Open Domain Generalization with a Single Network by Regularization Exploiting Pre-trained Features

Inseop Chung JIS3613@SNU.AC.KR
Graduate School of Convergence Science and Technology
Seoul National University, Seoul, South Korea
KiYoon Yoo 961230@SNU.AC.KR
Graduate School of Convergence Science and Technology
Seoul National University, Seoul, South Korea
Nojun Kwak NOJUNK@SNU.AC.KR
Graduate School of Convergence Science and Technology
Seoul National University, Seoul, South Korea

Reviewed on OpenReview: *https://openreview.net/forum?id=cammpwFvmQ&noteId=cammpwFvmQ*

Abstract

Open Domain Generalization (ODG) is a challenging task as it not only deals with distribution shifts but also category shifts between the source and target datasets. To handle this task, the model has to learn a generalizable representation that can be applied to unseen domains while also identify unknown classes that were not present during training. Previous work has used multiple source-specific networks, which involve a high computation cost. Therefore, this paper proposes a method that can handle ODG using only a single network. The proposed method utilizes a head that is pre-trained by linear-probing and employs two regularization terms, each targeting the regularization of feature extractor and the classification head, respectively. The two regularization terms fully utilize the pre-trained features and collaborate to modify the head of the model without excessively altering the feature extractor. This ensures a smoother softmax output and prevents the model from being biased towards the source domains. The proposed method shows improved adaptability to unseen domains and increased capability to detect unseen classes as well. Extensive experiments show that our method achieves competitive performance in several benchmarks.

Keywords: data distribution shifts, domain generalization, open-set recognition

1 Introduction

Despite the remarkable achievements of deep neural networks, they still often fail to generalize to out-of-distribution (OOD) data which are not seen during training. It is due to the underlying assumption that the train and test data are independent and identically distributed. This is highly unlikely in real-world scenarios where the target samples at test time may have a discrepant distribution from the train set. Domain generalization (DG) addresses this issue

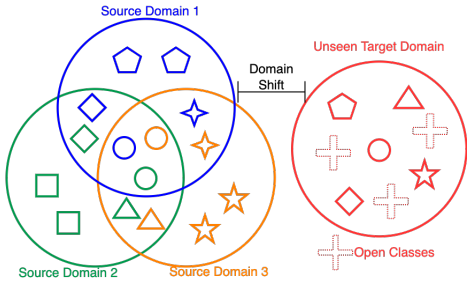

Figure 1: Concept of Open Domain Generalization. Each source domain has a different label set. The target domain contains the open classes which do not exist in the source domains.

by learning a generalizable representation with multiple source domains, but it assumes both source and target data share the same label set. In practice, the target domain may contain classes that do not exist in the source domains. Furthermore, the label sets may differ even among the source domains. Recently, Shu et al. (2021) proposed a very challenging problem called **Open Domain Generalization (ODG)** which assumes that the target domain has not only a different data distribution but also a different label set from the source domains and each source domain holds a different label set as well. Fig.1 shows the concept of ODG in which each source domain has a disparate label set and the target domain holds both the classes in the source domains (known classes) and that are not, which are called *open* classes. ODG aims to achieve two objectives: firstly, to train a model that can generalize to an unseen target domain that exhibits a significant distribution shift from the source domains, and secondly, to recognize the open classes that were not encountered during the training process. The red crosses with dotted lines in Fig.1 indicate the open classes in the target domain. The goal is to recognize the open classes as an unknown class regardless of its actual class. What makes it more challenging is the disparity of label sets among the source domains: some classes exist in multiple domains while the minor classes only exist in a certain source domain (e.g. ⬠, □, ☆ in Fig.1). This challenging setting makes it difficult to apply the existing DG methods (Li et al., 2018a,c; Muandet et al., 2013) which assume the source domains share the same label set. To address this problem, Shu et al. (2021) proposes a Domain-Augmented Meta-Learning (DAML) which employs meta-learning and domain augmentation methods to generalize to the unseen target domain. However, it requires a separate network for each source domain and uses the *ensemble* of all the source domain networks as the final prediction for the target domain input. This is highly undesirable in real-world scenarios where a variety of source domains can be included in the training set. It implies that as the number of source domains increases, the number of required networks and the inference cost increase linearly as well. It can be technically infeasible under a practical environment with limited resources such as power, memory, and computational budget.

Therefore, in this paper, we tackle ODG only with a single network. Since domain generalization typically assumes access to a pre-trained network, our main motivation is to fully exploit this. To safeguard the feature extractor from fitting too much to the source domains, we first train only the classification head instead of fully fine-tuning all the model parameters following previous works (Kumar et al., 2022; Levine et al., 2016; Kanavati and Tsuneki, 2021) such that the readily trained classification head allows the feature extractor not to adapt too much during fine-tuning. To further guide the network during fine-tuning, we propose two regularization methods for the feature extractor and the head respectively. For the feature extractor, we make a prototype of each class in the source domains using the features of the *pre-trained* feature extractor. Then, during fine-tuning, we minimize the distance between the feature of each input and its corresponding prototype. We also regularize the head by maximizing the entropy of its output given the *pre-trained* feature of the input. The first term aims to promote the clustering of features by classes and minimize the distortion of pre-trained features. Meanwhile, the second term encourages the head to not produce over-confident predictions. We observe this results in a smoother class probability distribution (i.e., smoother softmax output) while preserving the pre-trained features. It prevents the model from making overly confident predictions on the unknown classes and prohibiting it from being biased towards the source domains, allowing for better adaptability to the unseen domains.

We verify that our proposed method makes the features deviate less from the pre-trained features and restrains from making over-confident outputs. We empirically show that our proposed method

boosts performance under the ODG setting and it outperforms DAML in several benchmarks using only a single network. For open class recognition, as Vaze et al. (2022) claims that a good closed-set (known classes) classifier is a good open-set classifier, we focus more on improving the accuracy of known classes to achieve better performance on open class recognition. Note that the purpose of our work is not to propose a SOTA method but to demonstrate that fully exploiting the pre-trained features can improve the performance in ODG only with a single network, greatly reducing the inference cost.

2 Methods

2.1 Problem Definition

Open domain generalization assumes multiple source domains are available during training, $\{D^i\}_{i=1}^K$ where K refers to the number of source domains. Each source domain consists of N^i input-label pairs $D^i = \{(x_j^i, y_j^i)\}_{j=1}^{N^i}$ with its own label set, $C^i = \bigcup_j \{y_j^i\}$. As illustrated in Fig. 1, some classes are shared across the source domains while others only belong to one source domain. $C = \bigcup_{i=1}^K C^i$ is the union of all the classes in the source domains. The goal is to train the model only with the source domain data and learn a representation generalizable to the unseen target domain D^t. D^t contains all or subset of C (known classes) and the open classes (unknown classes) which do not exist in C. At evaluation time, the source-trained model is given inputs from D^t and has to correctly classify the input if it belongs to one of the classes in C, otherwise recognize it as the open class. Note that D^t is only used at test time for evaluation and not used at all during training.

