SUPPLEMENTARY MATERIAL
Learning Class and Domain Augmentations for Single-Source Open-Domain Generalization

Prathmesh Bele[1] Valay Bundele[1*] Avigyan Bhattacharya[1*] Ankit Jha[1] Gemma Roig[2] Biplab Banerjee[1]
[1]Indian Institute of Technology Bombay, India [2]Goethe University Frankfurt, Germany
prathmeshbele.iitb@gmail.com, valaybundele@gmail.com, avigyanbhattacharya123@gmail.com
ankitjha16@gmail.com, roig@cs.uni-frankfurt.de, getbiplab@gmail.com

In this supplementary material, we present a comprehensive dataset description, elaborate model architecture, algorithm details, and tables showcasing the accuracies of both known and unknown classes obtained from our experiments.

1. Dataset Description

(1) **Office-31** [10]: This dataset comprises 31 classes obtained from three distinct domains: Amazon, DSLR, and Webcam, totaling 4652 images. In our experiments, we consider the 10 classes shared by Office-31 and Caltech-256 [3] (backpack, bike, calculator, headphones, keyboard, laptop, monitor, mouse, mug, and projector) as the source domain label space, following the suggestion of [9]. The remaining eleven classes in alphabetical order (ruler, punchers, stapler, scissors, trash can, tape dispenser, pen, phone, printer, ring binder, and speaker) constitute the target unknown class space. Due to the relatively fewer number of samples in the DSLR and Webcam domains, we conduct experiments solely on the Amazon domain as the source domain.

(2) **Digits**: This dataset consists of five digit datasets: **MNIST** [6], **SVHN** [8], **USPS** [4], **MNIST-M**, and **SYN** [2]. In our setup, **MNIST** serves as the source domain with known classes representing numbers from 0 to 4, while the other datasets are considered as target domains, representing unknown classes for numbers 5 to 9. We select 10,000 images from the MNIST dataset, following the approach of [13] and [15], for the source domain.

(3) **Office-Home** [12]: This dataset comprises data from four different domains: Art, Clipart, Product, and Real-World, totaling 15,500 images. Each domain consists of 65 classes, with the first 15 classes (alarm clock, backpack, battery, bed, bike, bottle, bucket, calculator, calendar, candles, chair, clipboards, computer, couch, and curtains) used as the source label space, while the remaining 50 classes are considered as unknown target classes.

(4) **PACS** [7]: This dataset contains 9,991 images from four domains: Art Painting, Cartoon, Photo, and Sketch. Each domain includes images from seven different classes. In our setup, we utilize four classes (dog, elephant, giraffe, and guitar) as the label space in the source domain, while the remaining three classes (horse, house, and person) are treated as unknown classes in the target domains.

2. Model Architecture

The detailed architectures of each of the networks the style synthesis block $\mathcal{F}_{ss}$, the feature aggregation block $\mathcal{F}_{fa}$, g^l, $g^{l+1:L}$ are given in the tables 1, 2, 3, 4, respectively. In our experiments, we have taken the encoder network to be ResNet18 and g^l consists of first five convolutional blocks while $g^{l+1:L}$ is rest of the network i.e., ResNet18 $= g^{l+1:L} \circ g^l$. We also conduct experiments by changing the point of extraction of feature map to apply the style synthesis block. Table 5 represents the architecture of g^l which was shallower as compared to the one in 3 while table 6 shows the deeper version. In both the cases, rest of the ResNet18 was used as $g^{l+1:L}$. Results of these experiments are given in table 7. The '-1' in output shapes is a placeholder for the batch-size.

