Generalization on Unseen Domains via Inference-time Label-Preserving Target Projections –Supplementary–

Prashant Pandey[1], Mrigank Raman*[1], Sumanth Varambally*[1], Prathosh AP[1]
[1]IIT Delhi
{bsz178495, mt1170736, mt6170855, prathoshap}@iitd.ac.in

1. Theory

Proposition 1. *The label-preserving transformation f defined in Eq. 1 of the main paper reduces the $\mathcal{H}$-divergence between any pair of domains on which it is learned. Stated explicitly:*

Assume the case of Binary Classification. We assume that class prior probabilities for each class is equal in all the source domains, i.e.

$$\mathbf{P}(\mathcal{Y}_i = \ell) = \mathbf{P}(\mathcal{Y}_k = \ell) \quad \forall i, k \in \{1, ..., |S|\}, \forall \ell \in \{0, 1\}. \tag{1}$$

where $\mathbf{P}(\mathcal{Y}_i = \ell) = \mathbb{E}_{x \in \mathcal{D}_i^S}[\mathbb{1}_{g(x)=\ell}]$ denotes the proportion of examples having label ℓ in source domain $\mathcal{D}_i^S$. If there exists a metric space denoted by $(\mathcal{F}, d)$ and a transformation function $f : \mathcal{X} \to \mathcal{F}$ such that $d(f(x_1), f(x_2)) = 0 \iff g(x_1) = g(x_2)$, i.e. x_1 and x_2 have the same labels (irrespective of domain), then,

$$\Delta_{\mathcal{H}'}[f(\mathcal{D}_i^S), f(\mathcal{D}_k^S)] = 0 \quad \forall i, k \in \{1, ..., |S|\} \tag{2}$$

where $\mathcal{H}'$ denotes the space of hypotheses $h' : \mathcal{F} \to \mathcal{Y}$, and $f(\mathcal{D})$ denotes the distribution $\mathcal{D}$ under the transformation f.

Proof. For any $i, k \in [|S|]$ and any $h' \in \mathcal{H}'$,

$$\left|\mathbf{P}_{x \in \mathcal{D}_i^S}[h'(f(x)) = 1] - \mathbf{P}_{x \in \mathcal{D}_k^S}[h'(f(x)) = 1]\right| \tag{3}$$

$$= \left| \mathop{\mathbb{E}}_{x \in \mathcal{D}_i^S}[\mathbb{1}_{h'(f(x))=1}] - \mathop{\mathbb{E}}_{x \in \mathcal{D}_k^S}[\mathbb{1}_{h'(f(x))=1}] \right| \tag{4}$$

Now, note that $g(x_1) = g(x_2) \implies d(f(x_1), f(x_2)) = 0 \implies f(x_1) = f(x_2) \implies h'(f(x_1)) = h'(f(x_2))$. This means that all the examples of a given class are assigned the same label ℓ by h', regardless of the source domain. Depending on the labels assigned to the different classes by h', these three cases would arise:

Case 1: $h'(f(x)) = 1 \quad \forall x \in \mathcal{X}$

$$\implies \mathop{\mathbb{E}}_{x \in \mathcal{D}_i^S}[\mathbb{1}_{h'(f(x))=1}] = \int_{\mathcal{X}} \mathcal{D}_i^S(x)dx = 1$$

$$= \int_{\mathcal{X}} \mathcal{D}_k^S(x)dx = \mathop{\mathbb{E}}_{x \in \mathcal{D}_k^S}[\mathbb{1}_{h'(f(x))=1}] \tag{5}$$

$$\therefore \left|\mathbf{P}_{x \in \mathcal{D}_i^S}[h'(f(x)) = 1] - \mathbf{P}_{x \in \mathcal{D}_k^S}[h'(f(x)) = 1]\right| = 0$$

