# Supplementary Material: Robust Target Training for Multi-Source Domain Adaptation

Zhongying Deng[1, 2]
z.deng@surrey.ac.uk
Da Li[3]
dali.academic@gmail.com
Yi-Zhe Song[1,2]
y.song@surrey.ac.uk
Tao Xiang[1, 2]
t.xiang@surrey.ac.uk

[1] University of Surrey
Guildford, UK
[2] iFlyTek-Surrey Joint Research Center on Artificial Intelligence
[3] Samsung AI Center
Cambridge, UK

This Supplementary Material is organized as follows: Section A details the experimental settings. Section B introduce the FixMatch-CM. Section B.1 presents the ablation study of the FixMatch-CM on PACS. Section C analyzes the extra training cost of our BORT$^2$. Section D provides further analysis on the outer loop optimization. Section E evaluates the design of adaptive threshold.

## A Experimental Settings

### A.1 Datasets and Protocols

We validate the efficacy of our proposed method on three popular MSDA datasets, namely PACS [5], Digit-Five, and DomainNet [11]. **PACS** has four different domains (Cartoon, Photo, Sketch and Art Painting), including 9,991 images of 7 categories. We adopt the official train-val splits in [5]. **Digit-Five** has five domains, MNIST [4], SVHN [8], USPS, Synthetic Digits [1], and MNIST-M [1]. We follow the protocol in [11]. When USPS is used as a source domain, we use all its 9,298 images for training. For the other domains, the training set comprises 25,000 randomly sampled images while the test set has 9,000 images. **DomainNet** is the largest MSDA dataset available, with about 0.6 million images of 345 categories. These images are collected from six domains, including Sketch, Quickdraw, Painting, Infograph, Real and Clipart. Due to the diversity between different domains in terms of image style, background etc., DomainNet is also the most challenging MSDA dataset so far. In all setups, we conduct the leave one held-out protocol and report the average results of three runs.

### A.2 Implementation Details

For the first-step labeling function learning in BORT$^2$, we use FixMatch-CM (see Section B) as the model unless stated otherwise. The other details are as follows: On Digit-Five, we use

the backbone with three convolution layers and two fully connected layers, also the same as [11]. The model is optimized with SGD for 30 epochs with initial learning rate 0.05, decayed using the cosine annealing strategy [6], and batch size $B = 64$ per domain. On PACS, we adopt an ImageNet pretrained ResNet-18 [2] as our backbone and optimize it for 100 epochs with Adam [3]. We set the batch size 16 and the initial learning rate 5e-4. On DomainNet, an ImageNet pretrained ResNet-101 [2] is used following [11]. Then the model is trained with SGD for 40 epochs. The initial learning rate is 0.002. We use batch size 6 for each domain.

For training the final model in BORT$^2$, the pseudo-labels are generated in the same way as [12]. The weight of entropy maximization loss $\lambda$ in Eq. (5) is 0.1 and the threshold $m$ in $\mathcal{L}_{ment}$ is 4. The model performance is found to be insensitive to both hyper-parameters (see Section 4.2 in the main paper). We adopt an adaptive threshold for the $\tau$ in Eq. (5) to filter pseudo labels. Specifically, $\tau$ is initialized with the mean $p_{mean}$ and standard deviation $p_{std}$ of the prediction in a mini-batch, i.e., $p_{mean} + p_{std}$. Then it is gradually decreased to $p_{mean} - p_{std}$ in an exponential moving average way: $\tau = \alpha\tau + (1-\alpha)(p_{mean} - p_{std})$, where $\alpha$ is fixed to 0.999. This adaptive scheme is evaluated in Section E and can be regarded as a curriculum sampling strategy.

We initialize the noisy-robust final model $M_\psi$ first by copying the first-step trained MSDA model $F_\theta$, followed by adding a stochastic layer in the fourth residual block. We start the bi-level optimization when noise-robust model $M_\psi$ converges, with a learning rate 5e-5 for fine-tuning the labeling function $F_\theta$.

We run our experiments on PyTorch [9, 10]. Our code is based on Dassl [9, 14] [1].

# B FixMatch-CM

In this section, we introduce FixMatch-CM for the first-step MSDA model training. FixMatch-CM adapts the vanilla FixMatch [12], originally proposed for semi-supervised learning, to MSDA. Specifically, FixMatch-CM incorporates two different strong augmentations, image level CutMix [13] and feature level MixStyle [15], to the vanilla FixMatch for alleviating domain shift further.

