size for optimizing meta-learner. We set initial learning rate as 0.001 and batch size is 256. The auto-encoder architecture implemented in our experiments has been shown in Table.1.

Table 1. The architecture of our unsupervsied meta-learner.

	Input size	Output size	Activator
Encoder:			
En_Fc_1	image size	500	ReLU
En_Fc_2	500	1000	ReLU
En_Fc_3	1000	k	ReLU
Decoder:			
De_Fc_1	k	1000	ReLU
De_Fc_2	1000	1000	ReLU
De_Fc_3	1000	image-size	ReLU

1.2. Entropy Penalty

In our implementation, we leverage a well-known cluster assumption [4] to regulate the classifier C learning with unlabeled target data. It can be interpreted as the minimization of the conditional entropy term with respect to the output of $C\big(F(\boldsymbol{x})\big)$

$$L_{\mathrm{ent}}(F,C) = -\mathbb{E}_{\boldsymbol{x}\sim\mathcal{T}}\, C\big(F(\boldsymbol{x})\big)^T \log\, C\big(F(\boldsymbol{x})\big) \tag{2}$$

. The objective forces the classification to be confident on the unlabeled target example, which drives the classifier's decision boundaries away from the target unlabeled examples. It has been applied in wide range of domain adaptation researches [12] [9] [3]. However, while using available data to empirically estimate the expected loss, [4] demonstrates that such approximation provably breaks down if $C\big(F(\cdot)\big)$ does not satisfy local Lipschitz condition. Specifically, the classifier without local Lipschitz constraint can abruptly changes its prediction, which allows placement of the classifier decision boundaries close to target training examples while the empirical conditional entropy is still minimized. To prevent this issue, we follow the technique in [13] where virtual adversarial perturbation term [11] is incorporated to regulate the classifier and feature extractor:

$$
\begin{aligned}
&L_{\mathrm{vir}}(F,C) = \\
&\mathop{\mathbb{E}}_{\boldsymbol{x}^{(s)}\sim\mathcal{S}}\big[\max_{||\mathbf{r}||\le\epsilon}\mathbf{D}_{\mathrm{KL}}(C\big(F(\boldsymbol{x}^{(s)})\big)||C\big(F(\boldsymbol{x}^{(s)}+\mathbf{r})\big))\big] \\
&\quad +\rho\mathbb{E}_{\boldsymbol{x}^{(t)}\sim\mathcal{T}}\big[\max_{||\mathbf{r}||\le\epsilon}\mathbf{D}_{\mathrm{KL}}(C\big(F(\boldsymbol{x}^{(t)})\big)||C\big(F(\boldsymbol{x}^{(t)}+\mathbf{r})\big))\big]
\end{aligned}
\tag{3}
$$

where $\mathbf{D}_{\mathrm{KL}}$ indicates KL divergence. $\mathbf{r}$ indicates the virtual adversarial perturbation upper bounded by a magnitude $\epsilon > 0$ on source and target images ($\boldsymbol{x}^{(s)}\in\mathcal{S}$ and $\boldsymbol{x}^{(t)}\in\mathcal{T}$), which are obtained by maximizing the classification differences between $C\big(F(\boldsymbol{x}^{(s)})\big)$ and $C\big(F(\boldsymbol{x}^{(s)}+\mathbf{r})\big)$. This restrictions are simultaneously proposed on source and target and ρ is the balance factor between them. In this way, the collaborative meta-adversarial adaptation objectives (Eq.8-10 in our paper) are reformulated as:

*Corresponding author: Liang Lin.

Table 2. Backbone-1 in digit-five experiment.

	Kernel size	Output dimension	BN/IN	Activation	Dropout
Feature extractor:					
Conv1_1	5*5	64*24*24	BN	ReLU	0
Maxpool	2*2	64*12*12			0.5
Conv1_2	5*5	50*8*8	BN	ReLU	0
Maxpool	2*2	50*4*4			0.5
Classifier:					
Fc_1	50*4*4	100	BN	ReLU	0.5
Fc_2	100	100	BN	ReLU	0
Fc_3	100	10		Softmax	0
Discriminator:					
Reversed gradient layer					
Fc	50*4*4	100	BN	ReLU	0
Fc (D_{st})	100	2		Softmax	0
Fc (D_{mt})	100	4		Softmax	0

Table 3. Backbone-2 in digit-five experiment.

