Safe Self-Refinement for Transformer-based Domain Adaptation
Supplementary Material

Tao Sun[1], Cheng Lu[2], Tianshuo Zhang[2], Haibin Ling[1]
[1]Stony Brook University, [2]XPeng Motors
{tao,hling}@cs.stonybrook.edu, luc@xiaopeng.com, tonyzhang2035@gmail.com

A. More Model and Training Details

Our implementation is based on the *timm* library[1]. We use ViT-B/16 [2] (*vit_base_patch16_224* in *timm*) and ViT-S/16 [2] (*vit_small_patch16_224* in *timm*) as the vision transformer backbones in the paper. Transformer weights are restored from the checkpoints released by official Google JAX implementation[2], which are obtained by first training on ImageNet-21k [7] and then fine-tuning on Image-1k [7, 8]. The classifier head consists of a bottleneck module (`Linear` → `BatchNorm1d` → `ReLU` → `Dropout(0.5)`) and a class predictor (`Linear` → `ReLU` → `Dropout(0.5)` → `Linear`). The domain discriminator has the same network structure as the class predictor except having only one output.

During the training procedure, images are first resized to 256 × 256 pixels, randomly flipped horizontally, and then randomly cropped and resized to 224 × 224 pixels. The only exception is for VisDA-2017 [6], where center-cropping of size 224 × 224 is used. During the test procedure, images are first resized to 256 × 256 pixels and then center-cropped to 224 × 224 pixels.

To train the model, we adopt mini-batch Stochastic Gradient Descent (SGD) with momentum of 0.9. Learning rate is scheduled as $lr = lr_0 * (1 + 1e^{-3} \cdot i)^{-0.75}$, where lr_0 is initial learning rate, i is training step. The learning rate of parameters of vision transformer backbone is set to be 1/10 of lr. Complete hyper-parameters used for our experiments are listed in Tab. 1. Note that the same hyper-parameters are used for source-only training and baseline methods whenever applicable.

B. More Analysis on Multi-layer Perturbation

Figure 1 provides additional results when adding the same amount of perturbation to each layer while not using safe training. As can be seen in the left figure, the best layer to apply perturbation varies across tasks. Besides, a layer that works for one task may fail on others. To see the importance of allowing gradient back-propagation for b_x^l (see Sec. 3.3 and Sec. 3.4 in the paper), the right figure shows that the model collapses when add perturbation to relatively deep layers while blocking the gradients of b_x^l.

Table 5 includes comparison results when adding the perturbation to raw input or a single layer ({0} or {4} or {8}) in our proposed SSRT method. As can be seen, perturbing raw input performs similarly to perturbing the 0-th transformer block. Besides, perturbing any single layer degrades the performance on some adaptations tasks. In contrast, multi-layer perturbation combines their merits and obtains the best results.

Table 1. Complete list of SSRT hyper-parameters used in the experiments.

	Office-31	Office-Home	VisDA-2017	DomainNet
α	0.3			
β	0.2			
ϵ	0.4			
T	1000			
L	4			
batch_size	64 (32 source images + 32 target images)			
center_crop	False	False	True	False
lr_0	0.001	0.004	0.002	0.004
max_iters	10k	20k	20k	40k
bottleneck_dim	1024	2048	1024	1024

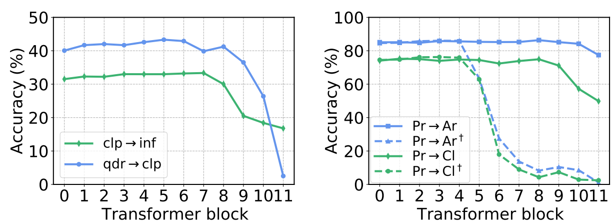

Figure 1. Perturbation at different layer. †No gradient back-propagation for b_x^l.

[1]https://github.com/rwightman/pytorch-image-models/blob/master/timm/models/vision_transformer.py

[2]https://github.com/google-research/vision_transformer

C. More Analysis on Bi-directional Self-Refinement

Table 2 provides additional results when blocking gradient back-propagation for different variables. Similar to the results listed in the paper (see Tab. 7), allowing gradient back-propagation of the teacher probabilities in KL divergence and b_x^l works better than other variants.

