Improving Domain Generalization with Interpolation Robustness

Ragja Palakkadavath[1] **Thanh Nguyen-Tang**[2] **Sunil Gupta**[1] **Svetha Venkatesh**[1]
[1]Applied Artificial Intelligence Institute (A^2I^2), Deakin University
[2]Department of Computer Science, Whiting School of Engineering, Johns Hopkins University
{s222101652,sunil.gupta,svetha.venkatesh}@deakin.edu.au
nguyent@cs.jhu.edu

Abstract

We address domain generalization (DG) by viewing the underlying distributional shift as performing interpolation between domains. We devise an algorithm to learn a representation that is robustly invariant under such interpolation and term it as *interpolation robustness*. We investigate the failure aspect of DG algorithms when availability of training data is scarce. Through extensive experiments, we show that our approach significantly outperforms the recent state-of-the-art algorithm Nguyen et al. [2021] and the baseline DeepAll on average across different sizes of data 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 is to learn 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. The next 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]. Finally, selectively augmenting the training data by generating new data from existing data [Zhou et al., 2021, Yao et al., 2022].

However, in critical applications, available data is often scarce, with limited labels and having shifts in its distribution. For example, in medical image analysis it is very expensive to get expert annotations or perform intensive experiments to obtain ground truths [Castro et al., 2020]. Furthermore, there are differences in the distribution between data during training versus testing. 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 has not been addressed in the domain generalization arena. Most existing DG algorithms assume the availability of adequate training data. 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 state-of-the-art 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, in a limited data setting. Our key perspective is that we view distributional shifts as an interpolation process between domains, and 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 term this as *Interpolation Robustness*.

INTERPOLATE@NeurIPS, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

We realize this insight by performing 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. The 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 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, creating an 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 algorithm outperforms state-of-art using PACS and VLCS datasets in DG with limited data.

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 $\mathcal{D} = \{(\boldsymbol{x_1}, y_1, d_1), \ldots, (\boldsymbol{x_N}, y_N, d_N)\} \subset \mathbb{R}^n \times \mathcal{Y} \times [D]$ are available during training, where $\boldsymbol{x_i} \in \mathbb{R}^n$ is the input, $y_i \in \mathcal{Y}$ is the label of $\boldsymbol{x_i}$, $d_i \in [D] := \{1, \ldots, D\}$ is the domain of $(\boldsymbol{x_i}, y_i)$, and D denotes the number of training domains. 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 $\{(\bar{\boldsymbol{x}}_{\boldsymbol{i}}, \bar{y}_i)\}_{i \in [M]}$ 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 $y \in \mathcal{Y}$ 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. We argue that a model that is robust against interpolation between the training domains can be generalized to unseen data. Moreover, if the intermediate images in the path of the interpolation are 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 [Chen et al., 2019]. Instead of performing the interpolation in the pixel space, we train a neural network to unfurl the image manifold to a latent space, and use that latent space for smooth interpolation. Having a robust representation to perform classification directly is more efficient than augmenting the interpolated images and re-training the model. It also doesn't require generation of high resolution intermediate images that are on the same level as original data.

3.1 Model Design

In this section, we describe our model design to realize interpolation robustness. It consists of three main components.

Encoder (E)**.** We consider the encoder as $E_\phi : \mathcal{X} \to \mathcal{Z}$ (with parameter ϕ). It maps data from the high-dimensional input space into a low-dimensional latent space. E must create the latent representation space $\mathcal{Z}$ such that the representations z obtained from inputs that share the same label should lie clustered together and z's obtained from inputs having different labels should lie spread apart. This is achieved with the help of the classifier module below.

