Improving Domain Generalization with Interpolation Robustness

Anonymous Author(s)
Affiliation
Address
email

Abstract

We address domain generalization by viewing the underlying distributional shift as interpolation between domains and subsequently devise an algorithm to learn a representation that is robustly invariant under such interpolation, which we coin our approach as *interpolation robustness*. Through extensive experiments, we show that our approach outperforms significantly the recent state-of-the-art algorithm Nguyen et al. [2021] and the baseline DeepAll in a limited data setting on PACS and VLCS datasets.

1 Introduction

Domain generalization (DG) aims at learning to generalize well to an unseen test distribution given data from multiple source distributions. In general, there are three main approaches to DG [Shen et al., 2021]. The first approach learns a domain-invariant representation [Arjovsky et al., 2019, Li et al., 2018, Nguyen et al., 2021] from the training domains. This stems from the assumption that there exists several properties that are common across all domains. Learning such a representation captures the essence common to all the domains while simultaneously eliminating any non-generalizable features from individual domains. Next set of approaches offer training strategies that can efficiently amalgamate multiple-domain-specific networks, including meta learning [Li et al., 2017a], and ensemble learning [Wang et al., 2020]. The third approach selectively augments the training data by generating new data from existing training data [Zhou et al., 2021, Yao et al., 2022]. Such techniques, especially selective augmentation has been found to be quite effective to DG [Wiles et al., 2021].

When it comes to critical applications, it is common to encounter scarcity of data along with distribution shift. For example, one of the notorious challenges in medical image analysis is acquiring adequate amount of data required to train deep learning models; it is very expensive to get expert annotations or perform intensive experiments to obtain ground truths [Castro et al., 2020]. Few-shot learning techniques [Song et al., 2022, Fei-Fei et al., 2006, Fink, 2004] have been developed to mitigate the limitation of availability of labelled data. However, this hasn't been addressed in the domain generalization arena enough. Most of the existing DG algorithms assume the availability of good amount of data for training purposes. They use additional generative models to perform density transformations [Nguyen et al., 2021, Robey et al., 2021] between source domains to enforce the invariance condition across them. This approach however quickly becomes unfavourable with less data. We show with our experiments in a limited data setting that a recent SOTA domain-invariant representation method [Nguyen et al., 2021] performs poorly when applied to a limited data setting.

In this paper, we address the DG problem, especially in a limited data setting. Our key perspective to tackle the DG problem is that we view distributional shifts as an interpolation process between domains, thus by learning a representation that is invariant and robust under such interpolation, we can improve the domain-invariant representation and the out-of-distribution generalization. We coin

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

our perspective as *Interpolation Robustness*. Though we propose a particular algorithm in this paper to obtain interpolation robustness, we remark that interpolation robustness is much more general.

To realize interpolation robustness for DG in this paper, we perform interpolation between latent representations of a pair of inputs from two different domains that have the same class label. We then enforce every intermediate representation in the interpolation path to have the same class label. Interpolation operation exposes the model to a novel set of representations that lie between any pair of domains for the same class label. As interpolation operation yields intermediate representations of an input as its domain gets changed, our objective is to find a representation that remains label-invariant to interpolation between any pair of domains in the training data. In summary:

- We show the failure aspect of existing DG methods in the limited data setting;
- We propose *interpolation robustness*, a novel perspective to DG that aims at learning a representation that is robustly invariant to interpolation operations between different source domains. To realize interpolation robustness, we provide a simple yet effective model-agnostic framework that can be used along with any of the existing DG algorithms that use domain-invariant representation;
- We show that our proposed algorithm outperforms Nguyen et al. [2021] and DeepAll a significant amount in a limited data setting on PACS and VLCS datasets.

2 Related Works

Enforcing label invariance on density transformations between a pair of training domains has been a recent area of research in state of the art works [Nguyen et al., 2021, Robey et al., 2021, Zhou et al., 2020] in DG. In general, large generative models (GANs) are used to perform this transformation. Then a classifier is made to classify the images from original domain and the transformed domain to be same. We hypothesize that the precursor step of training a generative model may not be as effective when the data available to train the generative model is limited. In contrast, performing interpolation to transform data from one domain to another exposes the model to unseen data that is present in between the interpolation path but does not require training of additional models.