Case 2: $h'(f(x)) = 0 \quad \forall x \in \mathcal{X}$

$$\mathop{\mathbb{E}}_{x \in \mathcal{D}_i^S}[\mathbb{1}_{h'(f(x))=1}] = \mathop{\mathbb{E}}_{x \in \mathcal{D}_j^S}[\mathbb{1}_{h'(f(x))=1}] = 0 \tag{6}$$

$$\therefore \left|\mathbf{P}_{x \in \mathcal{D}_i^S}[h'(f(x)) = 1] - \mathbf{P}_{x \in \mathcal{D}_k^S}[h'(f(x)) = 1]\right| = 0$$

Case 3: $h'(f(x)) = 1 \iff g(x) = \ell$ for some $\ell \in \{0, 1\}$

$$\mathop{\mathbb{E}}_{x \in \mathcal{D}_i^S}[\mathbb{1}_{h'(f(x))=1}] = \mathop{\mathbb{E}}_{x \in \mathcal{D}_i^S}[\mathbb{1}_{g(x)=\ell}] = \mathbf{P}(\mathcal{Y}_i = \ell) \tag{7}$$

Similarly,

$$\mathop{\mathbb{E}}_{x \in \mathcal{D}_k^S}[\mathbb{1}_{h'(f(x))=1}] = \mathbf{P}(\mathcal{Y}_k = \ell) \tag{8}$$

$$\therefore \left|\mathbf{P}_{x \in \mathcal{D}_i^S}[h'(f(x)) = 1] - \mathbf{P}_{x \in \mathcal{D}_k^S}[h'(f(x)) = 1]\right| = |\mathbf{P}(\mathcal{Y}_i = \ell) - \mathbf{P}(\mathcal{Y}_k = \ell)| = 0 \tag{9}$$

Since,

$$\left|\mathbf{P}_{x \in \mathcal{D}_i^S}[h'(f(x)) = 1] - \mathbf{P}_{x \in \mathcal{D}_k^S}[h'(f(x)) = 1]\right| = 0 \, \forall h' \in \mathcal{H}' \tag{10}$$

*Equal contribution

we have,

$$\Delta_{\mathcal{H}'}[f(\mathcal{D}_i^S), f(\mathcal{D}_k^S)] = \sup_{h' \in \mathcal{H}'} \left| \mathbf{P}_{x \in \mathcal{D}_i^S}[h'(f(x)) = 1] - \mathbf{P}_{x \in \mathcal{D}_k^S}[h'(f(x)) = 1] \right| = 0 \quad \forall i, k \in \{1, ..., |S|\} \quad (11)$$

□

We note here that in practice, the assumption $g(x_1) = g(x_2) \implies d(f(x_1), f(x_2)) = 0$ need not necessarily hold, i.e. two different images having the same label might not have coincident representations. However, the optimization procedure detailed in Eq. 1 of the main paper forces representations of the same class together into the same cluster, as described in Section 4.1, thus resulting in a higher similarity score for images with the same label.

Proposition 2. *The expected misclassification rate obtained with a classifier h' when the projected target $\mathbf{z_t^*}$ is used instead of the true target $\mathbf{z_t}$, obeys the following upper-bound:*

$$\mathbb{E}_{(\mathcal{D}^T, \mathcal{D}^{T^*})} |\tilde{g}(\mathbf{z_t}) - h'(\mathbf{z_t^*})| \leq \underbrace{\mathbb{E}_{\mathcal{D}^{T^*}} |\tilde{g}(\mathbf{z_t^*}) - h'(\mathbf{z_t^*})|}_{i} + \underbrace{\mathbb{E}_{(\mathcal{D}^T, \mathcal{D}^{T^*})} |\tilde{g}(\mathbf{z_t}) - \tilde{g}(\mathbf{z_t^*})|}_{ii} \quad (12)$$

where $\mathcal{D}^T$ and $\mathcal{D}^{T^*}$ respectively denote the true and the projected target distributions, respectively.