Table 2. Blocking gradient back-propagation for different variables. Note that $\boldsymbol{p}_x$ and $\tilde{\boldsymbol{p}}_x$ in the table only refer to the teacher probability in KL divergence. (Safe Training not applied)

	b_x^l	$\boldsymbol{p}_x$	$\tilde{\boldsymbol{p}}_x$	Cl→Ar	Cl→Pr	Cl→Rw
$\omega = 0$			×	1.61	12.71	6.08
$\omega = 1$		×		81.17	85.00	87.28
$\omega \sim \mathcal{B}(0.5)$		×	×	83.68	85.69	88.04
$\omega \sim \mathcal{B}(0.5)$	×			84.55	87.27	89.49
$\omega \sim \mathcal{B}(0.5)$				85.21	87.88	89.58

D. More Analysis on Safe Training

In our method, we adopt a *Confidence Filter* to remove noisy supervisions. If it not used (*i.e.*, $\epsilon = 0$), the performance may deteriorate. Table 3 shows that using Safe Training can avoid significant performance drops, making the method much safer.

Table 3. Accuracies (%) without Confidence Filter. (†Safe Training not applied)

	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw
Baseline-B	80.06	84.12	86.67	79.52	67.03	89.44
SSRT-B†	59.33	86.98	89.74	73.92	20.30	90.59
SSRT-B	84.51	86.98	89.30	82.65	67.79	91.16

E. Analysis on Model's Robustness

In our proposed SSRT, we use perturbed target domain data to refine the model during the training procedure. In this section, we provide analysis on model's robustness against perturbation during the test procedure. For each testing target domain data, we follow the same way as described in the paper to add a random offset to its latent token sequence, and use the perturbed token sequence to make prediction. To analyze model's robustness against perturbation at different layers, we add perturbation to different transformer block as well as the raw input. The perturbation magnitude is controlled by a scalar α as used in the paper. Figure 3 shows results (averaged over 6 random runs) on $Pr \rightarrow Ar$ and $clp \rightarrow pnt$. As can be seen, our method is more robust than Baseline. Even when adding a larger amount of perturbation ($\alpha = 0.4$) than seen during training, SSRT incurs less accuracy decrease.

F. Comparison with SSL methods

Since Unsupervised Domain Adaptation (UDA) is closely related to Semi-Supervised Learning (SSL), in this section, we compare our method with two representative techniques in SSL, *i.e.*, *Mixup* [11] and *VAT* [4].

Mixup regularizes the model to predict linearly between samples. Specifically, let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ be two target domain data, $p_1 = h(\boldsymbol{x}_1)$ and $p_2 = h(\boldsymbol{x}_2)$ be the corresponding model predictions, Mixup first interpolates between two samples by

$$\lambda \sim Beta(\alpha_\lambda, \alpha_\lambda) \tag{1}$$

$$\boldsymbol{x}' = \lambda \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{x}_2 \tag{2}$$

$$p' = \lambda p_1 + (1 - \lambda) p_2 \tag{3}$$

Its loss function is

$$\mathcal{L}_{\text{mixup}} = \mathbb{E}_{\boldsymbol{x}_1, \boldsymbol{x}_2 \sim \mathcal{D}_t} \| h(\boldsymbol{x}') - p' \|^2 \tag{4}$$

VAT enforces the model to predict consistently within the norm-ball neighborhood of each target data $\boldsymbol{x}$. Its loss function is

$$\mathcal{L}_{\text{VAT}} = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_t} \left[\max_{\|\boldsymbol{r}\| \le \rho} \mathrm{D}_{KL} \left(h(\boldsymbol{x}) \| h(\boldsymbol{x} + \boldsymbol{r}) \right) \right] \tag{5}$$

We use $\mathcal{L}_{\text{mixup}}$ and $\mathcal{L}_{\text{VAT}}$ as the $\mathcal{L}_{\text{tgt}}$ in our objective function. The trade-off parameter β is set to be 0.2 for both, same as used in our method. For Mixup, α_λ is set to be 0.5. We linearly ramp up β to its maximum value over 1/4 of all training steps as used in [1,9]. Instead of interpolating probabilities, we interpolate unnormalized logits, as it is shown to perform slightly better. For VAT, ρ is set to be 100. Both two techniques are applied to the raw input images.