	Kernel size	Output dimension	BN/IN	Activation	Dropout
Feature extractor:					
Conv1_1	3*3	64*32*32	IN/BN	LeakyReLU(0.1)	0
Conv1_2	3*3	64*32*32	BN	LeakyReLU(0.1)	0
Conv1_3	3*3	64*32*32	BN	LeakyReLU(0.1)	0.5
Maxpool	2*2	64*16*16			
Conv2_1	3*3	64*16*16	BN	LeakyReLU(0.1)	0
Conv2_2	3*3	64*16*16	BN	LeakyReLU(0.1)	0
Conv2_3	3*3	64*16*16	BN	LeakyReLU(0.1)	0.5
Maxpool	2*2	64*8*8			
Classifier:					
Conv2_1	3*3	64*8*8	BN	LeakyReLU(0.1)	0
Conv2_2	3*3	64*8*8	BN	LeakyReLU(0.1)	0
Conv2_3	3*3	64*8*8	BN	LeakyReLU(0.1)	0
Averagepool		64*1*1			
Fc	64*10	10		Softmax	0
Discriminator:					
Fc	64*8*8+10	100		ReLU	0
Fc (D_{st})	100*1	1		Sigmoid	0
Fc (D_{mt})	100*4	4		Softmax	0

$$
\begin{aligned}
\max_{D_{\mathrm{st}}, D_{\mathrm{mt}}} \min_{F,C} \; & V_{\mathrm{joint}}(D_{\mathrm{st}}, D_{\mathrm{mt}}, F, C) \\
&= V_{\mathrm{st}}(F, D_{\mathrm{st}}, C) + \gamma V_{\mathrm{mt}}(F, D_{\mathrm{mt}}) \\
&\quad + \beta L_{\mathrm{ent}}(F, C) + L_{\mathrm{vir}}(F, C)
\end{aligned} \tag{4}
$$

and

$$
\max_{D_{\mathrm{st}}, D_{\mathrm{mt}}} V_{\mathrm{alter}}(D_{\mathrm{st}}, D_{\mathrm{mt}}) = V_{\mathrm{st}}(F, D_{\mathrm{st}}, C) + V_{\mathrm{mt}}(F, D_{\mathrm{mt}}) \tag{5}
$$

$$
\begin{aligned}
\min_{F,C} V_{\mathrm{alter}}(F, C) =& V_{\mathrm{st}}(F, D_{\mathrm{st}}, C) + \gamma \widetilde{V}_{\mathrm{mt}}(F, D_{\mathrm{mt}}) \\
&+ \beta L_{\mathrm{ent}}(F, C) + L_{\mathrm{vir}}(F, C)
\end{aligned} \tag{6}
$$

. We provide the ablation study of the entropy penalty in Table 5 .

1.3. The Selection of k

In AMEAN, k denotes the number of sub-target domains and is pre-given. Table 6 demonstrates that choosing k as the number of sub-targets leads to the superior performance of AMEAN. However, whether AMEAN would achieve the better performance if k is adaptively determined, remains an open and interesting question. We would like to investigate this topic in the future.

1.4. Architectures

The architectures for digit recognition in Digit-five have been illustrated in Table.2 , 3 . The first backbone is based on LeNet and the second is derived from [13] for comparing their state-of-the-art models VADA and DIRT-T. The architectures for object recognition in Office-31 and Office-

Table 4. The hyper-parameters setting in our experiment.

	Office-31, Office-HOME		Digit-five	
	AlextNet	ResNet-50	Backbone-1	Backbone-2
mini-batch size	32	32	128	100
λ	0.1	1	1	1 (update D_{st}) / 0.01 (update F)
γ	0.01	$\frac{\mathrm{iter}}{\mathrm{max_iter}}$	0.1	$\frac{\mathrm{iter}}{\mathrm{max_iter}}$
β	0.01	0.1	0	0.01
ρ	0.01	0	0	0.01
M	2000	2000	20000	10000
image size	227×227	227×227	28×28	28×28

Table 5. Some ablation of entropy term.

	mt→mm,sv,up,sy	mm→mt,sv,up,sy	D→A,W	W→A,D
w	**85.1**	**77.6**	**62.8**	**59.7**
w/o	83.7	76.9	62.6	59.2

Table 6. **Acc** is the average accuracy over five sub-transfer tasks in Digit-five. The number of sub-targets in Digit-five is 4.

k	2	3	4	5	6	7	8
Acc	79.3	83.2	**83.7**	82.1	83.2	82.0	78.6

Home are based on AlexNet and ResNet-50, which are consistent with the previous studies [7] [8] [9] [1] .