Proof. Using the triangle inequality on $\mathbb{R}$, we get

$$|\tilde{g}(\mathbf{z_t}) - h'(\mathbf{z_t^*})| \leq |\tilde{g}(\mathbf{z_t^*}) - h'(\mathbf{z_t^*})| + |\tilde{g}(\mathbf{z_t}) - \tilde{g}(\mathbf{z_t^*})| \quad (13)$$

$\mathbf{z_t}$ and $\mathbf{z_t^*}$ are random variables denoting feature vectors corresponding to the target point and the point obtained by projecting the target point to the source manifold. We assume that these come from the true ($\mathcal{D}^T$) and projected ($\mathcal{D}^{T^*}$) target distributions, respectively. To obtain the average misclassification rate, we take an expectation with respect to their joint distribution.

Since expectation is a linear operation, we can take expectation over both sides of the inequality. Doing so with respect to the joint distribution of $(\mathcal{D}^T, \mathcal{D}^{T^*})$ on both sides,

$$\mathbb{E}_{(\mathcal{D}^T, \mathcal{D}^{T^*})} |\tilde{g}(\mathbf{z_t}) - h'(\mathbf{z_t^*})| \leq \mathbb{E}_{\mathcal{D}^{T^*}} |\tilde{g}(\mathbf{z_t^*}) - h'(\mathbf{z_t^*})| + \mathbb{E}_{(\mathcal{D}^T, \mathcal{D}^{T^*})} |\tilde{g}(\mathbf{z_t}) - \tilde{g}(\mathbf{z_t^*})| \quad (14)$$

Note that in term (i) of the inequality, we can omit the expectation over $\mathcal{D}^T$ since there is no dependence on the random variable $\mathbf{z_t}$.

□

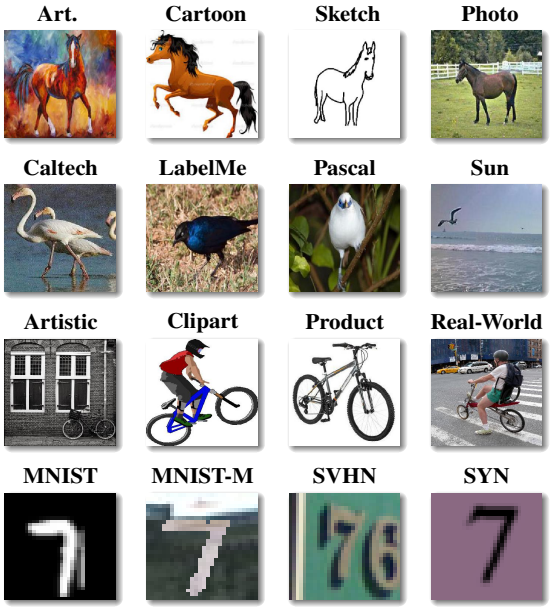


Figure 1: Few example images from PACS (1st row), VLCS (2nd row), Office-Home (3rd row) and Digits-DG (4th row) datasets.

Figure 2: Few example images from CIFAR-10-C dataset depicting 5 levels of severities and four of the 19 Corruption types.

2. Datasets

We demonstrate the effectiveness of our algorithm on the following DG datasets:

PACS: Stands for **P**hoto, **A**rt Painting, **C**artoon and

Sketch. This dataset contains a total of 9991 images taken from different sources such as Caltech256, Sketchy, TU-Berlin and Google Images. Each image is assigned one out of seven possible labels namely dog, elephant, giraffe, guitar, horse, house or person. The substantial domain shift in the dataset poses a significant challenge to DG methods.

VLCS: Stands for **V**OC2007(Pascal), **L**abelMe, **C**altech and **S**un where each one of them is a different dataset differing in the camera specifications. There are a total of 10729 photos in the whole dataset where each photo is assigned one out of five labels namely bird, car, chair, dog, or person.

Office-Home: This dataset is comprised of four domains namely Art, Clipart, Product and Real-World. There are a total of 15588 images in the dataset and each image is assigned one out of 65 classes. The Real-World images are taken with a regular camera and the Product images are taken from a vendor websites and thus differ in background and quality.