Given a batch of source and target images $[\{(x_i^{\mathcal{S}_1}, y_i^{\mathcal{S}_1}), ..., (x_i^{\mathcal{S}_K}, y_i^{\mathcal{S}_K})\}_{i=1}^B, \{x_i^{\mathcal{T}}\}_{i=1}^B]$, we first obtain the pseudo-labels $\{\hat{y}_i^{\mathcal{T}}\}_{i=1}^B$ for target images as Eq. (2). Then, following CutMix [13], we crop a patch of random size from each image $x_i^{\mathcal{S}_k}$, and fill in that region with a patch from another randomly sampled image $x^{\mathcal{D}_{k,i}}$ from $\mathcal{D} = \{\mathcal{S}_1, ..., \mathcal{S}_K, \mathcal{T}\}$. Correspondingly, we mix their labels $y_i^{\mathcal{S}_k}, y^{\mathcal{D}_{k,i}}$, so we have $\{(\dot{x}_i^{\mathcal{S}_1}, y_i^{\mathcal{S}_1}, y^{\mathcal{D}_{1,i}}, \lambda_i^{\mathcal{S}_1}), ..., (\dot{x}_i^{\mathcal{S}_K}, y_i^{\mathcal{S}_K}, y^{\mathcal{D}_{K,i}}, \lambda_i^{\mathcal{S}_K})\}_{i=1}^B$, $\{(\dot{x}_i^{\mathcal{T}}, \hat{y}_i^{\mathcal{T}}, y^{\mathcal{D}_{T,i}}, \lambda_i^{\mathcal{T}})\}_{i=1}^B$, where $\dot{x}_i^{\mathcal{S}_k}$ is the augmented image, and $\lambda_i^{\mathcal{S}_k}$ is its mixing ratio. We also employ a feature-level augmentation in our FixMatch-CM, namely MixStyle [15]. [15] claimed that the mean and standard deviation of $\dot{x}_i^{\mathcal{S}_k}$'s feature map encode domain-specific style statistics. Mixing styles of $\dot{x}_i^{\mathcal{S}_k}$ and $\dot{x}_i^{\bar{\mathcal{D}}(\mathcal{S}_k)}$ can generate interpolated styles, thus augmenting the vanilla feature space, where $\bar{\mathcal{D}}(\mathcal{S}_k)$ are the domains excluding $\mathcal{S}_k$. Then, the style-mixed images $\ddot{x}_i^{\mathcal{S}_k}$ is further forwarded to obtain the prediction $p_i^{\mathcal{S}_k}$. The same operation is also applied to the target images for the prediction $p_i^{\mathcal{T}}$. Finally, we exploit cross-entropy

[1] https://github.com/KaiyangZhou/Dassl.pytorch

Table A: Ablation study on FixMatch-CM.

| # | Methods | Avg |
|---|---|---|
| 1 | FixMatch-CM (MixStyle + CutMix) | 93.89 |
| 2 | FixMatch (MixStyle only) | 93.14 |
| 3 | FixMatch (vanilla) | 88.79 |
| 4 | Source only | 82.17 |

loss for the FixMatch-CM learning:

$$\begin{aligned}\arg\min_{\theta} L(\theta) = \frac{1}{KB}\sum_{k}^{K}\sum_{i}^{B}[\lambda_i^{\mathcal{S}_k} L_{ce}(p_i^{\mathcal{S}_k}, y_i^{\mathcal{S}_k}) \\ + (1-\lambda_i^{\mathcal{S}_k}) L_{ce}(p_i^{\mathcal{S}_k}, y^{\mathcal{D}_{k,i}})] \\ + \frac{1}{B}\sum_{i}^{B}[\mathbb{1}(q(\hat{y}_i^{\mathcal{T}}) \geq \tau_0)\lambda_i^{\mathcal{T}} L_{ce}(p_i^{\mathcal{T}}, \hat{y}_i^{\mathcal{T}}) \\ + (1-\lambda_i^{\mathcal{T}}) L_{ce}(p_i^{\mathcal{T}}, y^{\mathcal{D}_{T,i}})],\end{aligned} \tag{A}$$

where $q(\hat{y}_i^{\mathcal{T}})$ is the predicted probability corresponding to $\hat{y}_i^{\mathcal{T}}$. $\tau_0$ is a threshold to filter out low-confidence pseudo-labels. It is fixed as 0.95 same as FixMatch [12]. We focus on this first-step MSDA training method in our experiments.