Table 1. Architecture Summary of $\mathcal{F}_{ss}$

Layer (type)	Output Shape	Param #
Input Shape	$[-1, 256]$	0
Linear-1	$[-1, 192]$	49,344
ReLU-2	$[-1, 192]$	0
Linear-3	$[-1, 128]$	24,704
ReLU-4	$[-1, 128]$	0
Total parameters: 74,048		
Trainable parameters: 74,048		
Non-trainable parameters: 0		

*equal contribution

Table 2. Architecture Summary of $\mathcal{F}_{fa}$

Layer (type)	Output Shape	Param #
Input Shape	$[-1, 1024]$	0
Linear-1	$[-1, 512]$	524,800
ReLU-2	$[-1, 512]$	0
BatchNorm1d-3	$[-1, 512]$	1,024
Linear-4	$[-1, 512]$	262,656
Sigmoid-5	$[-1, 512]$	0
Total parameters: 788,480		
Trainable parameters: 788,480		
Non-trainable parameters: 0		

Table 3. Architecture Summary of g^l

Layer (type)	Output Shape	Param #
Input Shape	$[-, 3, 128, 128]$	0
Conv2d-1	$[-1, 64, 64, 64]$	9,408
BatchNorm2d-2	$[-1, 64, 64, 64]$	128
ReLU-3	$[-1, 64, 64, 64]$	0
MaxPool2d-4	$[-1, 64, 32, 32]$	0
Conv2d-5	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-6	$[-1, 64, 32, 32]$	128
ReLU-7	$[-1, 64, 32, 32]$	0
Conv2d-8	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-9	$[-1, 64, 32, 32]$	128
ReLU-10	$[-1, 64, 32, 32]$	0
BasicBlock-11	$[-1, 64, 32, 32]$	0
Conv2d-12	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-13	$[-1, 64, 32, 32]$	128
ReLU-14	$[-1, 64, 32, 32]$	0
Conv2d-15	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-16	$[-1, 64, 32, 32]$	128
ReLU-17	$[-1, 64, 32, 32]$	0
BasicBlock-18	$[-1, 64, 32, 32]$	0
Total parameters: 157,504		
Trainable parameters: 157,504		
Non-trainable parameters: 0		

3. Experiments on the large scale dataset DomainNet

The table 14 contains the results on *DomainNet* dataset. We had chosen four domains out of six from the dataset (Clipart, Painting, Sketch and Real). For our experiments, we selected alphabetically first 150 classes from each of the four domains and remaining 195 classes were treated as unknown target class. The total number of samples corresponding to these four domain is 362470. We compare our results against two bechmark methods, ERM [5] and ADA [13]. Here, we outperform the ADA by 11.17% for average accuracy (acc) and by 3.15% while compared to the h-score (hs).

4. Experimental Results with Known and Unknown Class Accuracies

In this section we report the known and unknown class accuracies (acc_k and acc_u) for the experiments conducted. The table 8 has the results for *Office31* and *Digits* datasets while table 9 and 10 have the results for *Office-Home* and *PACS* datasets, respectively.