Digits-DG: This digit recognition (0-9) dataset contains a mixture of 4 different domains namely MNIST, MNIST-M, SVHN, and SYN that differ drastically in font style and background. The dataset contains 24000 images.

CIFAR-10-C: This dataset is a robustness benchmark for deep learning systems. It consists of test images of CIFAR-10 with 19 different types of corruption having 5 different severity levels (1-5) with 5 indicating most severe. The models are trained on CIFAR-10 and evaluated on CIFAR-10-C. Please refer to Figure 2 for a few samples from different severity levels.

3. Implementation Details

3.1. System Configuration and Frameworks

All the experiments are performed using an Intel Xeon processor (12 Cores) with a base frequency of 2.7 GHz, 32GB RAM, Ubuntu OS and four NVIDIA® Tesla® V100 (16 GB Memory) GPUs. Our implementation was written in Python 3.6.10 and uses PyTorch version 1.6.0 running on CUDA version 10.1.

3.2. Architectural details

Please refer to Figure 7 for different architectures and backbones that we employed to evaluate the performance of our method on each dataset.

We have compared the number of parameters in other state-of-the-art (SoTA) models with ResNet-18 as the backbone: RSC - 11.18M, Jigen - 11.19M, EisNet - 23.5M, MMLD - 12.75M, DDAIG - 12.18M. For the proposed method, the number of parameters are 11.32M. This shows that it is comparable to the SoTA, since the classifier and the generator networks used are shallow.

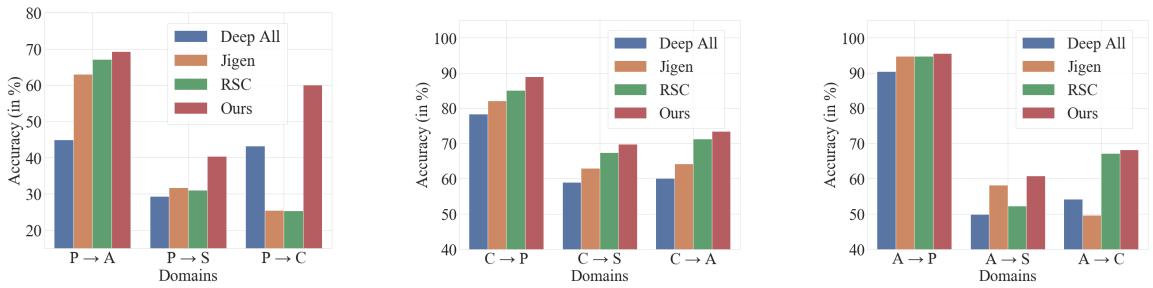
3.3. Hyperparameter choices

We select hyperparameters based on model performance on a validation set consisting of data from the source domains. We use the splits given with the dataset whenever they are available. When the validation split for the source domains are not explicitly provided, we split our training data into a small validation set for model selection. We choose the best model based on (a) the average loss $\mathcal{L}_{\mathcal{A}}$ (for f_θ) (b) the average accuracy (for $\mathcal{C}_\psi$) (c) the average reconstruction/discriminator loss (for G_ϕ) on the validation set.

Training the network f_θ: We use the SGD optimizer with a learning rate of 0.001 to train the f_θ network for all the datasets except for Digits-DG and CIFAR-10-C datasets. Digits-DG dataset is trained with a learning rate of 0.05 while for CIFAR-10-C, we train the network with an initial learning rate of 0.1. We use the same backbone as described in [4] for Digits-DG dataset and for CIFAR-10-C, the backbone employed is Wide Residual Network [3] (WRN) with depth and width of 16 and 4, respectively. We train for 200 epochs with the ResNet-18/ResNet-50 backbones, while AlexNet models are trained for 250 epochs. For the Digits-DG dataset, the f_θ network is trained for 100 epochs while WRN is trained for 200 epochs with a weight decay of 0.0005. In WRN, the initial learning rate of 0.1 is reduced to 0.02, 0.004 and 0.0008 at the 50th, 100th, and 150th epoch, respectively. As a regularization, we apply random image augmentations, namely horizontal flip (with probability 0.5), color jitter (with probability 0.8), greyscaling (with probability 0.2), applying Gaussian Blur (with a kernel of size 21) and adding Gaussian noise (with $\sigma = 0.2$). Batch size for all the backbones is fixed to be 128.