2.2 Preliminary: LP-FT

One of our main goals is to train the feature extractor only to an extent such that the pre-trained features remain favorable to the target domain. Having access only to the source domains, we can achieve this by preventing the feature extractor from overfitting to the source domains. Earlier works in transfer learning have used partial fine-tuning or linear probing to mitigate catastrophic forgetting of the pre-trained knowledge (Howard and Ruder, 2018; Xie et al., 2021; Zhai et al., 2022). To realize our goal, we draw inspiration from a recent analysis (Kumar et al., 2022), which shows that higher performance on OOD data[1] can be achieved when linear-probing (LP: updating only the classifier head while freezing the feature extractor layers) is followed by fine-tuning (FT: updating all the model parameters) given a good pre-trained features. We refer to this training scheme as LP-FT following the original work. Following Shu et al. (2021), we employ a classification network consisting of a feature extractor f and a head h. f is initialized with pre-trained weights (we use ImageNet Deng et al. (2009) pre-trained weights.) and h is a C-way linear classifier. We first train *only the head* by linear-probing:

$$\mathcal{L}_{\text{lp}} = \sum_i^K \sum_j^{N^i} CE(h(\bar{f}_0(x_j^i)), y_j^i), \tag{1}$$

where CE and f_0 refer to the cross-entropy loss and the pre-trained f respectively. We denote the frozen f_0 by $\bar{f}_0$. We obtain h^{lp}, a linear classifier head trained on the pre-trained features, by minimizing $\mathcal{L}_{\text{lp}}$. We then initialize the head with h^{lp} and update all model parameters of f and h.

1. OOD data refers to the data with a different distribution from the in-distribution (ID) data (data seen during training). In domain generalization, the target domain can be considered as the OOD data.

$$\mathcal{L}_{\text{lp-ft}} = \sum_{i}^{K} \sum_{j}^{N^i} CE(h^{lp}(f(x_j^i)), y_j^i). \tag{2}$$

In practice, we train the model with mini-batches, but we omit the details for simpler notations. The intuition of LP-FT is that with a randomly initialized head, FT distorts ID features more and changes OOD features less which makes the head overfit to the distorted ID features and leads to poor performance on OOD data. It resolves this problem by initializing with a pre-trained head via linear-probing which preserves the pre-trained features favorable for both ID and OOD data, reducing the amount of change in the feature extractor. Our method adopts the philosophy of LP-FT and further boosts the performance on OOD data, which is the target domain data in our setting.

2.3 Clustering Features by Pre-trained Prototype

With the linear-probed head, we expect that clustering the source domain features according to class would also help the target domain features cluster together class-wisely. We generate a prototype that works as a centroid for each class-wise cluster and minimize the distance between each source input feature and its corresponding class' prototype. Since the goal of LP-FT is to not distort the pre-trained features too much, we choose to generate the prototype of each class by averaging the features from the pre-trained feature extractor f_0:

$$P_c = \frac{1}{\sum_i^K N_c^i} \sum_{i}^{K} \sum_{j}^{N_c^i} (\bar{f}_0(x_j^i)), \tag{3}$$

where c is the class index and P_c is the generated prototype of class c. N_c^i is the number of samples labeled as class c in domain i. Therefore, P_c can be a prototype across multiple source domains if the class c belongs to more than one source domain. We then minimize the distance between each source input feature and its prototype as follows:

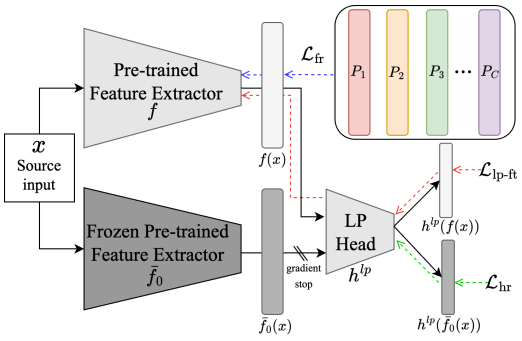

Figure 2: Overall schematic of our proposed training scheme. Each dotted line indicates the path of back-propagation for each loss. Note that $\mathcal{L}_{\text{hr}}$ back-propagates only upto the head and not to f. Also, no loss back-propagates to $\bar{f}_0$, which is only employed during training to provide the pre-trained features of the source domain inputs.

$$\mathcal{L}_{\text{fr}} = \sum_{i}^{K} \sum_{c}^{C} \sum_{j}^{N_c^i} ||f(x_j^i) - P_c||_2^2. \tag{4}$$

Note that each P_c is generated before the training and saved in memory. All P_c's are pre-defined and *fixed* during training, hence the gradient of $\mathcal{L}_{\text{fr}}$ only back-propagates to f, the target feature extractor that we want to train, but not to P_c. Along with $\mathcal{L}_{\text{lp-ft}}$, $\mathcal{L}_{\text{fr}}$ regulates f to cluster the features centered around the prototype of the class the inputs belong to and prevents distorting the pre-trained features excessively. Please be informed that f in $\mathcal{L}_{\text{lp-ft}}$ and $\mathcal{L}_{\text{fr}}$ are initialized as f_0 and its parameters are updated as the training proceeds.

2.4 Maximizing Entropy of Pre-trained Features

The head regularization term maximizes the entropy of the classifier's softmax output conditioned on the pre-trained features of the source domain inputs. We employ a frozen pre-trained feature extractor $\bar{f}_0$ which is a separate feature extractor different from the target f we want to train by

$\mathcal{L}_{\text{lp-ft}}$ and $\mathcal{L}_{\text{fr}}$. With the head initialized as h^{lp}, we produce the logit for each pre-trained source feature as follows: $\sigma(z^s) = \text{softmax}(h^{lp}(\bar{f}_0(x^s))) \in \mathbb{R}^C$ where $z = h^{lp}(\bar{f}_0(x))$ is the logit (the raw output of the head) and σ is the softmax operation. We minimize the following loss term to *maximize* its entropy:

$$\mathcal{L}_{\text{hr}} = \sum_{i}^{K} \sum_{j}^{N^i} \sum_{c}^{C} \sigma(z_j^i)^c \cdot \log \sigma(z_j^i)^c, \tag{5}$$

where $\sigma(z)^c$ denotes the c-th element of the softmax output $\sigma(z)$, or the probability of class c. Note that the loss term *only updates the parameters of the head h, and does not affect the feature extractor, f.*

2.5 Overall Objective

The overall loss function of our proposed method is as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{lp-ft}} + \mathcal{L}_{\text{fr}} + \lambda_{hr}\mathcal{L}_{\text{hr}}. \tag{6}$$

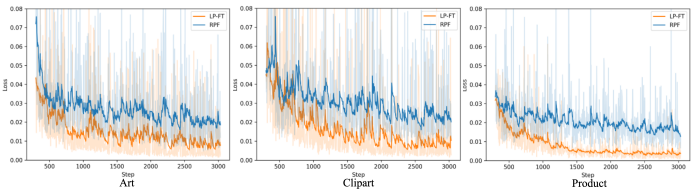

Figure 3: The plot of $\mathcal{L}_{\text{lp-ft}}$ for each source domain in the Office-Home dataset when the target domain is Real-World.