Table 4 presents results on three benchmarks using ViT-base backbone. Detailed numbers can be found in Tables 5-7. On Office-Home [10] and VisDA-2017 [6], Mixup and VAT perform better than Baseline-B, and slightly worse than ours. On DomainNet [5], VAT still works. However, for Mixup, although we tried different hyper-parameters, it is still inferior to Baseline-B. Figure 2 shows two adaptations tasks where Mixup fails.

Table 4. Comparisons with SSL methods. $X^\dagger$ means averaged over all 5 tasks with X being the target domain.

	Office-Home	VisDA	Domain-Net	clp†	inf†	pnt†	qdr†	rel†	skt†
Baseline-B	81.1	85.2	38.5	50.6	25.6	44.9	11.6	57.0	41.5
Mixup-B	83.2	88.2	–	–	–	–	–	–	–
VAT-B	84.1	88.5	41.1	54.8	27.6	48.3	12.5	58.4	45.0
SSRT-B	85.4	88.8	45.2	60.0	28.2	53.3	13.7	65.3	50.4

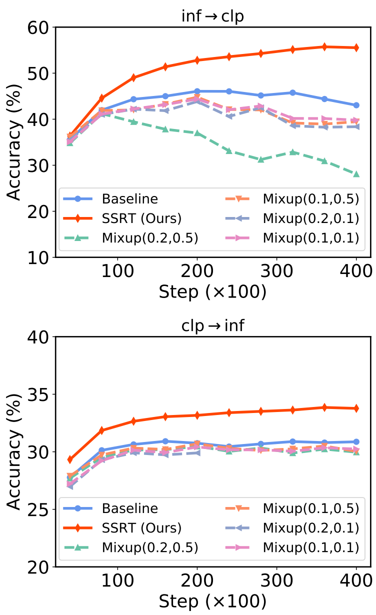

Figure 2. Mixup with different hyper-parameters. The legend for Mixup is formed as Mixup(β,α_λ).

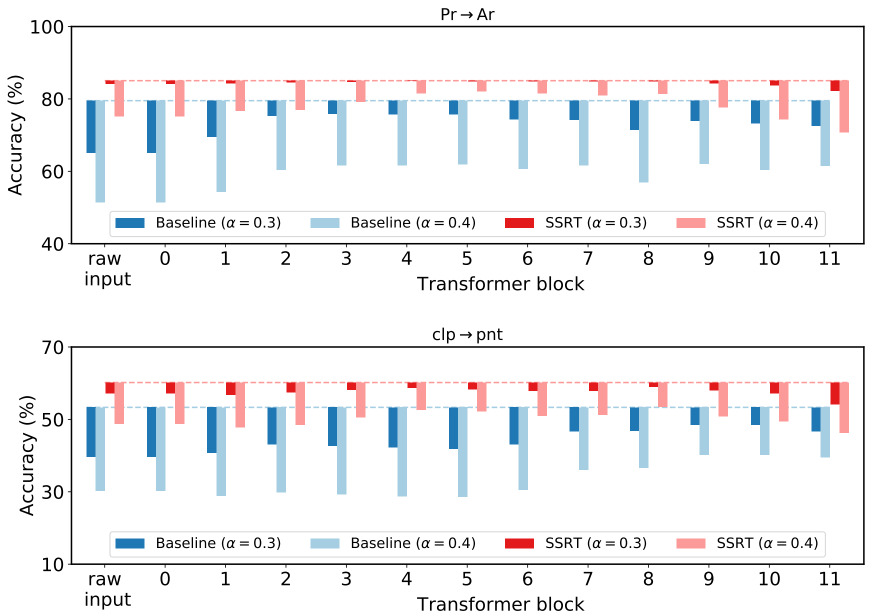

Figure 3. Analysis of model's robustness. The dashlines indicate true test accuracy on the target domain data. The bars show decreases of accuracies when adding perturbations to different layers during the test procedure.

Table 5. Accuracies (%) on **DomainNet**. In each sub-table, the column-wise means source domain and the row-wise means target domain. "-S/B" indicates ViT-small/base backbones, respectively.

