Supplementary Materials
Open Domain Generalization with Domain-Augmented Meta-Learning

Yang Shu*, Zhangjie Cao*, Chenyu Wang, Jianmin Wang, Mingsheng Long (✉)
School of Software, BNRist, Tsinghua University, China
{shu-y18,caozj14,cy-wang18}@mails.tsinghua.edu.cn, {jimwang,mingsheng}@tsinghua.edu.cn

In the supplementary materials, we will provide more details on the implementation and more experiment results.

1. Experiment Details

In this section, we clarify more details of the experiment settings due to the space limit in the main text.

1.1. Datasets

For each dataset, we show the exact class splits for each domain.

PACS [5] dataset consists of four domains corresponding to four different image styles, including photo (**P**), art-painting (**A**), cartoon (**C**) and sketch (**S**). The four domains have the same label set of 7 classes. We assign an index to each category, 0-Dog, 1-Elephant, 2-Giraffe, 3-Guitar, 4-Horse, 5-House, 6-Person. We use each domain as the target domain and the other three domains as source domains to form four cross-domain tasks: CPS-A, PAC-S, ACS-P, SPA-C. To construct the open-domain situations, we split the label space of the dataset, resulting in various label spaces across different domains. The specific categories contained in each domain are shown in Table 1.

Table 1. Open-domain split of PACS dataset.

Domain	Classes
Source-1	$3, 0, 1$
Source-2	$4, 0, 2$
Source-3	$5, 1, 2$
Target	$0, 1, 2, 3, 4, 5, 6$

Office-Home [10] comprises of images from four different domains: Artistic (**Ar**), Clip art (**Cl**), Product (**Pr**) and Real-world (**Rw**). It has a large domain gap and 65 classes which is much more than other DG datasets, so it is very challenging. Similar to the PACS dataset, we spread these 65 classes among the four domains to derive an open-domain dataset and construct four open generalization tasks based on it: ArPrRw-Cl, ArClPr-Rw, ArClRw-Pr, ClPrRw-Ar, where each domain is used as the target domain respectively, and the other three domains serve as source domains. With more classes, it is possible to construct more complicated open-domain situations compared with PACS dataset. The categories contained in each domain are shown in Figure 1.

*Equal contribution.

Table 2. Open-domain split of Office-Home dataset.

Domain	Classes
Source-1	$0-2, 3-8, 9-14, 21-31$
Source-2	$0-2, 3-8, 15-20, 32-42$
Source-3	$0-2, 9-14, 15-20, 43-53$
Target	$0, 3-4, 9-10, 15-16,$ $21-23, 32-34, 43-45, 54-64$

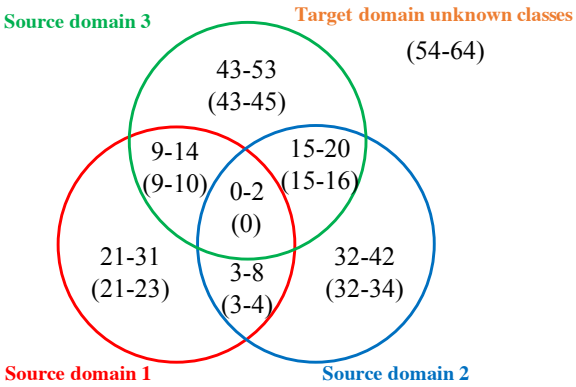


Figure 1. Illustration of the open-domain split of Office-Home Dataset. Indices without brackets show the distribution of categories among source domains, while indices in brackets indicate the categories of the target domain.

Table 3. Class details in Multi-Datasets.

Domain	Classes
Office-31	$0-30$
Visda	$1, 31-41$
STL-10	$31, 33, 34, 41, 42-47$
DomainNet	$0, 1, 5, 6, 10, 11, 14, 17, 20, 26$ $31-36, 39-43, 45-46, 48-67$

Multi-Datasets scenario is constructed in this paper to consider a more realistic situation of learning generalizable representations from arbitrary source domains. We simulate the process where we obtain source domains from different resources and try to learn a generalizable model to achieve high accuracy on an unseen target domain. We leverage several public datasets including **Office-31** [9], **STL-10** [1] and **Visda2017** [8] as source domains, and evaluate the generalization performance on **DomainNet** [7]. In Office-31, we use the Amazon domain, which consists of 31 classes of office environment objects, and the images are downloaded from online merchants, which is a very popular way to acquire data. STL-10 is composed of 10 classes for general object recognition, and we use its labeled data as one of the source domains. Visda2017 dataset forms a simulation-to-real situation. We leverage its training data as the source domain, which contains synthetic images of 12 classes from CAD models. DomainNet is a new benchmark for evaluating cross-domain generalization performance. We use its four domains: Clip art, Real, Painting and Sketch as the target domains. For DomainNet, we preserve all the 23 classes existing in the joint label set of source domains and randomly sample 20 other classes as unknown classes, since there are too many open classes in it. Note that there exist huge distribution discrepancy and label-set disparity across the four datasets, which forms a natural open-domain generalization scenario.