Table 4. Architecture Summary of $g^{l+1:L}$

Layer (type)	Output Shape	Param #
Input Shape	$[-1, 64, 32, 32]$	0
Conv2d-1	$[-1, 128, 16, 16]$	73,728
BatchNorm2d-2	$[-1, 128, 16, 16]$	256
ReLU-3	$[-1, 128, 16, 16]$	0
Conv2d-4	$[-1, 128, 16, 16]$	147,456
BatchNorm2d-5	$[-1, 128, 16, 16]$	256
Conv2d-6	$[-1, 128, 16, 16]$	8,192
BatchNorm2d-7	$[-1, 128, 16, 16]$	256
ReLU-8	$[-1, 128, 16, 16]$	0
BasicBlock-9	$[-1, 128, 16, 16]$	0
Conv2d-10	$[-1, 128, 16, 16]$	147,456
BatchNorm2d-11	$[-1, 128, 16, 16]$	256
ReLU-12	$[-1, 128, 16, 16]$	0
Conv2d-13	$[-1, 128, 16, 16]$	147,456
BatchNorm2d-14	$[-1, 128, 16, 16]$	256
ReLU-15	$[-1, 128, 16, 16]$	0
BasicBlock-16	$[-1, 128, 16, 16]$	0
Conv2d-17	$[-1, 256, 8, 8]$	294,912
BatchNorm2d-18	$[-1, 256, 8, 8]$	512
ReLU-19	$[-1, 256, 8, 8]$	0
Conv2d-20	$[-1, 256, 8, 8]$	589,824
BatchNorm2d-21	$[-1, 256, 8, 8]$	512
Conv2d-22	$[-1, 256, 8, 8]$	32,768
BatchNorm2d-23	$[-1, 256, 8, 8]$	512
ReLU-24	$[-1, 256, 8, 8]$	0
BasicBlock-25	$[-1, 256, 8, 8]$	0
Conv2d-26	$[-1, 256, 8, 8]$	589,824
BatchNorm2d-27	$[-1, 256, 8, 8]$	512
ReLU-28	$[-1, 256, 8, 8]$	0
Conv2d-29	$[-1, 256, 8, 8]$	589,824
BatchNorm2d-30	$[-1, 256, 8, 8]$	512
ReLU-31	$[-1, 256, 8, 8]$	0
BasicBlock-32	$[-1, 256, 8, 8]$	0
Conv2d-33	$[-1, 512, 4, 4]$	1,179,648
BatchNorm2d-34	$[-1, 512, 4, 4]$	1,024
ReLU-35	$[-1, 512, 4, 4]$	0
Conv2d-36	$[-1, 512, 4, 4]$	2,359,296
BatchNorm2d-37	$[-1, 512, 4, 4]$	1,024
Conv2d-38	$[-1, 512, 4, 4]$	131,072
BatchNorm2d-39	$[-1, 512, 4, 4]$	1,024
ReLU-40	$[-1, 512, 4, 4]$	0
BasicBlock-41	$[-1, 512, 4, 4]$	0
Conv2d-42	$[-1, 512, 4, 4]$	2,359,296
BatchNorm2d-43	$[-1, 512, 4, 4]$	1,024
ReLU-44	$[-1, 512, 4, 4]$	0
Conv2d-45	$[-1, 512, 4, 4]$	2,359,296
BatchNorm2d-46	$[-1, 512, 4, 4]$	1,024
ReLU-47	$[-1, 512, 4, 4]$	0
BasicBlock-48	$[-1, 512, 4, 4]$	0
AdaptiveAvgPool2d-49	$[-1, 512, 1, 1]$	0
Identity-50	$[-1, 512, 1, 1]$	0
Total parameters: 11,019,008		
Trainable parameters: 11,019,0080		
Non-trainable parameters: 0		

Table 5. Architecture Summary of shallow g^l

Layer (type)	Output Shape	Param #
Input Shape	$[-, 3, 128, 128]$	0
Conv2d-1	$[-1, 64, 64, 64]$	9,408
BatchNorm2d-2	$[-1, 64, 64, 64]$	128
ReLU-3	$[-1, 64, 64, 64]$	0
MaxPool2d-4	$[-1, 64, 32, 32]$	0
Total parameters: 9,536		
Trainable parameters: 9,536		
Non-trainable parameters: 0		

5. Ablation Studies

Comparison of $\mathcal{F}_{ss}$ with MixStyle [16]: We conducted the experiments by replacing the style synthesis block $\mathcal{F}_{ss}$ with