Prior works [Yao et al., 2022, Gong et al., 2019] synthesize intermediate images between two domains to improve generalization by augmenting the interpolated images to the classifier during training. Instead, we perform interpolation in the latent space and propose a representation that is also robust against the interpolation operation on the images.

3 Interpolation Robustness

Problem setting. We consider the standard setting of domain generalization (DG) where multiple labeled source domains $\mathcal{D} = \{(x_1, y_1, d_1), \ldots, (x_N, y_N, d_N)\} \subset \mathbb{R}^d \times [K] \times [D]$ are available during training, where $x_i \in \mathbb{R}^d$ are inputs, $y_i \in [K] := \{1, \ldots, K\}$ are the label of x_i, $d_i \in [D] := \{1, \ldots, D\}$ is the domain of (x_i, y_i), and K and D denote the number of classes and the number of training domains, respectively. Let P_{dom} be the domain distribution and P_d be the data distribution of domain d. The goal of a DG learner is to learn from $\mathcal{D}$ such that it generalizes well to an unseen data $\{(\bar{x}_i, \bar{y}_i)\}_{i \in [M]}$ in a novel domain that is different from any of the training domains. Let N_d and n_y be the total number of samples in each domain $d \in [D]$ and in each class $y \in [K]$ (over all training domains), respectively.

Our key idea is to learn a representation that is invariant under the interpolation operation under any pair of source domains. We assert that the mechanism behind the distribution shift between any pairs of domains is the interpolation operation. The images get smoothly transitioned from one domain to another through the interpolation operation. Hence, we argue that a model that is robust against interpolation between the training domains can be generalized to unseen data. Additionally, if the intermediate images in the path of the interpolation is semantically valid, the model will be exposed to unseen data. This can be thought of as implicit data augmentation. In general, natural images lie on a non-convex manifold, making interpolation tricky. Hence, instead of performing the interpolation in the pixel space, we train a neural network to unfurl the image manifold to a flattened latent space, and use that space for smooth interpolation. Having a robust representation to perform classification directly is more efficient than having to augment the interpolated images and train it with the rest of

the model as it doesn't require that the intermediate images be semantically meaningful and of high resolution as the original data.

3.1 Model Design

In this section, we describe our model designs to realize interpolation robustness which consists of the three main components.

Encoder (E)**.** We consider the encoder $E_\phi : \mathcal{X} \to \mathcal{Z}$ (with parameter ϕ) that maps from a high-dimensional input space into a low-dimensional latent space. This module unfolds the input images into the latent space $\mathcal{Z}$ and is trained to provide a one-to-one correspondence between the images and the points in the latent space. This is achieved with the help of the next module, C.

Classifier (C)**.** We consider the classifier $C_\theta : \mathcal{Z} \to \mathcal{Y}$ which is parameterized by θ. The goal of the classifier is that $C_\theta(z)$ predicts the class of all samples $\boldsymbol{x}$ that produce the latent representation $\boldsymbol{z}$ via the encoder E_ϕ. To do so, we minimize the following loss function:

$$\mathcal{L}_{\text{cls}}(\theta, \phi) = \mathbb{E}\left[L(C_\theta(E_\phi(x)), y)\right], \tag{1}$$

where $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the training loss such as cross entropy loss, and the expectation is taken with respect to $x \sim P_d(x|y), y \sim P_d(y), d \sim P_{\text{dom}}$.