1.2. Implementation

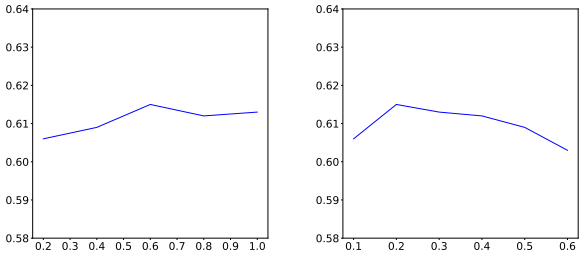
We implement our algorithm in PyTorch [6]. We use ResNet-18 [4] pre-trained on ImageNet as the backbone network and train our model for 30 epochs with SGD as the optimization algorithm. For the proposed DAML, similar to [2], we use fast first-order approximation to estimate gradients. To enable open-class detection for non-open-set methods, we set a confidence threshold T on the prediction, where T is selected similar to the open set recognition method [3], by sorting the confidence on the source validation data, and then picking a certain percentile. The initial learning rate β is 0.001, and is decayed after 24 epochs by a factor of 10. In PACS dataset, we follow the protocol in [5] for train and validation split. In other datasets, we randomly select 10% data in each category of the source domains as their validation sets. We tune the hyper-parameters and choose the models for test on the held-out validation sets. We choose the step for inner update of meta-training $\eta = 0.01$, and the parameters for Dirichlet mixup $\alpha_{\text{max}} = 0.6, \alpha_{\text{min}} = 0.2$. For DAML and all the compared methods, we use the same basic data preprocessing on the image and the same backbone. We run each experiment 3 times and compute the average and the standard deviation.

2. Computing Infrastructure

We use PyTorch 1.5, torchvison 0.6 and CUDA 10 libraries. We use a machine with 32 CPUs, 256 GB memory and one NVIDIA TITAN X. The average training time for each run is 2 hours.

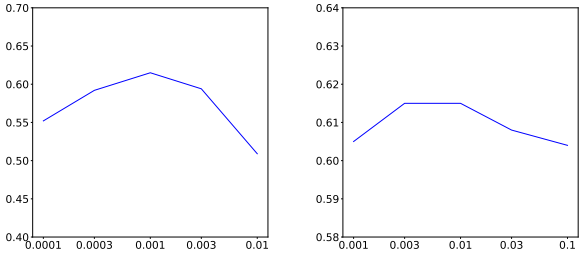
3. Experiment Results

In this section, we provide more experiment results, including the sensitivity of hyperparameters, the results of different classes, the effect of sharing parameters, and the visualization of classification results.



(a) α_{max} (b) α_{min}

Figure 2. Sensitivity of hyper-Parameters α_{max} and α_{min}.



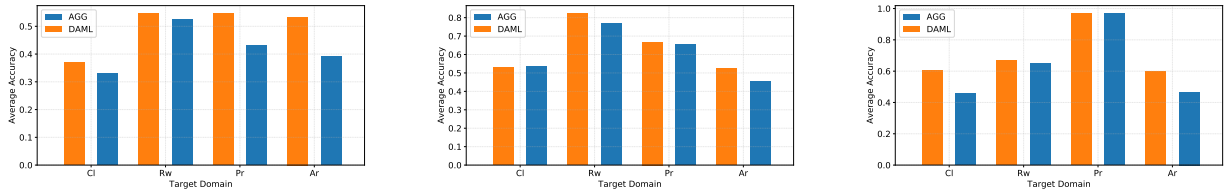
(a) β (b) η

Figure 3. Sensitivity of hyper-Parameters β and η.

3.1. Parameter Sensitivity

We test the sensitivity of parameter α_{max}, α_{min}, β and η. We want to demonstrate two claims: (1) The performance is stable near the optimal value of the hyper-parameters; (2) The performance will drop much when the hyper-parameters deviate from the optimal value much. The first claim demonstrates that the hyper-parameters are not sensitive and easy to tune while the second claim indicates that the hyper-parameters are still necessary even though they are not sensitive.

For α_{min}, We fix α_{max} to be optimal, *i.e.* $\alpha_{\text{max}} = 0.6$ and change α_{min}. For α_{max}, We fix α_{min} to be optimal, *i.e.* $\alpha_{\text{min}} = 0.2$ and change α_{max}. We evaluate the performance



(a) Classes in 1 source domain (b) Classes in 2 source domains (c) Classes in 3 source domains

Figure 4. The average accuracy of target data from classes existing in 1 source domain, 2 source domains and 3 source domains.

with different hyper-parameters on the DAML on ArClRw-Pr task. As shown in Figure 2 and 3, the performance is fairly stable around the optimal value for $\alpha_{\max}$, $\alpha_{\min}$ and η. For β, the learning rate to finally update the parameters, the performance is stable within range $[0.0003, 0.003]$, which is a widely adopted range for learning rate. On the other hand, when deviating from the optimal value a lot, the performance drops much.