MDD+SCDA [3]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	20.4	43.3	15.2	59.3	46.5	36.9
inf	32.7	-	34.5	6.3	47.6	29.2	30.1
pnt	46.4	19.9	-	8.1	58.8	42.9	35.2
qdr	31.1	6.6	18.0	-	28.8	22.0	21.3
rel	55.5	23.7	52.9	9.5	-	45.2	37.4
skt	55.8	20.1	46.5	15.0	56.7	-	38.8
Avg.	44.3	18.1	39.0	10.8	50.2	37.2	33.3

ViT-B	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

Baseline-B	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	30.9	53.3	16.3	72.7	55.4	45.7
inf	43.0	-	40.8	7.8	56.4	35.9	36.8
pnt	55.7	28.6	-	7.4	70.5	48.3	42.1
qdr	25.5	5.2	9.7	-	15.5	17.1	14.6
rel	62.3	32.5	62.5	8.2	-	50.7	43.2
skt	66.4	30.6	58.0	18.1	70.1	-	48.6
Avg.	50.6	25.6	44.9	11.6	57.0	41.5	38.5

VAT-B [4]	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	33.1	57.1	19.5	75.8	59.8	49.0
inf	48.3	-	45.2	9.8	55.0	37.4	39.2
pnt	60.0	30.9	-	7.9	71.1	52.6	44.5
qdr	26.7	5.4	9.2	-	18.1	18.3	15.5
rel	68.7	35.3	65.0	7.8	-	56.8	46.7
skt	70.2	33.3	65.0	17.6	72.2	-	51.7
Avg.	54.8	27.6	48.3	12.5	58.4	45.0	41.1

SSRT-B raw input	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	32.7	60.0	19.0	75.3	59.8	49.3
inf	55.0	-	54.0	8.9	67.8	48.1	46.8
pnt	61.6	28.6	-	8.2	71.3	55.4	45.0
qdr	36.3	6.2	16.1	-	32.1	31.2	24.4
rel	69.8	35.6	66.1	12.4	-	59.2	48.6
skt	70.3	30.5	62.3	20.0	73.2	-	51.3
Avg.	58.6	26.7	51.7	13.7	63.9	50.8	44.2

SSRT-B {0}	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	33.2	59.7	19.6	75.3	58.7	49.3
inf	54.8	-	53.5	9.3	67.7	46.1	46.3
pnt	61.2	29.0	-	7.1	71.2	55.0	44.7
qdr	40.8	7.0	13.2	-	35.4	31.1	25.5
rel	69.6	35.7	65.7	10.7	-	58.7	48.1
skt	69.7	32.1	62.0	19.0	72.8	-	51.1
Avg.	59.2	27.4	50.8	13.1	64.5	49.9	44.2

SSRT-B {4}	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	31.8	58.9	17.8	75.7	59.4	48.7
inf	53.5	-	50.5	8.6	67.8	47.5	45.6
pnt	61.3	29.2	-	8.1	71.3	54.3	44.8
qdr	42.5	7.7	17.0	-	23.3	33.4	24.8
rel	68.7	36.1	65.5	8.2	-	57.6	47.2
skt	70.1	31.8	62.2	17.7	73.1	-	51.0
Avg.	59.2	27.3	50.8	12.1	62.2	50.4	43.7

SSRT-B {8}	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	32.4	59.0	18.6	75.6	59.9	49.1
inf	55.9	-	54.8	7.6	68.5	48.2	47.0
pnt	61.5	27.4	-	8.5	71.4	54.6	44.7
qdr	33.6	5.7	11.3	-	31.4	31.8	22.7
rel	69.6	36.2	65.9	6.9	-	58.1	47.3
skt	69.9	30.9	62.3	19.8	73.3	-	51.2
Avg.	58.1	26.5	50.6	12.3	64.0	50.5	43.7

SSRT-B {0,4,8}	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	33.8	60.2	19.4	75.8	59.8	49.8
inf	55.5	-	54.0	9.0	68.2	44.7	46.3
pnt	61.7	28.5	-	8.4	71.4	55.2	45.0
qdr	42.5	8.8	24.2	-	37.6	33.6	29.3
rel	69.9	37.1	66.0	10.1	-	58.9	48.4
skt	70.6	32.8	62.2	21.7	73.2	-	52.1
Avg.	60.0	28.2	53.3	13.7	65.3	50.4	45.2