Fig. 2 shows the overall training scheme of our proposed method. The three loss terms are summed and optimized simultaneously. λ_{hr} is the balance weight for $\mathcal{L}_{\text{hr}}$. $\mathcal{L}_{\text{lp-ft}}$ affects both f and h, while $\mathcal{L}_{\text{fr}}$ and $\mathcal{L}_{\text{hr}}$ affect only f and h respectively. $\mathcal{L}_{\text{fr}}$ regulates f to cluster the features by classes and not to deviate too far from the pre-trained features by anchoring each feature to its corresponding prototype, P_c. On the other hand, $\mathcal{L}_{\text{hr}}$ regularizes the head to maximize the entropy of logits conditioned on the pre-trained features from f_0. Since the head is initialized as h^{lp} which is fitted to classify the pre-trained features from f_0 correctly, minimizing $\mathcal{L}_{\text{hr}}$ demands the head to deviate from h^{lp} to predict less confidently (higher entropy) on the pre-trained features. The combination of these two terms results in a larger $\mathcal{L}_{\text{lp-ft}}$ as shown in Fig.3. Larger $\mathcal{L}_{\text{lp-ft}}$ indicates that the softmax output has smoother class probability distribution. It implies that the model is less biased towards the source domains seen during training and less prone to making over-confident predictions on target domain which leads to improved adaptability to the target domain containing unknown classes. However, a high weight on $\mathcal{L}_{\text{hr}}$ can impair the head, thus we use a small λ_{hr}. We empirically find that $\lambda_{hr} = 0.1$ works reasonably well across the benchmarks. Based on our empirical observations, our proposed terms have the effect of smoothing the final output, helps to reduce the domain gap between the source and target domains, encourages greater clustering of target features (thus minimizing intra-class variance), and results in less deviation from the pre-trained features. We provide a more detailed analysis of these findings in experiments. We name our method (training by $\mathcal{L}_{\text{total}}$) RPF which stands for Regularization through Pre-trained Features.

2.6 Inference

At inference time, the model is given target domain inputs that are never seen during training. We only use a single network trained by $\mathcal{L}_{\text{total}}$ as opposed to DAML which uses an **ensemble** of multiple networks. When three source domains are used, DAML requires three separate networks and it takes 10.91 GFLOPs and 8.37 milliseconds for an inference of a single image while our proposed method only takes 3.64 GFLOPs and 3.25 milliseconds on a single NVIDIA RTX 3090. Note that $\bar{f}_0$ used in (5) is ***not*** needed at inference time; it is employed only at training time to perform head

regularization. Following You et al. (2019), we set a threshold on the confidence score and classify the target input as the open class if its maximum confidence score ($\max_{c \in C} \sigma(z^t)^c$) is lower than the threshold. Otherwise, it is classified as the class of the maximum confidence score.

3 Experiments

3.1 Experimental settings

In domain generalization, a dataset consists of multiple domains and shares the same label set (classes) across the domains. The experiment is performed by changing the target domain and using the remaining domains as the source domains. However, in ODG scenario, the label set is different between domains as depicted in Fig.1. Shu et al. (2021) provides the detailed information needed to implement their proposed ODG setting such as the class split between the domains. We exactly follow these experimental settings for a fair comparison. Experiments are conducted under three different benchmarks, **PACS** (Li et al., 2017), **Office-Home** (Venkateswara et al., 2017) and **Multi-Datasets** scenario. **Multi-Datasets** scenario is a practical setting proposed by Shu et al. (2021) where the model is trained on three public datasets, **Office-31** (Saenko et al., 2010), **STL-10** (Coates et al., 2011) and **Visda2017** (Peng et al., 2017), and evaluated on four different domains of **Domain-Net** (Peng et al., 2019) which is another cross-domain generalization benchmark. Experiments are conducted using a ResNet-18 (He et al., 2016) backbone pre-trained on ImageNet (Deng et al., 2009) with a head whose output dimension is defined as the number of classes in the source domains. Detailed information about datasets, class splits and implementation details are explained in the appendix. We report the average of 3 runs for each experiment. Two metrics are used for the evaluation: 1) Acc, which is the accuracy only on the known classes (open classes are not given when calculating Acc) and 2) H-score, which is the harmonic mean of known and open class accuracy (both known and open classes are given) following (Fu et al., 2020; Shu et al., 2021).

3.2 Ablation Study

We ablate each component of our method to show its effectiveness. Table 1 shows our ablation study on the Office-Home dataset. HR and FR denotes $\mathcal{L}_{\text{hr}}$ and $\mathcal{L}_{\text{fr}}$ respectively. The table clearly indicates that the exclusion of our proposed regularization terms - resulting in training solely with $\mathcal{L}_{\text{lp-ft}}$ (I) - leads to a decrease in both Acc and H-score. Using only one regularization term (II and III) leads to only a marginal gain compared to LP-FT (I) in most cases, and in some target domains, the performance deteriorates. On the other hand, when the two terms are used together (VII - RPF), the performance gain is marked. This implies that the two terms create a synergetic effect for boosting the ODG performance. Next, we compare some possible variants of our method to justify our key design choices. First, without the pre-trained head by linear-probing (IV), our method shows poor performance which indicates that the pre-trained head is inevitable for our method to be effective, bolstering our motivation of not distorting the pre-trained features excessively. HR_f (V) maximizes entropy of logits conditioned on the features from the target feature extractor f, not the frozen pre-trained one $\bar{f}_0$ i.e. z_j^i in (5) is $h^{lp}(f(x^s))$ not $h^{lp}(\bar{f}_0(x^s))$. This leads to improvement in some target domains but its performance is evidently lower than RPF (VII). It does not show much improvement because its objective contradicts with $\mathcal{L}_{\text{lp-ft}}$ which minimizes the cross-entropy loss on features from f. A similar trend is also observed in *Ent-Min HR* (VI) which minimizes (as opposed to maximizing) entropy of logits conditioned on the pre-trained features. *Ent-Min HR* (VI) is also not effective because it decreases the adaptability of the model by fitting the head more to the pre-trained features

Model	Method	Real-World		Product		Clipart		Art		Avg	
		Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score
I	w/o HR and FR (LP-FT)	68.44	61.74	60.53	57.04	43.56	42.24	53.07	50.07	56.40 ± 0.42	52.77 ± 0.76
II	w/o HR	68.77	61.65	61.62	58.39	44.24	42.33	54.50	52.03	57.28 ± 0.88	53.60 ± 0.59
III	w/o FR	68.90	61.43	61.48	57.61	44.33	43.44	53.67	48.56	57.09 ± 0.59	52.76 ± 0.33
IV	w/o pre-trained head	65.08	58.28	56.78	55.07	41.08	41.20	52.42	48.28	53.84 ± 0.47	50.71 ± 0.62
V	HR_f	67.93	60.94	61.03	58.11	43.47	42.31	55.34	52.16	56.94 ± 0.50	53.38 ± 0.40
VI	*Ent-Min HR*	67.28	60.07	61.06	57.47	44.02	42.22	54.92	52.05	56.82 ± 0.33	52.95 ± 0.61
VII	RPF	**70.67**	**61.92**	**63.51**	**59.55**	**44.55**	**43.55**	**56.05**	**53.02**	**58.70** ± 0.46	**54.51** ± 0.57

Table 1: Ablation study on the Office-Home dataset in the open-domain setting. HR_f and *Ent-Min HR* are trained with FR.

	Method	Art		Sketch		Photo		Cartoon		Avg	
		Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score
Single	FC Li et al. (2019)	51.12	39.01	51.15	49.28	60.94	45.79	69.32	52.67	58.13 ± 0.20	46.69 ± 0.25
	PAR Wang et al. (2019)	52.97	39.21	53.62	52.00	51.86	36.53	67.77	52.05	56.56 ± 0.51	44.95 ± 0.57
	RSC Huang et al. (2020)	50.47	38.43	50.17	44.59	67.53	49.82	67.51	47.35	58.92 ± 0.46	45.05 ± 0.60
	CuMix Mancini et al. (2020)	53.85	38.67	37.70	28.71	65.67	49.28	**74.16**	47.53	57.85 ± 0.32	41.05 ± 0.66
	LP-FT Kumar et al. (2022)	49.12	44.27	55.25	43.50	69.50	67.75	62.73	57.94	59.15 ± 0.83	53.36 ± 0.90
	RPF (Ours)	50.99	45.87	55.55	49.00	69.33	**70.51**	67.87	59.47	60.94 ± 0.78	**56.21** ± 0.64
Ensemble	DAML Shu et al. (2021)	**54.10**	43.02	58.50	**56.73**	**75.69**	53.29	73.65	54.47	**65.49** ± 0.36	51.88 ± 0.42

Table 4: Results on PACS dataset in the open-domain setting.

of the source domains. These two observations justify our method of maximizing entropy of logits conditioned on the pre-trained features.