3.2. Classes with Different Domain Variations

We have discussed in the main text that the disparate label sets between source domains cause different classes to have different domain variations. And the different domain variations lead to different performance and generalization abilities for different classes. We also argue that the previous domain generalization works fail to consider the minor class existing in few domains and thus does not perform well on such class. We empirically demonstrate the above claims in this section.

We evaluate the accuracy of target data in four tasks of the open-domain Office-Home dataset, where each task transfers from three domains to the remaining domain. We divide the non-open target classes into three parts by how many domains each class exists in, where we have classes existing in one, two and three domains. As shown in Figure 4, we can observe that DAML outperforms the performance of AGG in nearly all classes, especially on the classes that exist in only one domain, which demonstrates that DAML can address the different domain variations for different classes. Also, we can observe that the accuracy of classes existing in one domain is much lower than classes in two and three domains, which demonstrates our claim on the inferior performance of minor classes.

3.3. Trade-off between Accuracy and Efficiency

In the ODG problem, a large domain gap exists between the source and target domains. Using a shared network for all domains is detrimental to the discriminative power on all domains. We prioritize the performance in our network design, so we use separate networks for different domains. Although using separate networks for different domains makes the training and inference time increase linearly with the number of domains, the DAML framework also allows networks of different domains to share parameters. We explore the architecture where the three domains each have a specific classifier but share the whole backbone, denoted as DAML-S. We compare DAML, DAML-S and the baseline of domain aggregation in a shared network (AGG) on the open-domain Office-Home dataset. As shown in Table 4, the accuracy drops a little when sharing all the backbone parameters across domains, but DAML-S still outperforms the baseline with a large margin. Note that with the shared backbone, DAML-S has only a bit more per-batch training time and nearly the same per-image inference time compared with only one network. Thus, we can consider sharing parts of the network parameters across domains as a trade-off between accuracy and efficiency.

Table 4. Results on the Office-Home dataset with shared backbone.

Method	Cl	Rw	Pr	Ar	Avg
AGG	42.83	62.40	54.27	42.22	50.43
DAML	45.13	65.99	61.54	53.13	56.45
DAML-S	44.21	64.73	59.47	50.81	54.81

3.4. Visualization

We visualize the classification results of DAML and AGG on the ClPrRw-Ar task in the Office-Home dataset in Figure 5. We visualize the source images and target images classified wrongly by both, classified correctly by both DAML and AGG, only classified correctly by AGG, and only classified correctly by DAML. We can observe that the image classified wrongly by both and only classified correctly by AGG are quite different from all the source domains, like multiple clocks and confusing background. We manually check the images only classified correctly by AGG and find that most of them are accidentally classified correctly in one run while classified wrongly in a different random seed. For the images only classified correctly by DAML, we can see a digital clock among all the mechanical clocks. The digital clock also exists in the source domains but AGG fails to learn the knowledge of them, which demonstrates that DAML can learn a more generalizable representation.

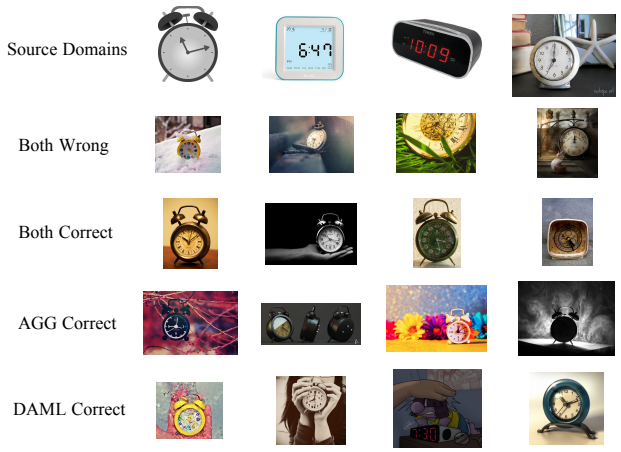


Figure 5. Visualization of classification results.

References

[1] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 215–223, 2011. 2

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, volume 70, pages 1126–1135, 2017. 2

[3] Mehadi Hassen and Philip K Chan. Learning a neural-network-based representation for open set recognition. In *SIAM International Conference on Data Mining (SDM)*, pages 154–162. SIAM, 2020. 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2

[5] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 5543–5551. IEEE, 2017. 1, 2

[6] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019. 2

[7] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1406–1415, 2019. 2

[8] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge, 2017. 2

[9] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision (ECCV)*, 2010. 2

[10] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1