1.5. Training Details

We evenly separate the proportion of the source and target examples in each mini-batch. Concretely, we promise that a half of examples in a mini-batch are drawn from $\mathcal{S}$ and the rest belong to the mixed target domain training set $\mathcal{T}^{\mathrm{train}}$: In *digit-five*, we randomly drew target examples from the mixed target set $\mathcal{T}^{\mathrm{train}}$ to construct our mini-batches; In *Office-31* and *Office-Home*, we promise the number of target examples from different meta-sub-target are the same by repeat sampling.

In the Digit-five experiment, we add a confusion loss [6] *w.r.t.* $\mathcal{S}$ to train the backbone-2. It stabilizes the alternating adaptation since the mixed target in Digit-five is more diverse than the other benchmarks' and the alternating learning manner is quite instable in these scenarios. The implementation can be found in our code.

The hyper-parameters are shown in Table 4 .

2. Appendix.B

2.1. Evalutation Metrics for BTDA

We elaborate how to calculate ANT and RNT in our experiment in details:

$$ANT = \max\{0, Acc_{\mathrm{BTDA}} - Acc_{\mathrm{Source-only}}\} \quad (7)$$

where $Acc_{\mathrm{Source-only}}$ denotes the classification accuracy about the model trained on the source labeled dataset $\mathcal{S}$ and tested on the mixed target set $\mathcal{T}^{\mathrm{test}} = \bigcup_{j=1}^{k} \mathcal{T}_j^{\mathrm{test}}$; Acc_{BTDA} denotes the multi-target-weighted classification accuracy of the evaluated DA model under BTDA setup:

$$Acc_{\mathrm{BTDA}} = \sum_{j=1}^{k} \alpha_j Acc_{\mathrm{BTDA}}^{(j)} \quad (8)$$

where $Acc_{\mathrm{BTDA}}^{(j)}$ denotes the DA model classification accuracy on the j^{th} sub-target domain test set $\mathcal{T}_j^{\mathrm{test}}$ when the evaluated DA models (*e.g.*, JAN, DAN, AMEAN, *etc*) is trained with the source labeled set $\mathcal{S}$ and the mixed target unlabeled set $\mathcal{T}^{\mathrm{train}} = \bigcup_{j=1}^{k} \mathcal{T}_j^{\mathrm{train}}$ (BTDA setup). $\{\alpha_j\}_{j=1}^{k}$ denotes the proportion of the multi-target mixture. It is derived from the domain-set proportion in benchmarks, which are valued by $\{0.236, 0.236, 0.236, 0.236, 0.056\}$, $\{0.686, 0.121, 0.193\}$ and $\{0.155, 0.280, 0.285, 0.280\}$ in Digit-five, Office-31 and Office-Home. When we draw the subset of domains to construct the mixed target, $\{\alpha_j\}_{j=1}^{k}$ is obtained by normalizing these corresponding benchmark-specific domain-set proportion [1] .

In reality, we can obtain Acc_{BTDA} by directly evaluating the DA models on the mixed test set $\mathcal{T}_j^{\mathrm{test}}$, which leads to the same results in (8).

Based on (8), we also define the RNT metric

$$RNT = Acc_{\mathrm{BTDA}} - \sum_{j=1}^{k} \alpha_j Acc_j \quad (9)$$

where Acc_j denotes the j^{th}-target test classification accuracy with respect to a single-target DA classifier trained on

[1]The numbers are based on the hidden sub-target test set proportions in a mixed target

the source labeled set $\mathcal{S}$ and the j^{th} sub-target unlabeled set $\mathcal{T}_j^{\text{train}}$. Note that,