Classifier (C)**.** We consider the classifier as $C_\theta : \mathcal{Z} \to \mathcal{Y}$ which is parameterized by θ. $C_\theta(\boldsymbol{z})$ predicts the class of all samples $\boldsymbol{x}$ that produce the latent representation $\boldsymbol{z}$ via the encoder E_ϕ. In order for C to associate the representation $\boldsymbol{z}$ to its correct label, we minimize the following loss function:

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

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

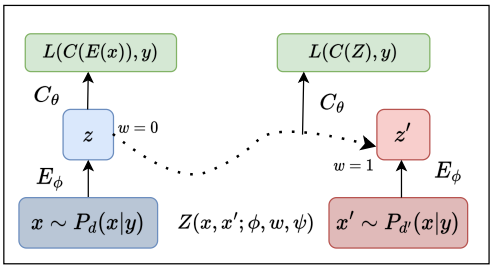


Figure 1: Model Design

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

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

where the expectation is taken with respect to

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

Overall, our final loss function is:

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

where λ is the regularization hyperparameter.

We also include a parametric interpolation within the latent space inspired from [Chen et al., 2019]. This leads to an additional module T_ψ such that: $Z(\boldsymbol{x}, \boldsymbol{x'}; \phi, w, \psi) := E_\phi(\boldsymbol{x}) + wT_\psi(E_\phi(\boldsymbol{x'}) - E_\phi(\boldsymbol{x}))$. Here, $T_\psi : \mathcal{Z} \to \mathcal{Z}$ is parameterized by ψ. It is trained jointly with E and C to minimize the loss in equation 2. In order for the traversal to be a valid interpolation, the interpolated sample obtained when $w = 1$ should be the same as the target $E_\phi(\boldsymbol{x'})$. This constraint is also added to the loss to get the parametric version of equation 2 below.

$$\mathcal{L}_{\text{int}}(\theta, \phi, \psi) = \mathbb{E}\left[L\left(C_\theta(Z(\boldsymbol{x}, \boldsymbol{x'}; \phi, w, \psi)), y\right)\right] + \|Z(\boldsymbol{x}, \boldsymbol{x'}; \phi, 1, \psi) - E_\phi(\boldsymbol{x'})\|_2, \quad (4)$$

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

Sampling for Limited Data Training data is modified into limited data settings via two ways. In the uniform sampling approach, given a particular (class, domain) pair, a fixed number n_i ($n_i < N_d, \forall d \in [D], n_i < n_y, \forall y \in [\mathcal{Y}]$) is sampled from each such pairs. Samples in the new training set will be uniformly distributed among domains and classes. As most real data does not occur that way, we also follow proportional sampling approach that doesn't modify $P_d(y)$ during the sampling process. We split the original training data (100%) into a subset of size n_j while keeping $P_d(y)$ intact. Validation and test data remain the same.

Table 2: Prediction accuracy % for PACS and VLCS. Our methods: DNT and DRINT outperform DeepAll and DIRT respectively on an average.

Datasets:	PACS							VLCS				
	100%	Proportional Sampling (n_j)			Uniform Sampling (n_i)			100%	Proportional Sampling (n_j)			Uniform Sampling (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**

Implementation Details We chose E as Resnet-18 [He et al., 2016] network for PACS and Alexnet [Krizhevsky et al., 2012] for VLCS following Nguyen et al. [2021]. C is a single layer dense network preceded by a ReLU layer activation. T is a 3 layer convolutional neural network. 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 and report the average over 3 different seeds.

Baselines 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] and uses only equation 1 while training on the aggregated dataset of all training domains. 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 our models as : DNT and DRINT. DNT: DeepAll+INT in which loss (INT: 2) is augmented to the DeepAll model. The next model DRINT: DIRT+INT is where INT is augmented to the DIRT model. We demonstrate that on an average, presence of INT in both the models increase their performance.