Table 6. Architecture Summary of deep g^l

Layer (type)	Output Shape	Param #
Input Shape	$[-, 3, 128, 128]$	0
Conv2d-1	$[-1, 64, 64, 64]$	9,408
BatchNorm2d-2	$[-1, 64, 64, 64]$	128
ReLU-3	$[-1, 64, 64, 64]$	0
MaxPool2d-4	$[-1, 64, 32, 32]$	0
Conv2d-5	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-6	$[-1, 64, 32, 32]$	128
ReLU-7	$[-1, 64, 32, 32]$	0
Conv2d-8	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-9	$[-1, 64, 32, 32]$	128
ReLU-10	$[-1, 64, 32, 32]$	0
BasicBlock-11	$[-1, 64, 32, 32]$	0
Conv2d-12	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-13	$[-1, 64, 32, 32]$	128
ReLU-14	$[-1, 64, 32, 32]$	0
Conv2d-15	$[-1, 64, 32, 32]$	36,864
BatchNorm2d-16	$[-1, 64, 32, 32]$	128
ReLU-17	$[-1, 64, 32, 32]$	0
BasicBlock-18	$[-1, 64, 32, 32]$	0
Conv2d-19	$[-1, 128, 16, 16]$	73,728
BatchNorm2d-20	$[-1, 128, 16, 16]$	256
ReLU-21	$[-1, 128, 16, 16]$	0
Conv2d-22	$[-1, 128, 16, 16]$	147,456
BatchNorm2d-23	$[-1, 128, 16, 16]$	256
Conv2d-24	$[-1, 128, 16, 16]$	8,192
BatchNorm2d-25	$[-1, 128, 16, 16]$	256
ReLU-26	$[-1, 128, 16, 16]$	0
BasicBlock-27	$[-1, 128, 16, 16]$	0
Conv2d-28	$[-1, 128, 16, 16]$	147,456
BatchNorm2d-29	$[-1, 128, 16, 16]$	256
ReLU-30	$[-1, 128, 16, 16]$	0
Conv2d-31	$[-1, 128, 16, 16]$	147,456
BatchNorm2d-32	$[-1, 128, 16, 16]$	256
ReLU-33	$[-1, 128, 16, 16]$	0
BasicBlock-34	$[-1, 128, 16, 16]$	0
Total parameters: 683,072 Trainable parameters: 683,072 Non-trainable parameters: 0		

Table 7. Results on different depths of g^l on Office31 dataset

Metric	Shallow g^l Table 5	Our g^l Table 3	Deeper g^l Table 6
acc_k	70.64	**73.96**	75.35
acc_u	60.59	**83.91**	66.76
acc	65.53	**79.02**	70.98
hs	65.23	**78.62**	70.79

Table 8. acc_k & acc_u (% Accuracy) on **Office31** and **Digits** Dataset.

Method	Office31		Digits	
	acc_k	acc_u	acc_k	acc_u
OSDAP [11]	75.77	84.28	35.59	70.60
OpenMax [1]	10.01	100	34.40	83.81
ERM [5]	85.1	27.04	56.40	13.04
ERM+CM [17]	82.37	37.6	48.67	53.52
ADA [13]	85.62	25.24	57.24	15.11
ADA+CM [17]	53.02	34.51	49.24	52.07
MEADA [15]	85.78	25.09	57.61	29.83
MEADA+CM [17]	82.77	41.08	52.30	46.11
SODG-NET	73.96	83.91	43.60	70.45
SODG-NET $-\mathcal{L}_{disc}$	74.24	82.31	44.63	64.71
SODG-NET $-\mathcal{L}_{disc} - \mathcal{L}_{sm}$	73.13	78.55	41.97	67.11
SODG-NET $-\mathcal{L}_{sm}$	65.65	71.58	37.76	60.12

MixStyle method. The detailed results are shown in table 12 and table 11 for *Office31* and *PACS* dataset respectively. On *Office31* dataset, we beat the results with MixStyle by 4.63% and 4.84% while on the *PACS* dataset, on an average, we are outperforming the MixStyle by 4.55% and 4.35% in terms of acc and hs respectively.

Effects of changes in noise parameters in $\mathcal{F}_{ss}$: To see the effects of added Gaussian noise to $\mu_1, \sigma_1, \mu_2, \sigma_2$ before passing them into $\mathcal{F}_{ss}$, we experiment with different values of μ, σ for the Gaussian distribution ($\mathbb{N}(\mu, \sigma)$). Table 13 shows the results of experiments on *Office31* dataset.

6. Closed set domain generalization

In this section we provide the results on closed set single source domain generalization, that is, when the training and testing data label space is same. These experiments were conducted on two datasets, *Office31* and *PACS*. In case of the *Office31* dataset, we have conducted an ablation study with varying number of classes in the dataset and source domain as Amazon (see table 15). For the *PACS* dataset, each one of the four domains were taken as source domain for training and rest were considered as target. The average performance on the *PACS* dataset for each case is given in table 16. We compare performance of our style synthesis block against two baselines, one being the ERM [5] and another one from Wang *et al*. [14]. We observe that our method performs convincingly when compared with the above mentioned ones.