- Acc_j is derived from a DA model trained with datasets $\mathcal{S}$ and $\mathcal{T}_j^{\text{train}}$. It means that Acc_i, Acc_j ($i \neq j$) are derived from different DA models, which employ the same DA algorithms yet are trained on $\mathcal{T}_i^{\text{train}}$, $\mathcal{T}_j^{\text{train}}$ and tested on $\mathcal{T}_i^{\text{test}}$, $\mathcal{T}_j^{\text{test}}$, respectively.
- $Acc_{\text{BTDA}}^{(j)}$ is derived from a DA model trained with $\mathcal{S}$ and $\mathcal{T}^{\text{train}} = \bigcup_{j=1}^{k} \mathcal{T}_j^{\text{train}}$. Hence $Acc_{\text{BTDA}}^{(i)}$, $Acc_{\text{BTDA}}^{(j)}$ ($i \neq j$) are derived from the same DA model, which employ the same DA algorithms and is trained on ($\mathcal{S} \cup \mathcal{T}^{\text{train}}$) and then, tested on $\mathcal{T}_i^{\text{test}}$ and $\mathcal{T}_j^{\text{test}}$ to induce $Acc_{\text{BTDA}}^{(i)}$, $Acc_{\text{BTDA}}^{(j)}$.

.

Equal-weight ANT, RNT. It worth noting that, though ANT/RNT in (7),(9) are able to reflect BTDA models' performances on a mixed target domain set, it is not enough to demonstrate the comprehensive performances of the models over multi-sub-target domains, since it does not equally weight hidden sub-target domains. More specifically, imagine that we have a small set of target images belonging to a hidden sub-target, which the model performs poorly on. Then the RNT metric would shield the model's incapacity on that domain.

In order to thoroughly reflect the capacities of evaluated models, we additionally report the results when the proportion $\{\alpha_j\}_{j=1}^{k}$ is equally set. In particular, we tend to consider the equal-weight classification accuracy ($Acc_{\text{BTDA}}^{(\text{EW})}$), and quantify the corresponding negative transfer ANT_{EW} and RNT_{EW} in this setup:

$$\begin{aligned} Acc_{\text{BTDA}}^{(\text{EW})} &= \frac{1}{k}\sum_{j=1}^{k} Acc_{\text{BTDA}}^{(j)} \\ ANT_{EW} &= \max\{0, Acc_{\text{BTDA}}^{(\text{EW})} - Acc_{\text{Source-only}}^{(\text{EW})}\} \\ RNT_{EW} &= Acc_{\text{BTDA}}^{(\text{EW})} - \frac{1}{k}\sum_{j=1}^{k} Acc_j \end{aligned} \tag{10}$$

. The metrics developed from (7 8 9) could be viewed as the complementary of what we report in the paper.

2.2. Evaluated Baselines in BTDA setup.

Beyond our AMEAN model, we also reported the BTDA performances from state-of-the-art DA baselines in Digit-five, Office-31, Office-Home. The baselines include Deep Adaptation Network (**DAN**) [7], Residual Transfer Network (**RTN**) [9], Joint Adaptation Network (**JAN**) [10], Generate To Adapt (**GTA**) [12], Adversarial Discriminative Domain Adaptation (**ADDA**) [14], Reverse Gradient (**RevGrad**) [1] [2], Virtual Adversarial Domain Adaptation (**VADA**) [13] and its variant **DIRT-T** [13].

In the *Digit-five* experiment, DAN, ADDA, GTA, ReGrad are all derived from their official codes. To promise a fair comparison, we standardize the backbones by LeNet to report Acc_{BTDA}, $Acc_{\text{BTDA}}^{(\text{EW})}$ and the negative transfer effects. VADA and DIRT-T are evaluated by their official codes to provide the results. Their model architectures are consistent with our backbone-2.

In the *Office-31* and *Office-Home* experiments, we employ the official codes of DAN, RTN, JAN, ReGrad to report Acc_{BTDA}, $Acc_{\text{BTDA}}^{(\text{EW})}$ in the *Office-31* and *Office-Home* experiments.

The codes of all evaluated baselines can be found in their literatures. For a fair comparison, Acc_j mainly originates from the reported results in their papers.

2.3. BTDA experiments by equal-weight evaluation metrics

The equal-weight versions of the classification accuracy ($Acc_{\text{BTDA}}^{(\text{EW})}$), absolute negative transfer (ANT_{EW}) and relative negative transfer RNT_{EW} over all the evaluated baselines in Digit-five, Office-31 and Office-Home are reported in Table 7- 11.