### B.1 Ablation Study of FixMatch-CM

Figure 3 of the main paper shows that FixMatch repurposed for DA achieves the state of the art performance on PACS benchmark already. Our FixMatch-CM further improves it. To better understand FixMatch-CM, we investigate its each component. First, we discard the image-level strong augmentation – CutMix [13] and observe an 0.75% accuracy drop from 93.89% to 93.14% (c.f. #1 vs. #2 in Table A). Further removing MixStyle [15] from #2 leads to a 4.35% accuracy drop as we assume that vanilla FixMatch can only weakly deal with the domain shift.

## C Extra Training Cost of BORT$^2$

The major limitation of our BORT$^2$ is the extra training cost brought by the second-step training. To reduce the training cost, we further evaluate our BORT$^2$ on PACS by controlling the total training epochs. We keep the total training epoch exactly the same as one-step training MSDA methods, i.e., 100 epochs (50 epochs for the first-step labeling function training and the rest for the second-step target model training). From the results in Table B, we can see that reducing the total training epochs makes the performance of our BORT$^2$ worse, however, it still clearly outperforms the base method FixMatch-CM by $\sim 0.7\%$. These results not only show the efficacy of our proposed BORT$^2$ and demonstrate that our BORT$^2$ works even if the labeling function is not thoroughly trained in the first step.

Table B: Reducing the training cost for BORT$^2$.

| Methods | Avg |
|---|---|
| BORT$^2$ | 95.38 |
| BORT$^2$ (Control training epochs) | 94.57 |
| FixMatch-CM | 93.89 |

Table C: Comparison of using training set and held-out validation set for the outer loop optimization. Entropy loss denotes the Eq. (7) in main paper.

| Setting | Digit-Five | PACS | DomainNet |
|---|---|---|---|
| Feature uncertainty loss on training set | 95.9 | 95.38 | 53.4 |
| Feature uncertainty loss on validation set | 95.9 | 95.19 | 53.2 |
| Cross-entropy on validation set | 95.8 | 95.20 | 52.7 |

# D Further Analysis on Outer Loop Optimization

In Eq. (7) of the main paper, we minimize the entropy (or feature uncertainty) loss on current training set for the outer loop. Here we further conduct experiments to see 1) whether the current training set is better than a held-out validation set, and 2) whether the feature uncertainty loss as objective is better than pseudo-label based cross-entropy loss. We show the comparative results in Table C. We can see from the first two rows that minimizing feature uncertainty loss on a held-out validation set obtains slightly worse performance on PACS and DomainNet. This observation suggests that a held-out validation set is not necessary for the feature uncertainty minimization objective. When we use such validation set to calculate cross-entropy for the outer loop, the performance on DomainNet even decreases from 53.4% to 52.7%. We assume that the degradation is caused by the noise in the pseudo-labels. Overall, the entropy loss optimized on current training batch achieves the best performance over the other alternatives, and saves the labor for splitting a held-out validation set.

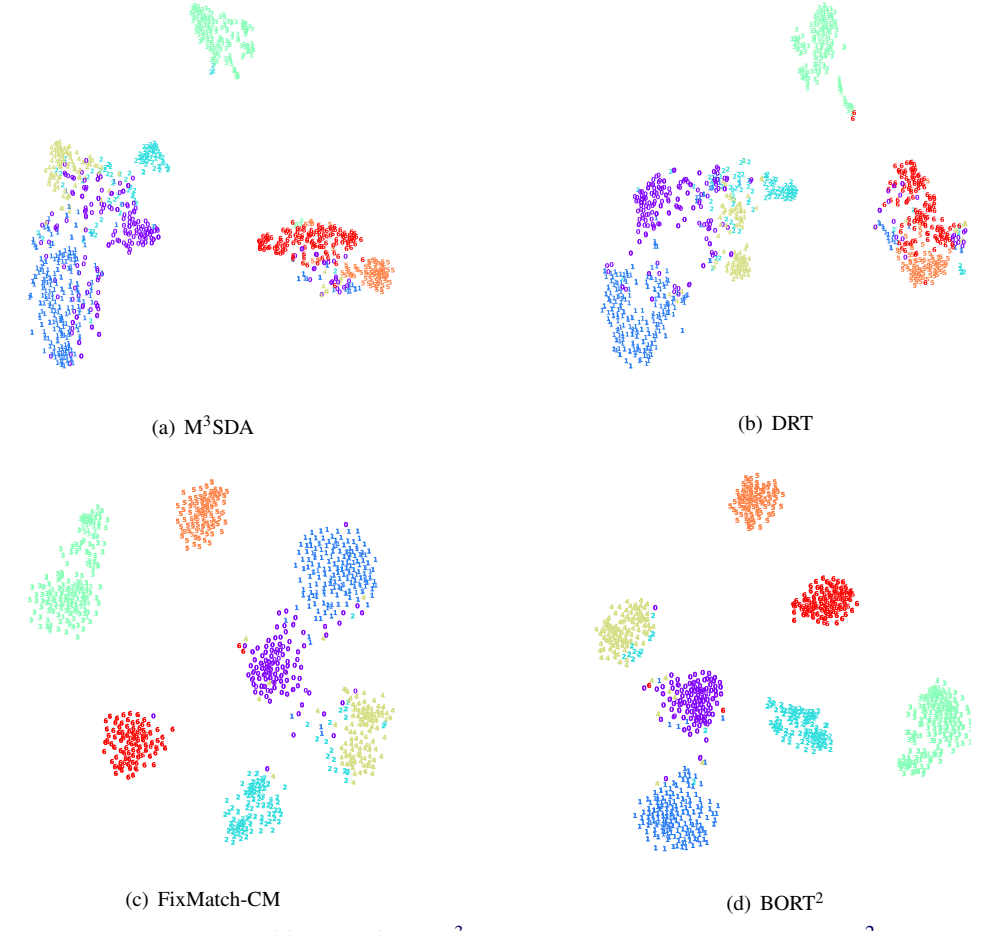
# E Ablation Study on the Adaptive Threshold