Metric	Domain(s)	Method	R	P	C	A	Avg
Domain gap	Trg. vs. Src	RPF	0.16	0.16	0.27	0.15	**0.19**
		LP-FT	0.19	0.26	0.30	0.22	0.24
	Source	RPF	0.30	0.20	0.30	0.19	**0.25**
		LP-FT	0.37	0.33	0.33	0.29	0.33
Intra-class dist.	Target	RPF	0.44	0.30	0.49	0.36	**0.40**
		LP-FT	0.51	0.48	0.55	0.58	0.53
	Source	RPF	0.35	0.26	0.40	0.28	**0.32**
		LP-FT	0.43	0.44	0.44	0.44	0.44
Diff. from f_0	Target	RPF	0.59	0.61	0.66	0.46	**0.58**
		LP-FT	0.64	0.71	0.69	0.54	0.64
	Source	RPF	0.65	0.57	0.72	0.65	**0.65**
		LP-FT	0.73	0.69	0.73	0.75	0.73

Table 2: Analysis on domain gap, intra-class distance and difference from pre-trained features.

Metric	Class	Method	R	P	C	A	Avg
Confidence score	Known	RPF	0.76	0.72	0.57	0.65	**0.67**
		LP-FT	0.81	0.77	0.59	0.70	0.72
	Unknown	RPF	0.56	0.49	0.47	0.50	**0.51**
		LP-FT	0.63	0.58	0.50	0.59	0.57
Entropy	Known	RPF	0.83	1.04	1.56	1.24	**1.17**
		LP-FT	0.65	0.80	1.46	1.01	0.98
	Unknown	RPF	1.53	1.80	1.96	1.78	**1.77**
		LP-FT	1.23	1.42	1.84	1.40	1.45

Table 3: Analysis on maximum confidence score and entropy of logits using the target domain inputs.

3.3 Analysis

We analyze the impact of our proposed method on both feature-level and logit-level.

Quantitative Analysis on Feature Table 2 shows our feature level analysis. All the numbers are mean squared error distances. LP-FT denotes model I in Table1. Source domain inputs are from the respective validation set. 'Domain gap' is the distance between the feature centroids of the same class from different domains (i.e. the centroid for class c of domain i is $\frac{1}{N_c^i}\sum_j^{N_c^i}(f(x_j^i))$). The first row is the mean of the domain gap between every possible pair of the source and the target while the second row is only between the sources. Our RPF shows shorter distances than LP-FT which means that the same class centroids from different domains are nearby, indicating a smaller domain gap. 'Intra-class dist.' is the average of distance between each feature of input and its corresponding centroid. It is measured for each class and then averaged over the classes. This metric signifies that the larger the value, the less the features are clustered by class or highly dispersed. For both the target and the source domains, RPF is smaller, indicating that our method reduces the intra-class variance. 'Diff. from f_0' shows how much each feature of input is changed from its pre-trained feature after training, hence showing $\|f(x) - f_0(x)\|_2^2$. It shows the average of difference over entire inputs. We observe that regardless of the target domain, features of RPF are changed less from their initial pre-

	Method	Real-World		Product		Clipart		Art		Avg	
		Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score
Single	FC Li et al. (2019)	63.79	55.16	54.41	52.02	41.80	41.65	44.13	43.25	51.03 ± 0.24	48.02 ± 0.57
	PAR Wang et al. (2019)	65.98	57.60	55.37	54.13	41.27	41.77	42.40	42.62	51.26 ± 0.27	49.03 ± 0.41
	RSC Huang et al. (2020)	60.85	53.73	54.61	54.66	38.60	38.39	44.19	44.77	49.56 ± 0.44	47.89 ± 0.79
	CuMix Mancini et al. (2020)	64.63	58.02	57.74	55.79	41.54	43.07	42.76	40.72	51.67 ± 0.12	49.40 ± 0.27
	LP-FT Kumar et al. (2022)	68.44	61.74	60.53	57.04	43.56	42.24	53.07	50.07	56.40 ± 0.42	52.77 ± 0.76
	RPF (Ours)	**70.67**	**61.92**	**63.51**	**59.55**	44.55	43.55	**56.05**	**53.02**	**58.70** ± 0.46	**54.51** ± 0.57
Ensemble	DAML Shu et al. (2021)	65.99	60.13	61.54	59.00	45.13	43.12	53.13	51.11	56.45 ± 0.21	53.34 ± 0.45

Table 5: Results on Office-Home dataset in the open-domain setting.

	Method	Clipart		Real		Painting		Sketch		Avg	
		Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score
Single	FC Li et al. (2019)	29.91	35.42	64.77	63.65	44.13	50.07	28.56	34.10	41.84 ± 0.73	45.81 ± 0.69
	PAR Wang et al. (2019)	29.29	39.99	64.09	62.59	42.36	46.37	30.21	39.96	41.49 ± 0.63	47.23 ± 0.55
	RSC Huang et al. (2020)	27.57	34.98	60.36	60.02	37.76	42.21	26.21	30.44	37.98 ± 0.77	41.91 ± 1.28
	CuMix Mancini et al. (2020)	30.03	40.18	64.61	65.07	44.37	48.70	29.72	33.70	42.18 ± 0.45	46.91 ± 0.40
	LP-FT Kumar et al. (2022)	33.70	36.74	**69.84**	66.34	47.24	49.29	29.74	33.46	45.13 ± 0.20	46.46 ± 0.29
	RPF (Ours)	36.28	40.54	69.71	67.08	**50.69**	**53.01**	31.75	35.79	**47.11** ± 0.61	49.11 ± 0.56
Ensemble	DAML Shu et al. (2021)	**37.62**	**44.27**	66.54	**67.80**	47.80	52.93	**34.48**	**41.82**	46.61 ± 0.59	**51.71** ± 0.52

Table 6: Results on Multi-Datasets scenario in the open-domain setting.

trained features compared to LP-FT, yet RPF shows higher performance. We conjecture that f is changed less from f_0 due to our $\mathcal{L}_{\text{fr}}$ which helps to preserve the pre-trained features and not distort them excessively but rather utilize them to generalize better to the target domains. Overall, with our proposed method, the domain gap is narrowed, intra-class variance is reduced and the pre-trained features are more preserved in favor of better generalization to the target domain.

Quantitative Analysis on Logit Table 3 analyzes the maximum confidence score $(\max_{c \in C} \sigma(z^t)^c)$ and the entropy of the logits given the target domain inputs. Since the input is the target domain, open (unknown) classes are included, so we distinguish the two types in the table. The average over the entire target samples is reported. As shown in the table, RPF produces logits with a lower maximum confidence score and higher entropy for both known and unknown classes over the four target domains, meaning that the class probability distribution of $\sigma(z)$ is smoother. It is noteworthy that even though RPF shows higher entropy, it achieves better performance than LP-FT. RPF trains the model not to predict too over-confidently on the unknown classes and prevents it from being biased towards the source samples during training. We conjecture that this is possible because $\mathcal{L}_{\text{fr}}$ regulates f to stay similar to f_0 while $\mathcal{L}_{\text{hr}}$ penalizes the head to output high entropy on the pre-trained features of the source domain inputs.