Table 1: Linear vs parametric interpolation on PACS

	Interpolation	A	C	P	S	Average
DNT	Linear	**82.49±0.4**	76.74±0.5	95.83±0.1	**76.56±1.6**	82.90
	Parametric	82.38±0.4	**78.42±0.5**	**96.12±0.1**	75.05±0.7	**82.99**
DRINT	Linear	82.48±0.6	77.28±0.4	95.27±0.1	78.09±1.4	83.28
	Parametric	**82.93±0.1**	**77.47±0.5**	**95.39±0.1**	**80.23±0.3**	**84.01**

4.2 Experimental Results

We use 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 other domains. Results on the individual domains are provided in Tables 4, 5, 7, 6 in Appendix A.2. We report the average prediction accuracy obtained over all the domains in Table 2 for VLCS and PACS. Entries under the column 100%[1] correspond to the accuracy obtained for the complete training data. We chose n_i : 70, 50 & 20 for uniform sampling approach for PACS. For VLCS we have n_i : 20. As the dataset size is reduced across the columns, there is an increase in accuracy from under 1% to over 3% between DIRT & DRINT and DeepAll & DNT for both datasets. We have repeated the experiments with the proportional sampling approach. We split the datasets into 20%, 10%, 5% (n_j) of its original size while keeping the distribution of its classes and domains unchanged. As we move towards the right over columns under n_j, we can observe an increase in accuracy from under 1% to over 3% between DIRT & DRINT for both datasets. DeepAll has slightly better results at first, but DNT fares better with limited data. Results in Table 2 are from linear interpolation. We provide results on parametric interpolation in Table 1. We note an improvement of roughly 0.1% for DNT and 0.7% for ERINT over linear interpolation on PACS.

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 domain generalization in comparison to DeepAll and DIRTNguyen et al. [2021]. Additionally, we show that the efficacy of our method increases compared to the baseline methods as there is further shortage in the training data.

[1]We couldn't reproduce the reported results of DIRT [Nguyen et al., 2021] for VLCS.

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.

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.

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.

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.

A Appendix

A.1 Visualization of the representation

In this section we visualize the 256 dimensional latent space $\mathcal{Z}$ in a 2D space using TSNE algorithm. 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 ($\boldsymbol{x}$) are sampled from $\mathcal{D}$ such that there are 20 samples per domain per class. Then, the representation z_i is computed as $z_i = E_\phi(x')$. For each class $y \in [\mathcal{Y}]$, two samples: $(\boldsymbol{x'}, y, d')$ and $(\boldsymbol{x''}, y, d'')$ are drawn from $\mathcal{D}$, $(d', d'') \overset{i.i.d.}{\sim} P_{\mathrm{dom}}$ and $d \neq d'$. First $z' = E_\phi(\boldsymbol{x'})$ and $z'' = E_\phi(\boldsymbol{x''})$ are computed. Then z_j is computed via linear interpolation between z' and z'' by varying w from 0 to 1 at a span of 0.01. We assign y as the original label of each of the interpolated samples based on the classes of the reference points $\boldsymbol{x'}$ and $\boldsymbol{x''}$. The following plots are obtained after running the TSNE algorithm on z (comprising 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: $\bullet$: L, $\blacktriangledown$: S, $\blacktriangle$: V. The interpolated samples are represented by the marker $+$.

Main figures. Figures 2, 3, 4 and 5 contain z's obtained from the algorithms DeepAll, DNT, DIRT and DRINT respectively.

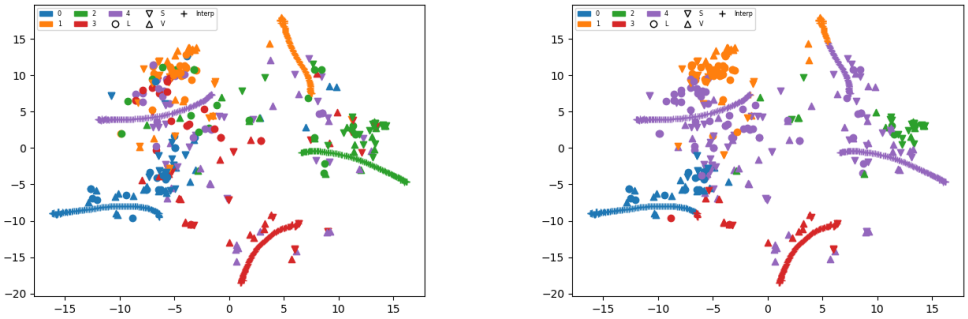
Sub figures on the left column. The colors corresponding to each z point present in the images are based on their true label y. Each of the interpolated z_j's corresponding have been assigned the color based on the class y their reference points were sampled from.