Interpolator (I)**.** The interpolator module takes as input any two instances x and x' and outputs a latent representation $Z(x, x'; \phi, w) := E_\phi(x) + w(E_\phi(x') - E_\phi(x))$ that is interpolated from the latent representation $z = E_\phi(x)$ and $z' = E_\phi(x')$ of the two inputs, respectively. Here $w \in [0, 1]$ is the interpolation weight. Given two instances x and x' of the same class y, we force the interpolated latent samples $\{Z(x, x'; \phi, w)\}_{w\in[0,1]}$ to be class-invariant under the interpolation, i.e. $\{Z(x, x'; \phi, w)\}_{w\in[0,1]}$ should yield the same label y as its reference points x and x'. To encode such behavior, we define the following interpolation loss:

$$\mathcal{L}_{\text{int}}(\theta, \phi) = \mathbb{E}\left[L\left(C_\theta(Z(x, x'; \phi, w)), y\right)\right], \tag{2}$$

where the expectation is taken with respect to

$$y \sim P_d(y), (d, d') \overset{i.i.d.}{\sim} P_{\text{dom}}, x \sim P_d(x|y), x' \sim P_{d'}(x'|y), \text{ and } w \sim \text{Unif}([0, 1]).$$

Overall, our final loss function is:

$$\mathcal{L}(\theta, \phi) = \mathcal{L}_{\text{cls}}(\theta, \phi) + \lambda \mathcal{L}_{\text{int}}(\theta, \phi). \tag{3}$$

4 Experiments

We perform our experiments on **PACS** [Li et al., 2017b] and **VLCS** [Ghifary et al., 2015] datasets. PACS dataset consists of a total 9,991 images of 7 different classes from four domains: art painting (A), cartoon (C), photo (P), sketch (S). VLCS contains 10,729 from 4 domains each of which are datasets themselves. They are VOC2007 (V), LabelMe (L), Caltech-101 (C), and SUN09 (S).

4.1 Experimental Settings

Implementation Details We chose the Encoder to be a Resnet-18 [He et al., 2016] network for PACS and Alexnet [Krizhevsky et al., 2012] for VLCS dataset following Nguyen et al. [2021]. Classifier C is a single layer dense network preceded by a ReLU layer activation. We follow the experimental set up of Nguyen et al. [2021] in assigning the hyperparameters of the networks which are dimension of $\mathcal{Z}$: 256, learning algorithm: SGD, learning rate: 0.001, momentum: 0.9, minibatch size: 64, weight decay: 0.001 and λ: 1. We train every method for 100 epochs. We choose one of the baselines to be DeepAll which is one of the commonly used baselines in DG literature [Carlucci et al., 2019, Dou et al., 2019, Nguyen et al., 2021]. The data from all the domains is aggregated and the model is trained without any DG specific algorithms and uses equation 1. Then we consider DIRT, the model proposed in [Nguyen et al., 2021] which enforces invariance on z via domain transformations in the form of an additional regularization term to DeepAll. We introduce one of our models as :

Table 2: Results of leave one domain out experiments averaged on all domains for PACS and VLCS. As the data size is reduced from 100% down to 5%, a significant improvement happens in results with our proposed method: DNT and DRINT.

Datasets:	PACS							VLCS				
	100%	n_j			n_i			100%	n_j			n_i
Model		**20%**	**10%**	**5%**	**70**	**50**	**20**		**20%**	**10%**	**5%**	**20**
DeepAll	81.91	67.48	64.44	58.74	71.72	68.86	64.23	**71.27**	**68.71**	67.80	63.13	57.87
DNT	**82.90**	**69.96**	**66.34**	**61.09**	**74.14**	**71.11**	**65.71**	70.97	68.50	**67.87**	**65.65**	**58.68**
DIRT	82.55	72.76	67.72	59.19	75.72	72.40	64.14	72.71	69.90	68.18	63.53	57.76
DRINT	**83.28**	**74.61**	**70.21**	**64.11**	**76.31**	**73.86**	**67.81**	**73.02**	**70.20**	**68.74**	**66.66**	**60.94**

DNT: DeepAll+INT in which our method INT is augmented to the DeepAll model. It uses both losses (1,2). The next model DRINT: DIRT+INT is where our method is augmented to the DIRT model. We demonstrate that on an average, addition of our regularization to both the models increase their performance. In a limited data setting, the performance difference is quite significant.

Sampling for Limited Data Training data is modified into limited data settings via two ways. In the first approach, given a particular (class, domain) pair, a fixed number n_i ($n_i < N_d, \forall d \in [D], n_i < n_y, \forall y \in [K]$) is sampled from each such pairs. This procedure considers uniform sampling of data across classes/domains. As most real data do not occur that way, we also decided to follow a second approach that doesn't modify $P_d(y)$ during the sampling process. We splits the original training data (100%) into a subset of size $n_j\%$ while keeping $P_d(y)$ intact.