Training the VAE/GAN G_ϕ: VAE is trained with a learning rate of 0.005 and momentum of 0.9 using SGD optimizer for 350 epochs in all cases except for the Digits-DG dataset, where it is trained for 150 epochs. We train with a batch size of 64 in all settings. We use the standard VAE objective, with a combination of both the L1 and L2 losses for the reconstruction error term. In case of GAN (on VLCS dataset), the learning rate is set to be 0.0002 and it is trained for 450 epochs.

Training the Classifier $\mathcal{C}_\psi$: The classifier $\mathcal{C}_\psi$ is trained using the Adam optimizer with a learning rate of 0.003 for 15, 20 and 30 epochs on the Digits-DG backbone, ResNet-based models and AlexNet-based models respectively, with a batch size of 64 in all cases.



(a) Photo as source (b) Cartoon as source (c) Art as source

Figure 3: Relative performance of DG methods trained on a single domain of PACS dataset.

Iteration rate (β)	Avg. number of iterations to stop	Total iterations (M)	Acc. (in %)
0.05	465	20000	81.32
0.01	602	20000	81.79
0.005	2672	20000	80.92
0.001	15976	20000	81.56

Table 1: Comparison of performance on Sketch as a target domain for different values of iteration rate β.

3.4. Iteration rate β

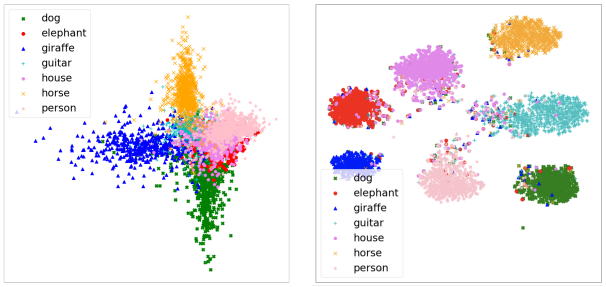
To project a target example $\mathbf{z_t}$ on to the manifold of source features ($\mathcal{Z}_s$) during inference, we follow an iterative optimization procedure where a latent vector $\mathbf{u}$ (initially drawn from an isotropic Gaussian distribution) is optimized in the generative latent space of G_ϕ. We examine the effect of varying the iteration rate β in this optimization procedure through the following experiment: for a fixed iteration rate β, selected from the values $\{0.05, 0.01, 0.005, 0.001\}$, we perform the optimization process until the 'optimal stopping point' (which corresponds to the "elbow" point) is reached. We report the test accuracy in each case. As shown in Table 1, we have validated that for a large number of iterations ($M = 20000$), all reasonably small iteration rates give consistent performance on target domains. The second column in Table 1 reports the average of all optimal stopping points (n*) of the target examples for the Sketch domain of the PACS dataset.

Thus, we conclude that for reasonably small choice of β, *the performance of the inference-time optimization procedure is unaffected by varying* β. Hence, we fix the iteration rate to be 0.01 in all our experiments for uniformity.

4. Additional Results

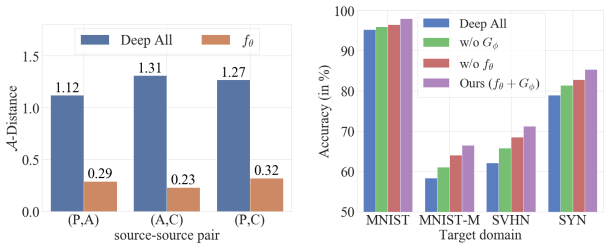
4.1. Class-wise alignment

Despite the large inter-domain discrepancy in the PACS dataset, it is observed that when we train the f_θ network on



(a) w/o f_θ (Deep All) (b) with f_θ

Figure 4: T-SNE plot of features obtained from mixture of PAC (Photo, Art, Cartoon) as source domains using **a)** Deep All and **b)** f_θ network.