ViT-S	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	23.0	46.2	11.9	66.3	46.2	38.7
inf	42.9	-	42.8	3.8	62.3	33.9	37.1
pnt	45.2	22.2	-	3.5	66.5	35.7	34.6
qdr	19.7	3.3	7.8	-	14.6	12.7	11.6
rel	50.8	24.2	54.2	4.6	-	37.3	34.2
skt	57.2	19.5	47.1	13.9	62.5	-	40.0
Avg.	43.1	18.5	39.6	7.5	54.4	33.2	32.7

Baseline-S	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	27.0	49.0	12.8	68.2	49.1	41.2
inf	41.8	-	43.1	2.7	63.0	33.0	36.7
pnt	48.8	25.7	-	3.1	67.0	40.8	37.1
qdr	21.8	5.8	9.6	-	15.3	15.2	13.5
rel	54.6	28.7	57.5	3.6	-	41.3	37.1
skt	60.9	26.2	53.9	10.6	67.5	-	43.8
Avg.	45.6	22.7	42.6	6.5	56.2	35.9	34.9

SSRT-S	clp	inf	pnt	qdr	rel	skt	Avg.
clp	-	28.5	53.1	12.1	69.9	52.1	43.1
inf	47.5	-	49.8	1.5	64.9	39.7	40.7
pnt	53.0	26.5	-	4.4	67.3	46.7	39.6
qdr	31.3	6.9	13.0	-	24.4	24.0	19.9
rel	60.0	31.2	60.5	4.6	-	48.5	41.0
skt	63.8	28.6	57.0	13.7	68.7	-	46.4
Avg.	51.1	24.4	46.7	7.3	59.0	42.2	38.4

Table 6. Accuracies (%) on **Office-Home**.

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.
Baseline-B	66.96	85.74	88.07	80.06	84.12	86.67	79.52	67.03	89.44	83.64	70.15	91.17	81.05
Mixup-B [11]	71.32	86.66	88.82	82.45	84.79	87.58	82.90	71.68	90.77	85.46	74.36	91.37	83.18
VAT-B [4]	71.52	**89.39**	90.48	**86.11**	**88.53**	89.33	84.59	72.23	90.84	**86.61**	72.83	**92.48**	84.58
SSRT-B (ours)	**75.17**	88.98	**91.09**	85.13	88.29	**89.95**	**85.04**	**74.23**	**91.26**	85.70	**78.58**	91.78	**85.43**

Table 7. Accuracies (%) on **VisDA-2017**.

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Avg.
Baseline-B	98.55	82.59	85.97	57.07	94.93	97.20	94.58	76.68	92.11	96.54	94.31	52.24	85.23
Mixup-B [11]	98.88	86.56	88.64	72.32	98.06	98.07	95.91	**83.00**	94.09	98.07	94.55	50.36	88.21
VAT-B [4]	**99.15**	**87.71**	**90.85**	67.81	**98.81**	98.17	**97.57**	76.65	92.88	**98.73**	**96.27**	**57.37**	88.50
SSRT-B (ours)	98.93	87.60	89.10	**84.77**	98.34	**98.70**	96.27	81.08	**94.86**	97.90	94.50	43.13	**88.76**

G. Results with ViT-small Backbone

ViT-small is a smaller version of ViT-base by halving the number of Self-Attention Heads and token embedding dimension of ViT-base. It has fewer parameters (∼22M params) than ResNet-101 (∼45M params). We empirically found that it convergences much slower than ViT-base, so we double the maximum training iterations. An alternative is to pretrain on the source data first and then adapted to the target data. As can be seen from Tab. 5, our proposed SSRT-S achieves +5.1% higher accuracy than MDD+SCDA (ResNet-101 backbone) on DomainNet, despite that ViT-small has fewer parameters than ResNet-101.

References

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 2

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 1

[3] Shuang Li, Mixue Xie, Fangrui Lv, Chi Harold Liu, Jian Liang, Chen Qin, and Wei Li. Semantic concentration for domain adaptation. In *ICCV*, pages 9102–9111, 2021. 3

[4] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *TPAMI*, 41(8):1979–1993, 2018. 2, 3, 4

[5] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, pages 1406–1415, 2019. 2

[6] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 1, 2

[7] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 1

[8] Andreas Steiner, Alexander Kolesnikov, , Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021. 1

[9] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017. 2

[10] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5018–5027, 2017. 2

[11] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: beyond empirical risk minimization. In *ICLR*, 2018. 2, 4