4.2 Experimental Results

We use the leave-one-domain-out evaluation scheme where one of the domains is chosen as the target domain and kept aside for evaluation while the model is trained on the aggregated set of all the other domains. The results of individual domains are provided in Tables 4, 5, 7, 6 in appendix A.3. We report the average classification accuracy obtained over all the domains in Table 2 for VLCS and PACS dataset. Entries under the column Full correspond to the accuracy obtained for the full training set of PACS and VLCS. For PACS we have chosen samples of size n_i : 70, 50 and 20 according to the first sampling approach. In the case of VLCS we have n_i: 20. As the dataset size is reduced across the columns, there is an increase in accuracy from less than 1% to over 3% between DIRT & DRINT for both datasets. A similar trend is seen between DeepAll & DNT as well. We have repeated the experiments for the second sampling approach. We divide the datasets into n_j: 20%, 10%, 5% of its original size while keeping the distribution of its classes and domains unchanged. As the dataset size is reduced across the columns, we can observe an increase in accuracy from less than 1% to over 3% between DIRT & DRINT for both datasets. Initially DeepAll has slightly better results at first, but as the size of the dataset reduces, DNT fares better.

Table 1: Results of parametric interpolation vs linear interpolation on the full PACS data set. The improvement of average accuracy over linear interpolation is 0.1% for DNT and 0.7% for ERINT. Details on parametric interpolation is present in A.1

	Interpolation	A	C	P	S	Average
DNT	Linear	**82.49**	76.74	95.83	**76.56**	82.90
	Parametric	82.38	**78.42**	**96.12**	75.05	**82.99**
DRINT	Linear	82.48	77.28	95.27	78.09	83.28
	Parametric	**82.93**	**77.47**	**95.39**	**80.23**	**84.01**

5 Conclusion

We introduce a framework for learning a representation that is invariant to the interpolation operation. We demonstrate that the learned representation is robust to the domain generalization problem. Additionally, we show that the efficacy of our method increases compared to the baseline methods despite scarcity in the availability of training data.

References

A. Tuan Nguyen, Toan Tran, Yarin Gal, and Atilim Gunes Baydin. Domain invariant representation learning with domain density transformations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 5264–5275. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/2a2717956118b4d223ceca17ce3865e2-Paper.pdf.

Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey, 2021. URL https://arxiv.org/abs/2108.13624.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2019. URL https://arxiv.org/abs/1907.02893.

Ya Li, Mingming Gong, Xinmei Tian, Tongliang Liu, and Dacheng Tao. Domain generalization via conditional invariant representations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to generalize: Meta-learning for domain generalization, 2017a.

Shujun Wang, Lequan Yu, Kang Li, Xin Yang, Chi-Wing Fu, and Pheng-Ann Heng. Dofe: Domain-oriented feature embedding for generalizable fundus image segmentation on unseen datasets. *IEEE Transactions on Medical Imaging*, 39(12):4237–4248, 2020. doi: 10.1109/TMI.2020.3015224.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization, 2021.

Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *Proceeding of the Thirty-ninth International Conference on Machine Learning*, 2022.

Olivia Wiles, Sven Gowal, Florian Stimberg, Sylvestre Alvise-Rebuffi, Ira Ktena, Taylan Cemgil, et al. A fine-grained analysis on distribution shift. *arXiv preprint arXiv:2110.11328*, 2021.

Daniel C Castro, Ian Walker, and Ben Glocker. Causality matters in medical imaging. *Nat Commun*, 11(1):3673, July 2020.

Yisheng Song, Ting Wang, Subrota K Mondal, and Jyoti Prakash Sahoo. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities, 2022. URL https://arxiv.org/abs/2205.06743.

Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(4):594–611, apr 2006. ISSN 0162-8828. doi: 10.1109/TPAMI.2006.79. URL https://doi.org/10.1109/TPAMI.2006.79.

Michael Fink. Object classification from a single example utilizing class relevance metrics. In *NIPS*, 2004.