3.4 Comparison with other methods

In Table 4, 5, and 6, we compare our method with other methods under the ODG setting. We show the performance on four different target domains for each benchmark and report the average over the four target domains as well. The performance gap between LP-FT and RPF presents the effectiveness of our proposed terms. In all the three benchmarks, our method leads to better Acc and H-score. DAML (Shu et al., 2021) is another baseline that initially proposed this ODG setting as explained earlier. Note that DAML shows the performance of an *ensemble* of multiple source domain networks which means that in this case, it is the ensemble of 3 networks, since 3 source domains are used in each benchmark. RPF reaches the highest performance on all three benchmarks among the non-ensemble methods when averaged across the target domains. In Office-Home, it even outperforms DAML and also shows the best performance for the H-score of PACS and Acc of Multi-Datasets. These results are quite remarkable considering that the performance of DAML is from the ensembled network.

4 Conclusion

We investigate the open domain generalization problem and introduce a method to boost the performance only with a single network. Our method employs two regularization terms for the feature extractor and the head, respectively, utilizing the pre-trained features of inputs while adopting a sequential learning scheme of training the head first and then fine-tuning the whole model. We observe that our approach inhibits the distortion of pre-trained features and alleviates producing overly confident predictions by the head. With extensive experiments, we analyze how our proposed method affects the model and validate its competitive performance against previous leading method, DAML, in several benchmarks, while greatly reducing the inference cost with a single network.

5 Acknowledgement

This work was supported by NRF grant (2021R1A2C3006659) and IITP grant (2022-0-00953), both of which were funded by MSIT, Korean Government.

References

Spearman Rank Correlation Coefficient, pages 502–505. Springer New York, New York, NY, 2008. ISBN 978-0-387-32833-1. doi: 10.1007/978-0-387-32833-1_379. URL https://doi.org/10.1007/978-0-387-32833-1_379.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems*, 34:22405–22418, 2021.

Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. Domain generalization by mutual-information regularization with pre-trained models. *arXiv preprint arXiv:2203.10789*, 2022.

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.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. *Advances in Neural Information Processing Systems*, 32, 2019.

Bo Fu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Learning to detect open classes for universal domain adaptation. In *European Conference on Computer Vision (ECCV)*, August 2020.

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pages 2551–2559, 2015.

Muhammad Ghifary, David Balduzzi, W Bastiaan Kleijn, and Mengjie Zhang. Scatter component analysis: A unified framework for domain adaptation and domain generalization. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1414–1430, 2016.

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.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *European Conference on Computer Vision (ECCV)*, 2020.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Fahdi Kanavati and Masayuki Tsuneki. Partial transfusion: on the expressive influence of trainable batch norm parameters for transfer learning. In *Medical Imaging with Deep Learning*, pages 338–353. PMLR, 2021.

Daehee Kim, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9619–9628, 2021.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=UYneFzXSJWh`.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

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.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018a.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018b.

Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 624–639, 2018c.

Yiying Li, Yongxin Yang, Wei Zhou, and Timothy M. Hospedales. Feature-critic networks for heterogeneous domain generalization. In *International Conference on Machine Learning (ICML)*, volume 97, pages 3915–3924, 2019.

Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

Massimiliano Mancini, Zeynep Akata, Elisa Ricci, and Barbara Caputo. Towards recognizing unseen categories in unseen domains. In *European Conference on Computer Vision (ECCV)*, August 2020.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

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

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.

Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Efficient domain generalization via common-specific low-rank decomposition. In *International Conference on Machine Learning*, pages 7728–7738. PMLR, 2020.

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

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8050–8058, 2019.

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1Dx7fbCW.

Yang Shu, Zhangjie Cao, Chenyu Wang, Jianmin Wang, and Mingsheng Long. Open domain generalization with domain-augmented meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9624–9633, 2021.

Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-set recognition: A good closed-set classifier is all you need. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=5hLP5JY9S2d.

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.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 10506–10518, 2019.

Sang Michael Xie, Tengyu Ma, and Percy Liang. Composed fine-tuning: Freezing pre-trained denoising autoencoders for improved generalization. In *International Conference on Machine Learning*, pages 11424–11435. PMLR, 2021.

Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Universal domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2720–2729, 2019.

Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18123–18133, 2022.

Shanshan Zhao, Mingming Gong, Tongliang Liu, Huan Fu, and Dacheng Tao. Domain generalization via entropy regularization. *Advances in Neural Information Processing Systems*, 33: 16096–16107, 2020.

Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Learning placeholders for open-set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2021a.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *European conference on computer vision*, pages 561–578. Springer, 2020.

Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=6xHJ37MVxxp.

Ronghang Zhu and Sheng Li. Crossmatch: Cross-classifier consistency regularization for open-set single domain generalization. In *International Conference on Learning Representations*, 2021.

Appendix A. Related Work

Domain Generalization is a task of training a model to generalize well to the unseen target domains only with given source domain data. The most popular solution is to learn a domain-invariant feature representation across source domains (Ghifary et al., 2015, 2016; Li et al., 2018b,c; Muandet et al., 2013). Data augmentation is another widely employed strategy. Shankar et al. (2018) directly perturbed the inputs by the gradients of loss to augment the inputs, while Zhou et al. (2020) learned a generator model to diversify the source domains. Zhou et al. (2021b) mixed the style of training instances to generate novel domains. Meta-learning approach (Li et al., 2018a; Dou et al., 2019) is also widely used to simulate domain shifts. Saito et al. (2019) proposed a minimax entropy approach for semi-supervised domain adaptation. Zhao et al. (2020) proposed an entropy-regularization approach to learn domain-invariant features. Mancini et al. (2020) introduced mixing up source samples during training to recognize unseen classes in unseen domains. Kim et al. (2021) proposed a regularization method based on self-supervised contrastive learning. Cha et al. (2021) theoretically showed that finding flat minima results in better generalization and proposed to sample weights densely for stochastic weights averaging (Izmailov et al., 2018). Cha et al. (2022) exploited an oracle model and maximized the mutual information between the oracle and the target model. Shu et al. (2021) primarily proposed an ODG method by combining DG and open class recognition. It utilized the meta-learning scheme along with domain augmentation. Recently, Zhu and Li (2021) proposed another ODG problem having only a single source domain, it used an adversarial data augmentation strategy to generate auxiliary samples and adopted a consistency regularization on them. Kumar et al. (2022) did not particularly tackle the domain generalization but it showed that fine-tuning can distort the pre-trained features and underperform on OOD data. It insists that training the head prior to fine-tuning improves the performance on both the ID and the OOD data. Our work tackles the same problem proposed in Shu et al. (2021) and competes with it, but our method uses only a single network unlike Shu et al. (2021) which utilizes multiple source domain networks.

Open-Set Recognition is a task of identifying whether an input belongs to the classes seen during training or not. The open classes are the classes that are not included in the training set but present at the test time. In the open-set setting, the model has to not only accurately predict a class in the closed-set but also distinguish samples belonging to the open classes as "unknown". Liang et al. (2017) claimed that using temperature scaling and adding small perturbations to the input can separate the softmax score between the known and open classes. Zhou et al. (2021a) proposed placeholders for both data and the classifier to transform closed-set training into open-set training. Recently, Vaze et al. (2022) presented that there is a high correlation between the closed-set accuracy and open-set accuray and showed that open-set performance can be enhanced by improving the closed-set accuracy. Based on this analysis, we resolve the open-set recognition by improving the performance on the closed-set accuracy of the target domain.