References

[1] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

[2] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. *Domain-Adversarial Training of Neural Networks*. 2017.

[3] T. Gebru, J. Hoffman, and L. Fei-Fei. Fine-grained recognition in the wild: A multi-task domain adaptation approach. *arXiv preprint arXiv:1709.02476*, 2017.

[4] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.

[5] X. Guo, L. Gao, X. Liu, and J. Yin. Improved deep embedded clustering with local structure preservation. In *International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 1753–1759, 2017.

[6] J. Hoffman, E. Tzeng, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *Domain Adaptation in Computer Vision Applications*, pages 173–187. Springer, 2017.

[7] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015.

[8] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. *arXiv preprint arXiv:1605.06636*, 2016.

[9] 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.

[10] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. 2017.

Table 7. The equal-weight (EW) Classification accuracy (ACC %), *absolute negative transfer* (ANT%) and *relative negative transfer* (RNT%) on Digit-five in BTDA setup. **BLUE**, **RED** indicate ANT and RNT, respectively. More viewed in (10).

Models	mt→mm,sv,up,sy		mm→mt,sv,up,sy		sv→mm,mt,up,sy		sy→mm,mt,sv,up		up→mm,mt,sv,sy		Avg	
	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT
Backbone-1:												
Source only	36.6	0	57.3	0	67.1	0	74.9	0	36.9	0	54.6	0
ADDA	52.5	-7.4	58.9	-1.2	$46.4^{(-20.7)}$	-16.0	$67.0^{(-7.9)}$	-7.0	$34.8^{(-2.1)}$	-13.3	$51.9^{(-2.7)}$	-9.0
DAN	38.8	-8.6	$53.5^{(-3.8)}$	-4.5	$55.1^{(-12.0)}$	-3.0	$65.8^{(-9.1)}$	-2.8	$27.0^{(-9.9)}$	-11.0	$48.0^{(-6.6)}$	-6.0
GTA	51.4	-9.0	$54.2^{(-3.1)}$	-2.1	$59.8^{(-7.3)}$	-3.6	**76.2(+1.3)**	-0.6	41.3	-2.0	56.6	-3.6
RevGrad	60.2	-6.2	66.0	-4.6	$64.7^{(-2.3)}$	-6.0	$69.2^{(-5.7)}$	-7.1	44.3	-6.3	60.9	-6.0
AMEAN	**61.2 (+1.0)**	-	**66.9 (+0.9)**	-	**67.2 (+0.1)**	-	$73.3^{(-1.6)}$	-	**47.5 (+3.2)**	-	**63.2 (+2.3)**	-
Backbone-2:												
Source only	55.8	0	55.2	0	74.3	0	76.4	0	50.6	0	62.5	0
VADA	79.4	-4.9	72.5	-3.1	76.4	-2.2	82.8	-3.8	56.4	-8.7	73.5	-4.5
DIRT-T	77.5	-6.5	76.8	-4.4	**79.7 (+1.8)**	-4.9	80.9	-3.9	47.0	-7.5	72.4	-5.5
AMEAN	**86.9 (+7.5)**	-	**78.5 (+1.7)**	-	77.9	-	**85.6 (+2.8)**	-	**75.5 (+19.1)**	-	**80.9 (+7.4)**	-

Table 8. The equal-weight (EW) Classification accuracy (ACC %), *absolute negative transfer* (ANT%) and *relative negative transfer* (RNT%) on Office31 in BTDA setup. **BLUE**, **RED** indicate ANT and RNT, respectively. More viewed in (10).

Backbones	Models	A→D,W		D→A,W		W→A,D		Avg	
		ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT
AlexNet	Source only	62.7	0	73.3	0	74.4	0	70.1	0
	DAN	68.2	0.0	$71.4^{(-1.9)}$	-4.0	$73.2^{(-1.2)}$	-3.3	70.9	-2.4
	RTN	70.7	-1.7	$69.8^{(-3.5)}$	-4.1	$71.5^{(-2.9)}$	-3.9	70.7	-3.2
	JAN	73.5	-0.1	73.6	-4.1	75.0	-2.5	74.0	-2.2
	RevGrad	74.1	1.0	$72.1^{(-1.2)}$	-2.8	$73.4^{(-1.0)}$	-1.8	73.2	-1.2
	AMEAN (ours)	**74.9 (+0.8)**	-	**74.9 (+1.3)**	-	**76.2 (+1.2)**	-	**75.3 (+1.3)**	-

Table 9. The equal-weight (EW) Classification accuracy (ACC %), *absolute negative transfer* (ANT%) and *relative negative transfer* (RNT%) on Office31 in BTDA setup. **BLUE**, **RED** indicate ANT and RNT, respectively. More viewed in (10).