Alexander Robey, George J Pappas, and Hamed Hassani. Model-based domain generalization. *Advances in Neural Information Processing Systems*, 34:20210–20229, 2021.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13025–13032, 2020.

Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2472–2481, 2019.

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

Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders, 2015. URL https://arxiv.org/abs/1508.07680.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012. doi: 10.1145/3065386.

Fabio Maria Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2224–2233, 2019.

Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *NeurIPS*, 2019.

Yingcong Chen, Xiaogang Xu, Zhuotao Tian, and Jiaya Jia. Homomorphic latent space interpolation for unpaired image-to-image translation. pages 2403–2411, 06 2019. doi: 10.1109/CVPR.2019.00251.

A Appendix

A.1 Non linear Interpolation

Due to the success of linear interpolation, we decided to include a parametric version to the latent space interpolation inspired from [Chen et al., 2019].

$$Z(x, x'; \phi, w, \psi) := E_\phi(x) + w * T_\psi(E_\phi(x') - E_\phi(x)) \tag{4}$$

$T_\psi : \mathcal{Z} \to \mathcal{Z}$ is a 3 layer convolutional neural network parameterized by ψ. It is trained jointly with the Encoder E and the Classifier C to minimize the loss in equation 2. In order to make the the operation in equation 4 a valid interpolation, we ensure that the interpolated sample obtained when $w = 1$ is the same as the target $E_\phi(x')$. So the following constraint is added to equation 3 $\parallel Z(x, x'; \phi, w, \psi) - E_\phi(x') \parallel$ when $w = 1$.

We report the results after replacing the linear interpolation operation with parameteric interpolation in Table 1.

A.2 Visualization of the representation

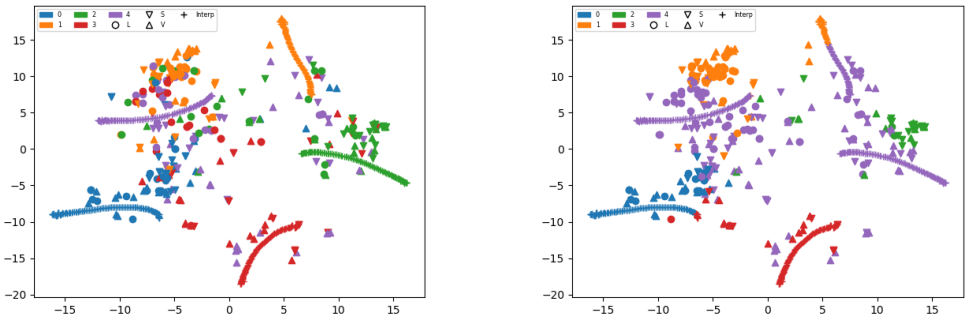
Data from domains S,L,V of the VLCS data is trained on DeepAll, DNT, DIRT and DRINT algorithms where $n_j = 5\%$. Around 300 data points (x) are sampled from the training (includes the validation data) set such that there are 20 from each domain per class. Then, the representation z_i is computed as $z_i = E_\phi(x')$. For each class y in the data set, two samples: (x', y, d') and (x'', y, d'') are drawn from $\mathcal{D}$, $(d', d'') \overset{i.i.d.}{\sim} P_{\text{dom}}$ and $d \neq d'$. First $z' = E_\phi(x')$ and $z'' = E_\phi(x'')$ are computed. Then z_j is computed through linear interpolation between z' and z'' by varying w from 0 to 1 at a span of 0.01. Since we do not want the class of the samples to get modified from interpolation, we assign y as the original label of each of the interpolated samples (for every y in $[K]$). The following plots are obtained after running the TSNE algorithm on z (comprises of both z_i and z_j) by reducing its dimension from 256 to 2. Each of the colors in the plot represents a different class: 0: bird, 1: car, 2: chair, 3: dog, and 4: person. The different domains of the dataset are represented by the following markers: filled circle: L, inverted triangle: S, triangle: V. The interpolated samples are represented by the marker $+$.

Main figures. Figures 1, 2, 3 and 4 contain z's obtained from the algorithms DeepAll, DNT, DIRT and DRINT respectively.