Appendix B. Datasets and Class splits

In this section, we further explain the details of the experimental setting of open domain generalization. Please note that this experimental setting is originally proposed by Shu et al. (2021) and we exactly follow its details for a fair comparison. We clearly note that the following tables 7, 8 and 9

are borrowed from the supplementary materials of Shu et al. (2021) and all the credits for proposing the ODG setting and providing details of class splits goes to Shu et al. (2021).

PACS (Li et al., 2017) comprises of four distinct image styles, namely photo (**P**), art-painting (**A**), cartoon (**C**), and sketch (**S**). All four domains share a common label set of 7 classes. Four cross-domain tasks are formed by utilizing each domain as the target domain and the remaining three domains as the source domains: CPS-A, PAC-S, ACS-P, and SPA-C. In order to implement open-domain generalization environment, the label space of the dataset is split, resulting in varying label spaces across different domains. Table 7 displays the specific categories contained within each domain. Note that the order of the source domains matters. For instance, CPS-A and CSP-A would be considered different because Source-2 and Source-3 have different label sets. The target domain includes all the classes present in the dataset, with class 6 (person) being the open class.

Office-Home (Venkateswara et al., 2017) dataset is comprised of images from four distinct domains: Real-world (**R**), Product (**P**), Clip art (**C**) and Artistic (**A**) and consists of 65 classes which makes it more challenging than **PACS** (Li et al., 2017). Similar to **PACS**, four different open domain generalization tasks: ACP-R, ACR-P, APR-C and CPR-A are formed where each domain is used as the target domain, and the remaining three domains serve as the source domains respectively. The 65 classes are distributed among the four domains to realize the open-domain setting. The order of source domains matters as in **PACS**, for example, APC-R would be different from ACP-R. Different from **PACS**, this setting assigns more open classes (54-64) in the target domain and some classes in the source domains are not included in the target domain which makes this ODG setting more intricate.

As explained in the main paper, Shu et al. (2021) proposed the **Multi-Datasets** scenario to realize a more realistic open domain generalization setting in which the source domains are obtained from different resources and the model is trained to show high performance on an unseen target domain. Its purpose is to simulate the situation where the source domains are obtained from different resources. Three public datasets, **Office-31** (Saenko et al., 2010), **STL-10** (Coates et al., 2011) and **Visda2017** (Peng et al., 2017) are used as the source domains and the trained model is evaluated on the four domains (Clipart, Real, Painting and Sketch) of **DomainNet** (Peng et al., 2019). **Office-31** contains 31 classes in three domains: Amazon, DSLR and Webcam. In this setting, the Amazon domain is used. It consists of objects commonly encountered in an office environment. **STL-10** consists of 10 common objects and its labeled data is used as one of the source domains. **Visda2017** is a simulation-to-real dataset for domain adaptation with 12 categories. Its train set is utilized as one of the source domains which is synthetic 2D images of 3D models generated from different angles and lighting conditions. **DomainNet** is another dataset for a cross-domain generalization benchmark. It has 345 classes shared across the four domains. Since there exist too many classes in each domain, 23 classes that exist in the union of the three source domain classes are preserved

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

Table 7: Class split of PACS 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	$0, 3-4, 9-10, 15-16,$ $21-23, 32-34, 43-45, 54-64$

Table 8: Class split of Office-Home dataset.

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

Table 9: Class split of Multi-Datasets scenario.

as the known classes, and other 20 classes are sampled as the unknown classes. Similar to **Office-Home**, some classes in the source domains are not included in the target domain. Since there is a large domain and label set discrepancy between the sources and the target, this scenario naturally forms a decent open-domain generalization scenario.

Once again, we notify that all the contributions and credits for proposing the above ODG setting and the detailed class splits should be given to Shu et al. (2021). For the train and validation split of the source domains, Shu et al. (2021) explains that it follows the protocol in Li et al. (2017) for **PACS** and for other datasets, it randomly selects 10% of data in each category of the source domains as their validation sets.

Appendix C. Implementation Details

Experiments are conducted using a ResNet-18 (He et al., 2016) backbone pre-trained on ImageNet (Deng et al., 2009) with a head whose final output dimension is defined as the number of classes in the source domains. We train the model for 30 epochs using the SGD optimizer with an initial learning rate of 0.001 and batch size of 32. The learning rate is decreased after 24 epochs by a factor of 10. We choose the model that shows the highest accuracy on the held-out source validation set and evaluate it on the target domain. We use the Pytorch (Paszke et al., 2019) automatic differentiation framework and a NVIDIA RTX 3090 for conducting experiments. We report the average of 3 runs for each experiment. Two metrics are used for the evaluation: 1) Acc, which is the accuracy only on the known classes (open classes are not given when calculating Acc) and 2) H-score, which is the harmonic mean of known and open class accuracy (both known and open classes are given) following (Fu et al., 2020; Shu et al., 2021).

Metric	Method	R	P	C	A	Avg
Euclidean Dist.	RPF	0.1072	0.1412	0.0848	0.1416	0.1187
	LP-FT	0.1022	0.0852	0.1104	0.1129	0.1027
Imp.1(%)	RPF	6.87	4.17	12.69	8.10	7.96
	LP-FT	4.25	−0.80	11.13	2.55	4.28
	Δ	2.62	4.98	1.56	5.56	3.68
Imp.2(%)	RPF	4.26	−9.98	1.44	−6.93	−2.81
	LP-FT	6.17	1.95	4.59	3.77	4.12
	Δ	−1.91	−11.93	−3.15	−10.70	−6.92

Table 10: Head parameters similarity and performance improvement over the linear-probing model.

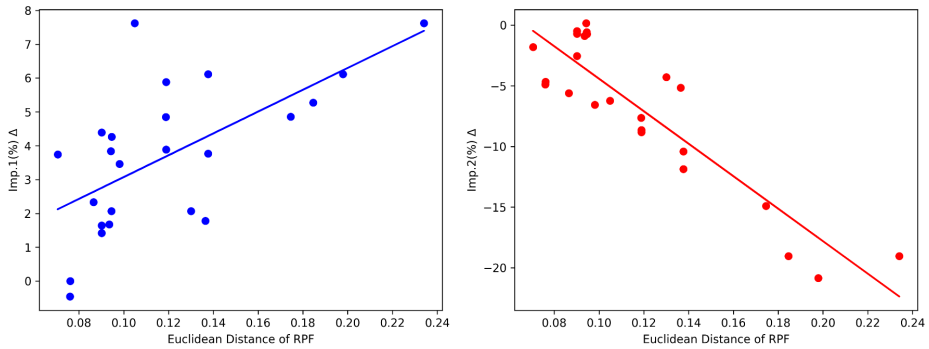

Figure 4: The correlation between the Euclidean distance of RPF and the Δ of Imp.1 (Left) and Imp.2 (Right) respectively.

Appendix D. Head parameters Analysis

In Table 10, we analyze and observe a correlation between the distance of the head parameters from that of the linear-probing model and the performance gain. As explained earlier, our method requires linear-probing the head in advance with a fixed pre-trained feature extractor $\bar{f}_0(.)$. We analyze how our method affects the head in the parameter space by measuring its Euclidean distance to the pre-trained head, h^{lp}. Since the head is a C-way classifier, the shape of its weights and bias are $\mathbb{R}^{C\times d}$ and $\mathbb{R}^C$ respectively, where d is the dimension of the feature, $f(x) \in \mathbb{R}^d$. We concatenate the weights and the bias term, creating $W \in \mathbb{R}^{C\times d+1}$, then measure the Euclidean distance between W of the head trained by each method and that of the h^{lp}. The Euclidean distance is calculated row-wise (class-wise) for W and then averaged over the class dimension. In the table, the head of RPF shows a larger Euclidean distance from h^{lp} than LP-FT which means that RPF has changed the head parameters more from its initial weight h^{lp} than LP-FT.