(a) source-source $\mathcal{A}$-distance (b) ablation with Digits-DG

Figure 5: **a)** $\mathcal{A}$-distance (lower is better) between sources using Deep All and label-preserving features from f_θ. **b)** Ablation on different components of the proposed method with Digits-DG dataset.

Photo, Art and Cartoon as source domains, we obtain features that align (or cluster) class-wise which is not the case with Deep All features on same source domains as shown by T-SNE plots in Figure 4. This demonstrates the effectiveness of the objective of f_θ network in bringing examples from the same class together in the feature space $\mathcal{F}$ irrespective of the domains they belong to. It helps shallow classifiers (with a single fully connected layer as in $\mathcal{C}_\psi$) to distinguish between features which are more distinct across

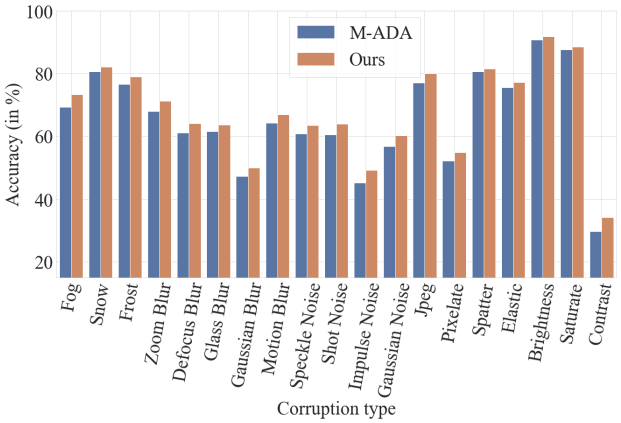


Figure 6: Comparison of performance on all 19 Corruption types with severity level 5.

the class-labels. For features extracted from the Deep All model, the separation across the classes is less which makes the shallow classifier less robust and more error-prone.

We also demonstrate the effectiveness of the f_θ network in reducing source-source $\mathcal{H}$-divergence by comparing the $\mathcal{A}$-distance between features extracted from different source domains through Deep All and the f_θ network. These results are presented in 5a.

4.2. Low resource settings

We train different models on each of the source domains, namely Photo, Cartoon and Art and test on the other three domains of the PACS dataset. We observe that our method is significantly more data efficient compared to the baselines, as shown in Figure 3, owing to the fact that f_θ learns label information from pairs of images and G_ϕ facilitates projection on the manifold of label-reserving source features.

4.3. Ablation with Digits-DG

We conduct ablation with Digits-DG [4] dataset by training the proposed method with and without f_θ and G_ϕ networks. Figure 5b shows the merit of the proposed components in improving the performance on the Digits-DG dataset.

4.4. Robust DG

Figure 6 shows the robustness of our method against all 19 types of corruptions from the CIFAR-10-C dataset [1]. The proposed method trained on CIFAR-10 dataset, achieves generalization to the corrupted target images having severity level 5.

5. Limitations

One potential drawback of the proposed method is the requirement of a relatively larger inference time as optimization is performed for each target example. While this is a potential drawback of the method, we believe it doesn't prevent its practical use, as also noted in [2]. Further, there are only a few hundred iterations needed (very less as compared to training) which takes about 50-100ms (on GPU) to execute. Also note that we do not update any of the model parameters during inference using the target data.