References

[1] Abhijit Bendale and Terrance E. Boult. Towards open set deep networks. *CoRR*, abs/1511.06233, 2015. 3, 4

[2] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1180–1189, Lille, France, 07–09 Jul 2015. PMLR. 1

[3] Boqing Gong, Yuan Shi, Fei Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. pages 2066–2073, 06 2012. 1

[4] J.J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994. 1

[5] Vladimir Koltchinskii. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems: École d'Été de Probabilités de Saint-Flour XXXVIII-2008*, volume 2033. 01 2011. 2, 3, 4, 5

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, 12 1989. 1

Table 9. acc_k & acc_u (% Accuracy) on **Office-Home** Dataset.

Method	Art		Clipart		Product		Real-World		Average	
	acc_k	acc_u	acc_k	acc_u	acc_k	acc_u	acc_k	acc_u	acc_k	acc_u
OSDAP [11]	44.13	67.84	51.69	69.26	40.00	63.47	52.48	66.92	47.07	66.87
OpenMax [1]	17.38	98.08	17.72	97.04	9.53	98.59	20.78	97.01	16.35	97.68
ERM [5]	68.54	20.53	66.75	24.65	62.81	26.26	69.48	23.18	66.90	23.66
ERM+CM [17]	66.48	48.57	64.80	41.95	59.17	40.94	69.36	43.69	64.95	43.79
ADA [13]	71.36	22.05	67.37	31.19	62.91	24.55	69.92	23.88	67.89	25.42
ADA+CM [17]	67.53	39.59	64.10	40.67	59.92	40.72	68.53	40.79	65.02	40.44
MEADA [15]	71.37	22.36	66.45	31.27	62.75	25.60	69.92	23.71	67.62	25.74
MEADA+CM [17]	66.63	45.28	64.43	37.84	59.74	37.71	68.82	41.28	64.90	40.53
SODG-NET	48.33	71.41	56.23	73.86	53.28	71.98	55.38	72.77	53.31	72.50
SODG-NET $-\mathcal{L}_{disc}$	46.62	71.64	58.81	66.96	53.47	70.49	54.61	72.57	53.38	70.41
SODG-NET $-\mathcal{L}_{disc} - \mathcal{L}_{sm}$	51.09	61.39	61.18	60.47	52.24	68.33	58.52	64.13	55.76	63.58

Table 10. acc_k & acc_u (% Accuracy) on **PACS** Dataset .

Method	Art Painting		Cartoon		Sketch		Photo		Average	
	acc_k	acc_u	acc_k	acc_u	acc_k	acc_u	acc_k	acc_u	acc_k	acc_u
OSDAP [11]	54.17	49.84	41.36	51.68	38.84	54.92	28.09	41.62	40.62	49.51
OpenMax [1]	42.87	91.48	15.27	97.44	13.16	96.61	11.96	90.22	20.82	93.94
ERM [5]	68.80	24.57	59.46	33.08	43.34	20.27	37.54	30.03	52.29	26.99
ERM+CM [17]	68.66	44.56	62.25	43.18	41.01	33.16	39.91	54.21	52.96	44.53
ADA [13]	70.95	28.80	62.08	33.83	43.18	22.41	40.65	38.77	54.22	30.93
ADA+CM [17]	72.93	40.12	64.39	49.06	44.98	40.85	43.27	52.53	56.40	45.64
MEADA [15]	70.90	28.65	62.09	33.55	43.42	22.90	39.78	40.31	54.05	31.35
MEADA+CM [17]	70.45	33.36	63.76	53.74	40.25	48.79	42.89	50.57	54.34	46.61
SODG-NET	49.10	69.17	49.77	65.65	48.19	74.09	32.02	71.68	44.77	70.15
SODG-NET $-\mathcal{L}_{disc}$	46.22	72.15	52.79	59.45	45.30	77.95	37.34	51.93	45.41	65.37
SODG-NET $-\mathcal{L}_{disc} - \mathcal{L}_{sm}$	49.70	64.02	47.53	55.49	43.66	76.02	31.89	53.86	43.19	62.35