Imp.1(%) is the performance improvement ratio of Acc compared to the LP model, $((\frac{ModelAcc}{LPAcc} - 1) \times 100)$. RPF consistently achieves higher improvement from the LP model compared to LP-FT. Imp.2(%) shows how much improvement is achieved only by changing the head with the one trained by each method i.e. improvement of $h^*(\bar{f}_0(.))$ compared to $h^{lp}(\bar{f}_0(.))$ where h^* is the trained head by each method. RPF performs worse than LP-FT in Imp.2. This is because RPF changes the head more from its initial weight, h^{lp} via $\mathcal{L}_{\text{hr}}$ which maximizes entropy of logits given the features from $\bar{f}_0(.)$ while LP-FT does not change h^{lp} much but rather adopts it.

	Real-World		Product		Clipart		Art		Avg	
λ_{hr}	Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score	Acc	H-score
1.0	64.66	58.91	57.90	54.87	41.21	41.31	50.45	47.04	53.56 ± 1.26	50.53 ± 0.49
0.5	65.73	59.49	60.81	57.77	42.86	43.41	55.40	51.76	56.20 ± 0.60	53.11 ± 0.49
0.1	**70.67**	**61.92**	**63.51**	**59.55**	**44.55**	**43.55**	**56.05**	**53.02**	**58.70** ± 0.46	**54.51** ± 0.57

Table 11: Ablation study on λ_{hr} using the Office-Home dataset under the open-domain setting. All experiments are trained with $\mathcal{L}_{\text{fr}}$.

Metric	Target domain	Real-World			Product			Clipart			Art		
	Source domains	A	C	P	A	C	R	A	P	R	C	P	R
Domain gap	RPF	0.17	0.18	0.13	0.17	0.18	0.12	0.24	0.30	0.28	0.11	0.18	0.16
	LP-FT	0.19	0.23	0.16	0.26	0.30	0.22	0.26	0.33	0.30	0.20	0.26	0.21
Intra-class dist.	RPF	0.32	0.37	0.36	0.23	0.26	0.28	0.36	0.40	0.43	0.28	0.28	0.30
	LP-FT	0.39	0.46	0.43	0.39	0.46	0.45	0.40	0.44	0.47	0.46	0.42	0.45

Table 12: Analysis on domain gap and intra-class distance for each source-target pair and each source domain.

Δ refers to the absolute difference between RPF and LP-FT in the two Imp.'s. To better capture the correlation between Δ and the Euclidean distance of RPF, we additionally run three more experiments on each domain, hence 24 runs total. The Spearman rank correlation coefficient (ref, 2008) between the Euclidean distance of RPF and Δ of Imp.1 and Imp.2 are 0.6562 and -0.7561 respectively. Fig. 4 shows the scatter plots of the two correlations for all 24 runs. The negative correlation of Imp.2 is expected since the head changes more from h^{lp}, it becomes less fitted to correctly classify features from $\bar{f}_0$. The positive correlation of Imp.1 implies that adjusting the head to deviate from h^{lp} plays a key role in improving the adaptability of the model for unseen domains. We conjecture this is because $\mathcal{L}_{\text{fr}}$ preserves the pre-trained features by clustering them around the P_cs and $\mathcal{L}_{\text{hr}}$ makes the head flexible by maximizing the entropy while minimizing $\mathcal{L}_{\text{lp-ft}}$ as well to ensure the model correctly classifies training samples without compromising its representation. Consequently, the softmax output becomes smoother (Tab.3) and $\mathcal{L}_{\text{lp-ft}}$ increases (Fig.3) which prevents the model from being biased towards the source domains and enhances its adaptability to unseen domains.

Appendix E. Ablation study on λ_{hr}

In the main paper, we mention that a large λ_{hr} leads to the impairment of the head, thus deteriorating the performance. Table. 11 shows our ablation study on λ_{hr}. As λ_{hr} increases, both Acc and H-score decrease in all four target domains. Since $\mathcal{L}_{\text{fr}}$ regulates the f to stay similar to f_0 while $\mathcal{L}_{\text{hr}}$ regulates h to deviate from h^{lp}, if $\mathcal{L}_{\text{hr}}$ is too strong, the head will deviate too far from h^{lp} and contradicts with $\mathcal{L}_{\text{lp-ft}}$ as well which results in a sub-optimal solution that can not correctly classify the features from the f.

Appendix F. More results on domain gap and intra-class distance analysis

In the main paper, we show the average domain gap between every possible pair of the source and the target for each ODG task in the Office-Home dataset. We also report the intra-class distance averaged over the source domains. Therefore, in Table. 12, we show the domain gap for each source-