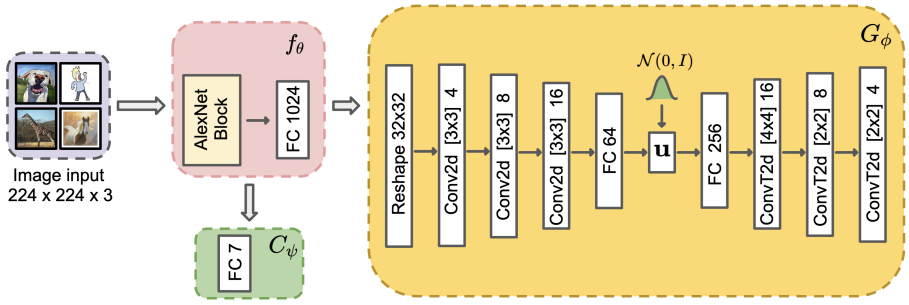
References

[1] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

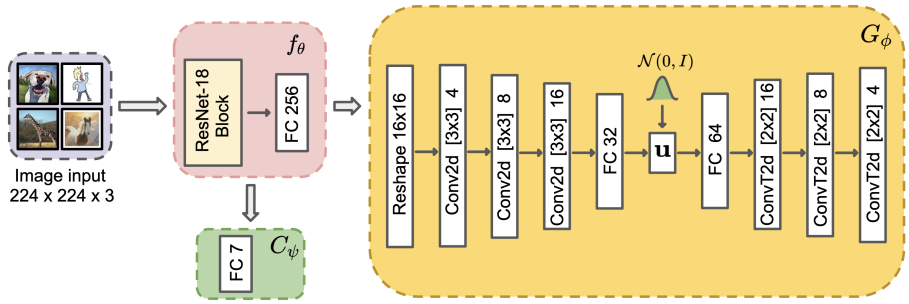
[2] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning*, pages 9229–9248. PMLR, 2020.

[3] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016.

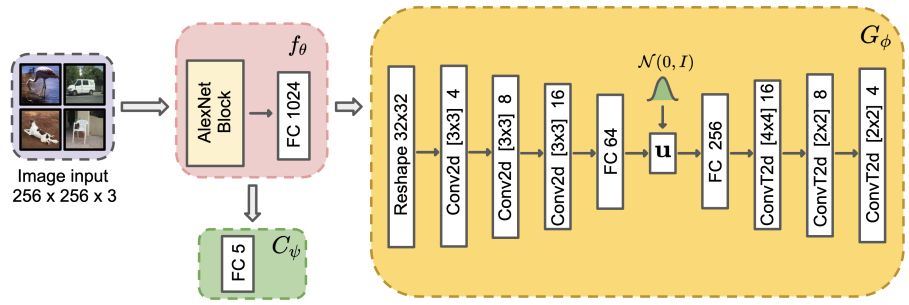
[4] Kaiyang Zhou, Yongxin Yang, Timothy M Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. In *AAAI*, pages 13025–13032, 2020.



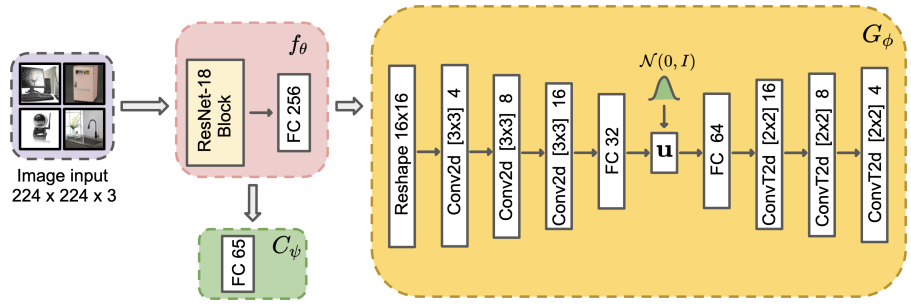
(a) Multi-source DG on PACS with AlexNet



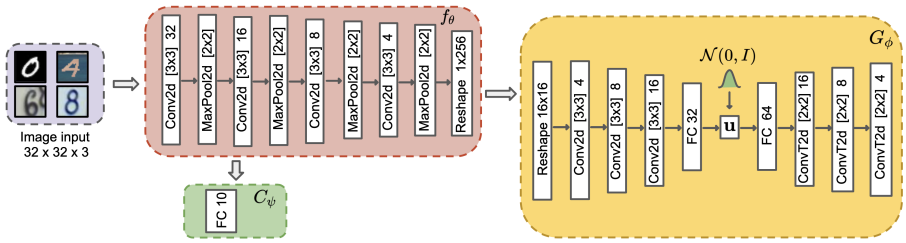
(b) Multi-source DG on PACS with RestNet-18