Table 11. Comparison of $\mathcal{F}_{ss}$ with MixStyle on **PACS** dataset

Metric	Art Painting		Cartoon		Sketch		Photo		Average	
	$\mathcal{F}_{ss}$	MixStyle	$\mathcal{F}_{ss}$	MixStyle	$\mathcal{F}_{ss}$	MixStyle	$\mathcal{F}_{ss}$	MixStyle	$\mathcal{F}_{ss}$	MixStyle
acc_k	**49.10**	44.44	49.77	**56.41**	**48.19**	45.36	32.02	**34.98**	44.77	**45.30**
acc_u	**69.17**	67.78	**65.65**	46.65	**74.09**	67.17	**71.68**	46.04	**70.15**	56.91
acc	**57.02**	53.61	**56.01**	52.57	**58.36**	53.93	**46.27**	39.32	**54.41**	49.86
hs	**57.44**	53.69	**56.62**	51.07	**58.40**	54.16	**43.60**	39.75	**54.02**	49.67

Table 12. Comparison of $\mathcal{F}_{ss}$ with MixStyle on **Office31** dataset

Metric	$\mathcal{F}_{ss}$	MixStyle
acc_k	**73.96**	67.14
acc_u	**83.91**	80.43
acc	**79.02**	74.39
hs	**78.62**	73.78

Table 13. Effects of changes in noise parameters in $\mathcal{F}_{ss}$

Metric	$\mathbb{N}(0,1)$	$\mathbb{N}(1,1)$	$\mathbb{N}(0,2)$	$\mathbb{N}(0,3)$
acc_k	**73.96**	60.66	73.41	72.3
acc_u	**83.91**	81.5	74.53	80.7
acc	**79.02**	71.25	73.98	76.57
hs	**78.62**	69.56	73.96	76.27

[7] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization. *CoRR*, abs/1710.03077, 2017. 1

[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 *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. 1

[9] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[10] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 213–226, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 1

[11] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 3, 4

[12] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE*

Table 14. Results (% Accuracy) on **DomainNet** Dataset for different source domains.

Method	Clipart		Painting		Sketch		Real		Average	
	acc	hs	acc	hs	acc	hs	acc	hs	acc	hs
ERM [5]	27.21	19.65	23.14	16.27	35.83	28.69	40.18	38.20	31.59	25.70
ADA [13]	32.42	30.60	36.65	31.52	38.97	**31.23**	41.26	40.65	37.32	33.50
SODG-NET	**47.10**	**34.21**	**49.97**	**34.64**	**47.22**	27.55	**49.68**	**46.19**	**48.49**	**35.65**

Table 15. Closed set domain generalization on **Office31** dataset with Amazon as source domain with varying number of classes

Office-31				
Number of Classes	Amazon	DSLR	Webcam	Overall
31	79.49	50.75	49.28	69.9
25	82.53	51.92	52.63	72.8
20	89.25	59.76	59.12	79.56
15	89.22	76.92	62.5	82.42
10	94.74	77.5	72.13	88.06

Table 16. Closed set single source domain generalization results on **PACS** dataset

Method	Source Domain				
	Art	Cartoon	Photo	Sketch	Average
ERM [5]	44.72	41.96	38.17	29.24	38.52
Wang *et al.* [14]	**61.28**	66.35	45.17	39.51	53.07
Our (SSB)	60.30	**69.41**	**46.23**	**42.79**	**54.68**

Conference on Computer Vision and Pattern Recognition (CVPR), July 2017. 1

[13] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. *Advances in neural information processing systems*, 31, 2018. 1, 2, 3, 4, 5

[14] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 834–843, 2021. 3, 5

[15] Long Zhao, Ting Liu, Xi Peng, and Dimitris Metaxas. Maximum-entropy adversarial data augmentation for improved generalization and robustness. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14435–14447. Curran Associates, Inc., 2020. 1, 3, 4

[16] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *arXiv preprint arXiv:2104.02008*, 2021. 2

[17] Ronghang Zhu and Sheng Li. Crossmatch: Cross-classifier consistency regularization for open-set single domain generalization. In *International Conference on Learning Representations*, 2022. 3, 4