We practically adopt an adaptive threshold in Eq. (5) as an alternative to fixed threshold, e.g., $\tau = 0.95$ as in FixMatch [12]. We compare the adaptive threshold with the fixed threshold in Table D. We can observe that the adaptive threshold works better than the fixed threshold. This is possibly because the adaptive threshold gradually includes more and more samples for training in an easy-to-hard way, leading to a curriculum learning strategy. Moreover, we found that different MSDA datasets usually need different $\tau$ for optimal model performance, e.g., fixing $\tau$ to 0.95 works well on PACS but causes poor performance on DomainNet (accuracy$\approx$36%). Therefore, with this adaptive threshold a hyper parameter tuning is perfectly saved.

Considering that class imbalance can happen, we also try using different thresholds for

Table D: Ablation study on adaptive threshold.

| Methods | Avg |
|---|---|
| BORT$^2$ (fixed $\tau = 0.95$) | 95.21 |
| BORT$^2$ (adaptive $\tau$ for each class) | 95.36 |
| BORT$^2$ (adaptive $\tau$) | 95.38 |



(a) M$^3$SDA (b) DRT

(c) FixMatch-CM (d) BORT$^2$

Figure A: Visualization of features from M$^3$SDA, DRT, FixMatch-CM and BORT$^2$ on PACS (target domain: Sketch) using t-SNE [7]. Different colors (or digits 0-6) denote different classes. Better viewed with zoom-in.

different classes, i.e., for each class $c$, we set an adaptive threshold $\tau_c$ for that class. Here, $\tau_c$ is updated via $\tau_c = \alpha\tau_c + (1-\alpha)(p^c_{mean} - p^c_{std})$, with $p^c_{mean}$ and $p^c_{std}$ denoting the mean and standard deviation of the predictions of class $c$ in a mini-batch. This alternative choice achieves 95.36% on PACS, similar to using a single threshold for all the classes (95.38%). This is probably because a single threshold can already pick a reasonable amount of samples in each class on PACS.

# F Visualization of Learned Features

To better understand how our BORT$^2$ works, we further provide a t-SNE visualization of feature distributions in Figure A. We can see that FixMatch-CM enables the better feature separability compared with M$^3$SDA and DRT, which illustrates the effectiveness of FixMatch-CM. Based on FixMatch-CM, our BORT$^2$ further increases its inter-class distance, leading

to the best class-wise separability. We attribute this to our two-step training pipeline which eliminates the source domain bias in our model.

## References


[1] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[4] Y Lecun and L Bottou. Gradient-based learning applied to document recognition. *IEEE*, 1998.

[5] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *CVPR*, 2017.

[6] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[7] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[8] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS-W*, 2011.

[9] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS-W*, 2017.

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[11] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019.

[12] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020.

[13] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019.

[14] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain adaptive ensemble learning. *arXiv preprint arXiv:2003.07325*, 2020.

[15] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *arXiv preprint arXiv:2104.02008*, 2021.