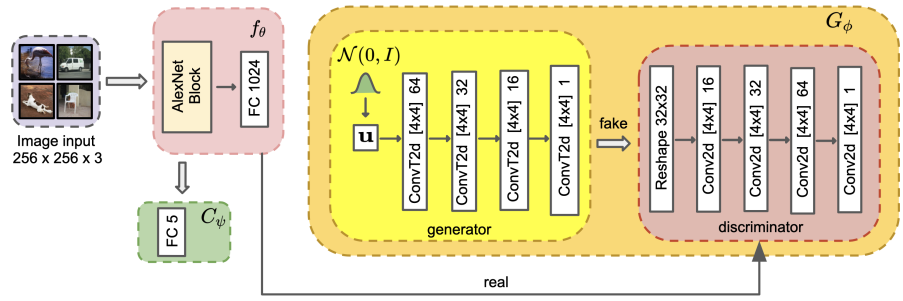
(c) Multi-source DG on VLCS with AlexNet



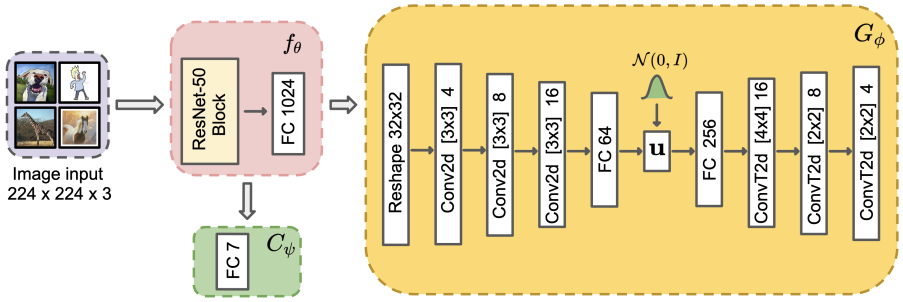
(d) Multi-source DG on Office-Home with ResNet-18



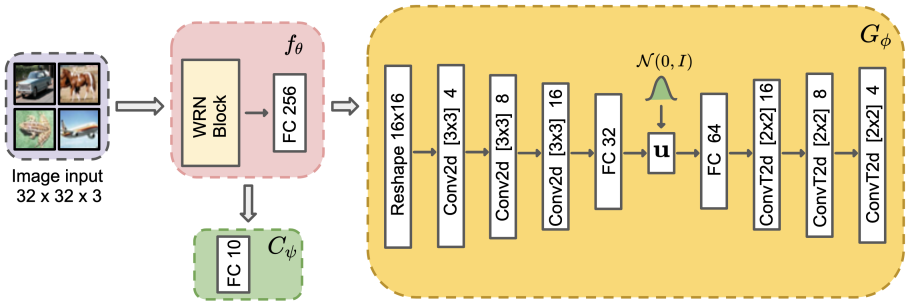
(e) Multi-source DG on Digits-DG



(f) Multi-source DG on VLCS with GAN



(g) Multi-source DG on PACS with ResNet-50



(h) Robust DG on CIFAR-10-C with Wide Residual Network (WRN)

Figure 7: Architectures used for each dataset. Conv2d, ConvT2d, MaxPool2d and FC are Convolution 2D, Convolution Transpose 2D, 2D Max Pooling and Fully Connected layers, respectively. $\mathcal{C}_\psi$ is a single hidden layer classifier. Square bracket represents kernel size with number of output channels written right after it. $\mathbf{u}$ represents a vector from the latent space of G_ϕ. ReLU/LeakyReLU activations are used in all blocks. All components f_θ, G_ϕ and $\mathcal{C}_\psi$ are trained independently.