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 3 and 5) 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 2 and 4) respectively. We also compute the mismatch error between the true labels and the predicted labels for each of the models in Table 3. It is % of unequal predictions to that of the original labels. From Table 3, it can be seen that the presence of interpolation loss always reduces the mismatch error.

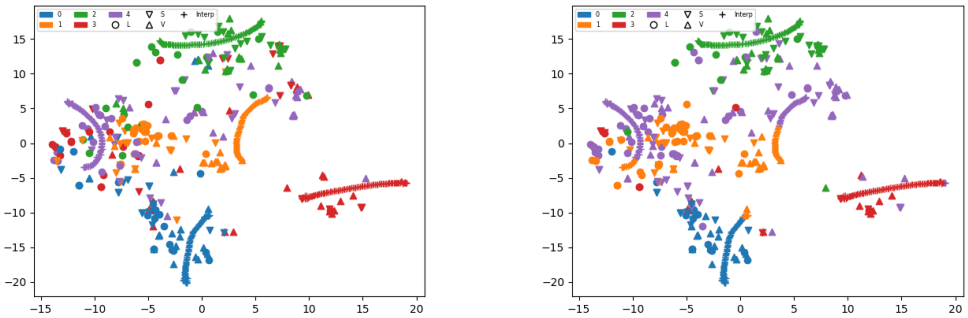
A.2 Further Results

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



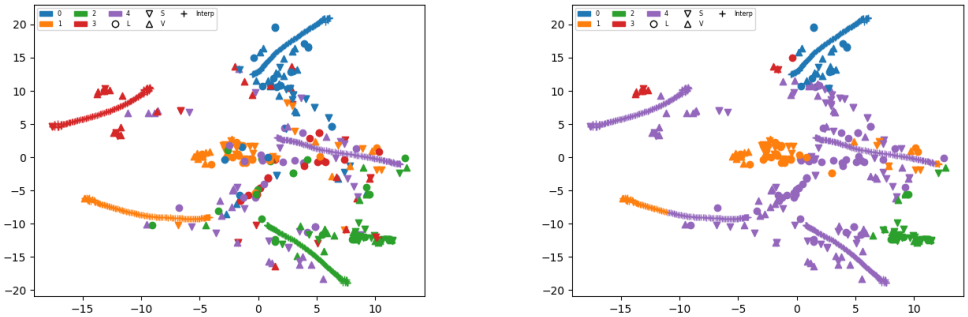
(a) Colors are from True Labels (b) Colors are from Predicted Labels

Figure 2: DeepAll



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

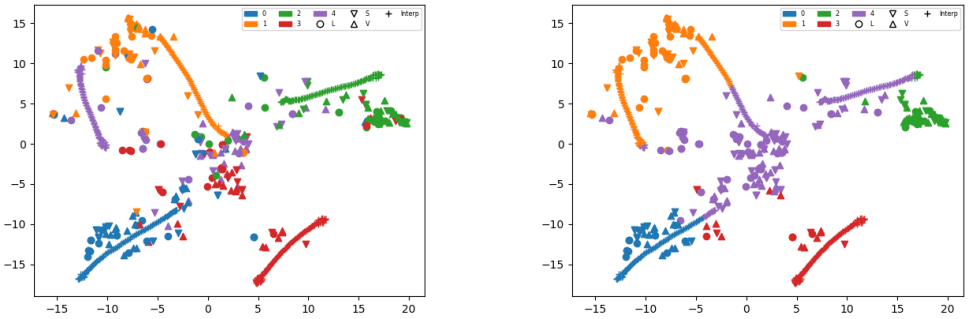
Figure 3: DNT



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

Figure 4: DIRT

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.



(a) Colors are from True Labels

(b) Colors are from Predicted Labels

Figure 5: DRINT

Table 3: Mismatch error % for the visualizations

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

Table 4: PACS - Proportional Sampling

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 - Uniform Sampling

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 - Proportional Sampling

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 - Uniform Sampling

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