Backbones	Models	A→D,W		D→A,W		W→A,D		Avg	
		ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT
ResNet-50	Source only	68.7	0	79.6	0	80.0	0	76.1	0
	DAN	77.9	-2.0	$75.0^{(-4.6)}$	-5.0	80.0	-1.3	77.6	-3.0
	RTN	84.1	+2.9	$77.2^{(-2.4)}$	-4.4	$79.0^{(-1.0)}$	-3.3	80.1	-1.6
	JAN	84.6	-0.8	82.7	-0.6	83.4	-1.8	83.6	-1.0
	RevGrad	79.0	-2.3	81.4	-1.5	82.3	-1.3	80.9	-1.7
	AMEAN (ours)	**89.8 (+5.2)**	-	**84.6 (+1.9)**	-	**84.3 (+0.9)**	-	**86.2 (+2.6)**	-

Table 10. The equal-weight (EW) Classification accuracy (ACC %), *absolute negative transfer* (ANT%) and *relative negative transfer* (RNT%) on OfficeHome in BTDA setup. **BLUE**, **RED** indicate ANT and RNT, respectively. More viewed in (10).

Backbones	Models	Ar→Cl,Pr,Rw		Cl→Ar,Pr,Rw		Pr→Ar,Cl,Rw		Rw→Ar,Cl,Pr		Avg	
		ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT
AlexNet	Source only	33.4	0	35.3	0	30.6	0	37.9	0	34.3	0
	DAN	39.7	-3.6	41.6	-2.8	37.8	-3.1	46.8	-2.4	41.5	-3.0
	RTN	42.8	-2.0	43.4	-2.4	39.1	-2.1	48.8	-2.5	43.5	-2.2
	JAN	43.5	-2.9	44.6	-3.5	39.4	-5.2	48.5	-5.2	44.0	-4.2
	RevGrad	42.2	-3.3	43.8	-3.5	39.9	-3.6	47.7	-5.0	43.4	-3.9
	AMEAN (ours)	**44.6 (+1.1)**	-	**45.6 (+1.0)**	-	**41.4 (+1.5)**	-	**49.3 (+0.5)**		**45.2 (+1.2)**	-

Table 11. The equal-weight (EW) Classification accuracy (ACC %), *absolute negative transfer* (ANT%) and *relative negative transfer* (RNT%) on OfficeHome in BTDA setup. **BLUE**, **RED** indicate ANT and RNT, respectively. More viewed in (10).

Backbones	Models	Ar→Cl,Pr,Rw		Cl→Ar,Pr,Rw		Pr→Ar,Cl,Rw		Rw→Ar,Cl,Pr		Avg	
		ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT	ACC^{ANT}	RNT
ResNet-50	Source only	47.6	0	41.8	0	43.4	0	51.7	0	46.1	0
	DAN	55.6	-0.5	55.1	+0.9	47.8	-4.0	56.6	-6.3	53.8	-2.5
	RTN	53.9	-1.8	55.4	-0.7	47.2	-3.3	51.8	-3.0	52.1	-2.2
	JAN	58.3	-0.4	59.2	+2.1	51.9	-1.2	57.8	-6.1	56.8	-1.5
	RevGrad	58.4	-3.0	57.0	-2.2	52.2	-4.6	62.0	-3.2	57.4	-3.2
	AMEAN (ours)	**64.3 (+5.9)**	-	**64.2 (+5.0)**	-	**59.0 (+6.8)**	-	**66.4 (+4.4)**		**63.5 (+6.1)**	-

[11] T. Miyato, S.-i. Maeda, S. Ishii, and M. Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[12] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. *ArXiv e-prints, abs/1704.01705*, 2017.

[13] R. Shu, H. H. Bui, H. Narui, and S. Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018.

[14] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017.

[15] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.