Sub figures on the left column. The colors corresponding to each z point present in the images are based on their true label y. Each of the interpolated z_j's corresponding have been assigned the color based on the class y their initial and final points were sampled from.

Sub figures on the right column. Colors assigned to z_i and z_j are based on the label values y predicted by the model.

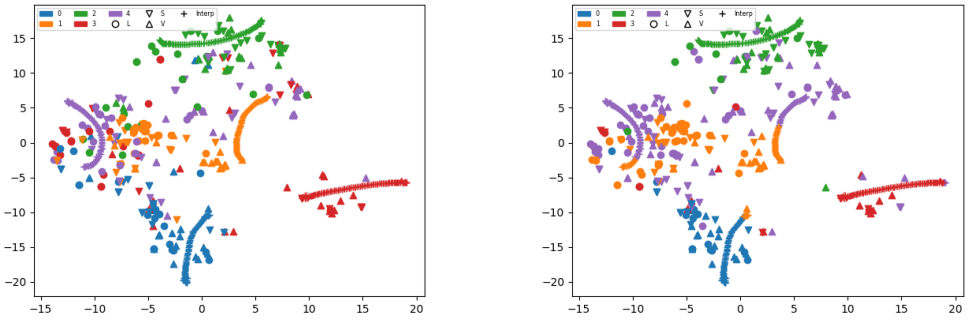
Discussion. It can be seen that the presence of interpolation loss (figures 2 and 4) in DNT and DRINT models has made the prediction of data to be closer to the true labels than without its presence in DeepAll and DIRT (figures 1 and 3). We also compute the mismatch error between the true labels and the predicted labels for each of the models in Table 3. It can be seen that the presence of interpolation loss always reduces the mismatch error.



(a) Colors are from True Labels

(b) Colors are from Predicted Labels

Figure 1: DeepAll



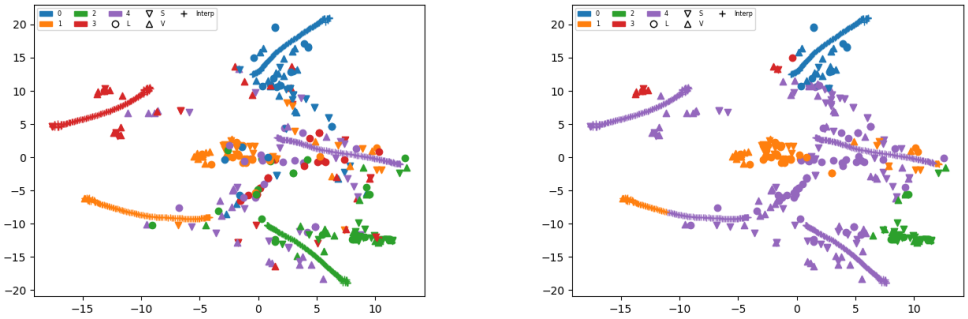
(a) Colors are from True Labels

(b) Colors are from Predicted Labels

Figure 2: DNT

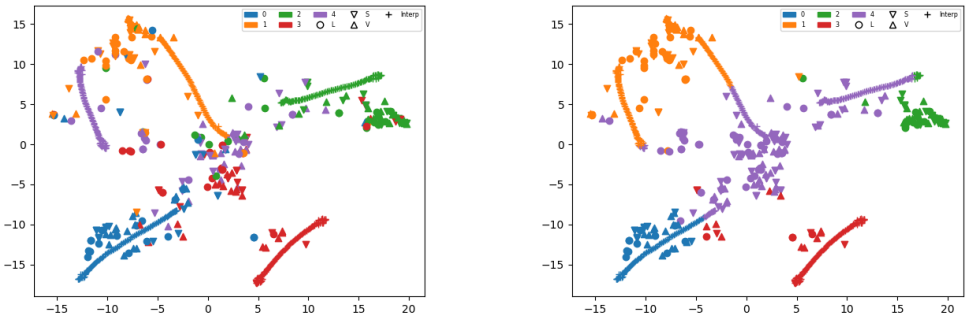
Table 3: Mismatch error % for the visualizations

Model	Mis-match error %
DeepAll	37.51
DNT	22.61
DIRT	49.81
DRINT	44.47



(a) Colors are from True Labels (b) Colors are from Predicted Labels

Figure 3: DIRT



(a) Colors are from True Labels (b) Colors are from Predicted Labels

Figure 4: DRINT

A.3 Further Results

We report the test accuracy of individual domains in both PACS and VLCS dataset in this section. Every reported accuracy is the average over 3 different runs. 90% of the data is used as training data and 10% as the validation data for the results of every dataset. The test accuracy is reported on the model with the best validation accuracy and loss after epoch 50.

Table 4: PACS - Sampling based on Approach 2

Domain	A				C				P				S			
n_j:	**100%**	**20%**	**10%**	**5%**	**100%**	**20%**	**10%**	**5%**	**100%**	**20%**	**10%**	**5%**	**100%**	**20%**	**10%**	**5%**
DeepAll	80.53	66.66	63.29	57.41	75.81	61.57	59.82	53.79	95.23	90.17	88.12	80.73	76.08	51.53	46.55	43.03
DNT	82.49	71.51	66.00	60.74	76.74	65.56	62.47	56.19	95.83	90.87	89.54	83.69	76.56	51.93	47.35	43.77
DIRT	81.39	69.05	66.04	57.61	76.44	67.73	59.35	53.76	94.38	90.69	88.66	79.66	77.99	63.59	56.84	45.73
DRINT	82.48	72.76	68.18	62.59	77.28	70.67	64.02	61.16	95.27	91.07	89.76	82.85	78.09	63.94	58.90	49.86

Table 5: PACS - Sampling based on Approach 1

Domain	A				C				P				S			
n_i:	**Full**	**70**	**50**	**20**	**Full**	**70**	**50**	**20**	**Full**	**70**	**50**	**20**	**Full**	**70**	**50**	**20**
DeepAll	80.53	69.59	67.98	59.69	75.81	60.89	61.27	54.81	95.23	93.03	91.47	88.22	76.08	63.39	54.71	54.22
DNT	82.49	73.37	70.57	60.17	76.74	64.33	65.55	56.72	95.83	94.39	91.43	89.76	76.56	64.50	56.89	56.20
DIRT	81.39	71.97	71.20	58.57	76.44	67.61	65.89	55.60	94.38	92.35	90.99	87.40	77.99	70.95	61.52	54.98
DRINT	82.48	74.86	72.39	63.19	77.28	67.88	68.28	60.84	95.27	93.77	92.53	89.90	78.09	68.76	61.54	57.31

Table 6: VLCS - Sampling based on Approach 2

Domain	C				L				S				V			
n_j:	**100%**	**20%**	**10%**	**5%**	**100%**	**20%**	**10%**	**5%**	**100%**	**20%**	**10%**	**5%**	**100%**	**20%**	**10%**	**5%**
DeepAll	95.09	92.48	87.25	80.42	56.36	53.67	56.15	52.75	66.15	63.07	64.06	61.02	67.49	65.62	63.77	58.32
DNT	94.32	91.96	89.96	88.15	58.38	55.7	56.57	54.27	63.41	60.60	61.61	58.35	67.76	65.65	63.37	61.86
DIRT	95.99	92.15	87.32	77.66	57.06	55.95	57.45	55.10	67.37	65.13	64.83	61.31	70.43	66.38	63.13	60.07
DRINT	95.78	93.96	89.98	88.10	58.06	56.88	57.90	55.56	67.95	62.54	63.22	60.72	70.30	67.41	63.87	62.26

Table 7: VLCS - Sampling based on Approach 1

Domain	C		L		S		V	
n_i	**Full**	**20**	**Full**	**20**	**Full**	**20**	**Full**	**20**
DeepAll	95.09	74.30	56.36	51.10	66.15	52.61	67.49	53.46
DNT	94.32	69.29	58.38	53.70	63.41	53.67	67.76	58.08
DIRT	95.99	65.47	57.06	54.06	67.37	53.44	70.43	58.07
DRINT	95.78	78.83	58.06	54.04	67.95	52.80	70.30	58.08