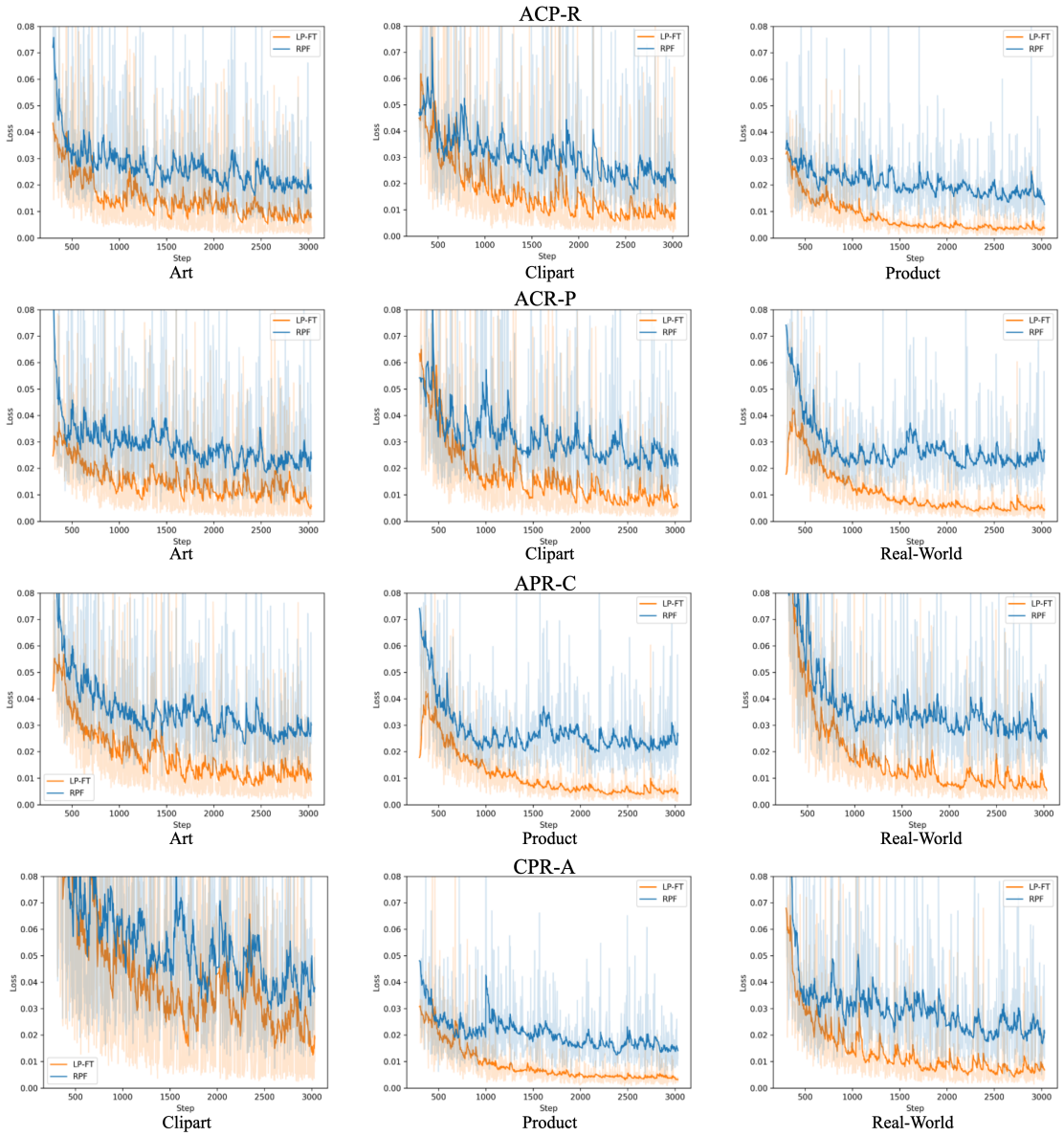

Figure 5: The scale of $\mathcal{L}_{\text{lp-ft}}$ for different ODG task in the Office-Home dataset.

target pair and intra-class distance for each source domain. In the table, we divide the ODG task via vertical lines. 'Domain gap' is the mean squared distance between feature centroids of the target and the source domain while 'Intra-class dist.' refers to the mean distance between each feature point to its centroid feature for each source domain. As we already observed in the main paper, PRF shows smaller domain gap and intra-class distance which indicates that our proposed regularization terms contribute to narrow down the domain gap and cluster the features by classes.

Appendix G. $\mathcal{L}_{\text{lp-ft}}$ comparison between LP-FT and RPF

Fig. 5 shows the scale of $\mathcal{L}_{\text{lp-ft}}$ on each source domain in each ODG task of Office-Home dataset. Each row of the figure refers to a different ODG task: ACP-R, ACR-P, APR-C and CPR-A. Plots with higher opacity are smoothed plots using the exponential moving average. As shown in the table, RPF shows a higher loss scale than LP-FT on each source domain of each ODG task. Even though the two methods show different loss scales, they both converge to 100% train accuracy. RPF showing a higher cross-entropy loss on the source domain samples seen during training implies that the model is less overfitted to the source domains and its predicted class probability distribution is more smooth rather than peaky compared to LP-FT. We believe that RPF learns a more generalizable representation because it is less biased toward the source domains.

Appendix H. Maximum confidence score histograms

Fig. 6 7 and 8 show the histograms of maximum confidence score for known and unknown classes over the four target domains of each ODG benchmark. A similar trend is observed from all three benchmarks, which is that RPF outputs a lower maximum confidence score on the unknown classes compared to LP-FT. RPF also shows a relatively lower maximum confidence score on known classes as well. LP-FT exhibits a tendency of showing overly confident predictions for both known and unknown classes while RPF shows rather under-confident predictions. It indicates that the predicted softmax output of RPF has more smooth distribution and higher entropy. Higher entropy implies that the model is not too biased towards the source domains samples and classes seen during training.

Appendix I. Threshold robustness analysis

For a fair comparison, we follow the evaluation protocol of Shu et al. (2021) which is to measure the H-score using the threshold that shows the best score. Therefore, in Fig. 10, 9 and 11, we present the H-score of both RPF and LP-FT using 8 different thresholds with equal interval to show the robustness to change in the threshold of each method. The results are the average of three rusn of experiments. We observe that RPF achieves a better H-score than LP-FT on most of the thresholds while it shows lower H-scores on high thresholds. The reason why RPF shows lower H-scores on high thresholds is because it predicts less confidently on the known classes as shown in the previous section, thus classifying most of the known classes as unknowns when high threshold is employed.

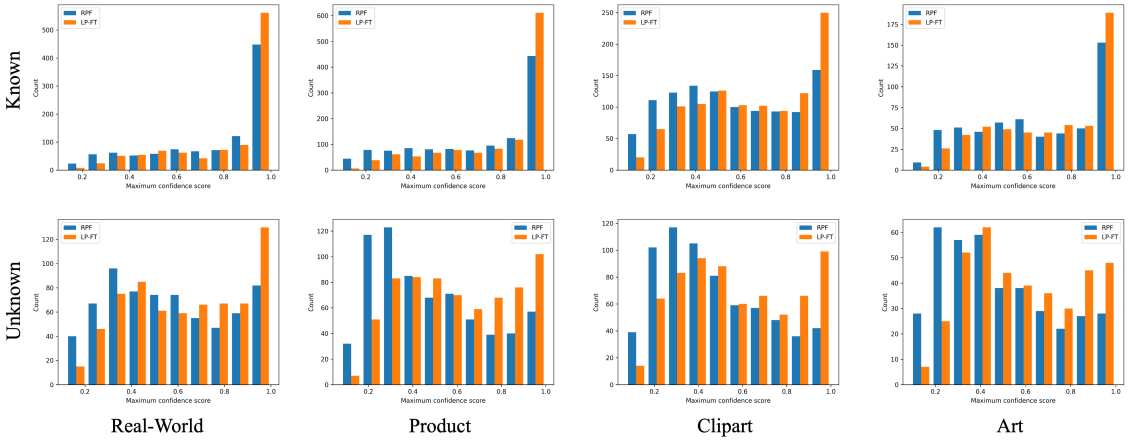

Figure 6: Histograms of known and unknown classes on the four target domains of the Office-Home dataset.

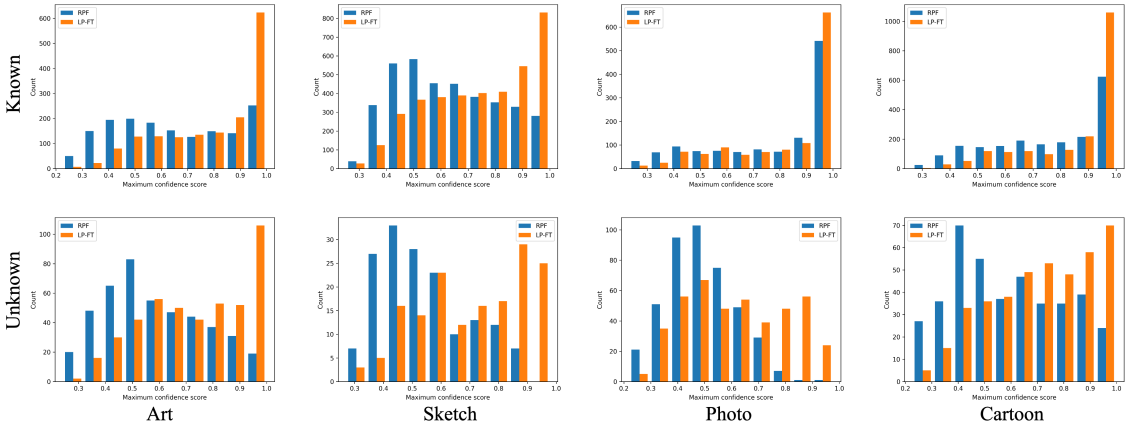

Figure 7: Histograms of known and unknown classes on the four target domains of the PACS dataset.

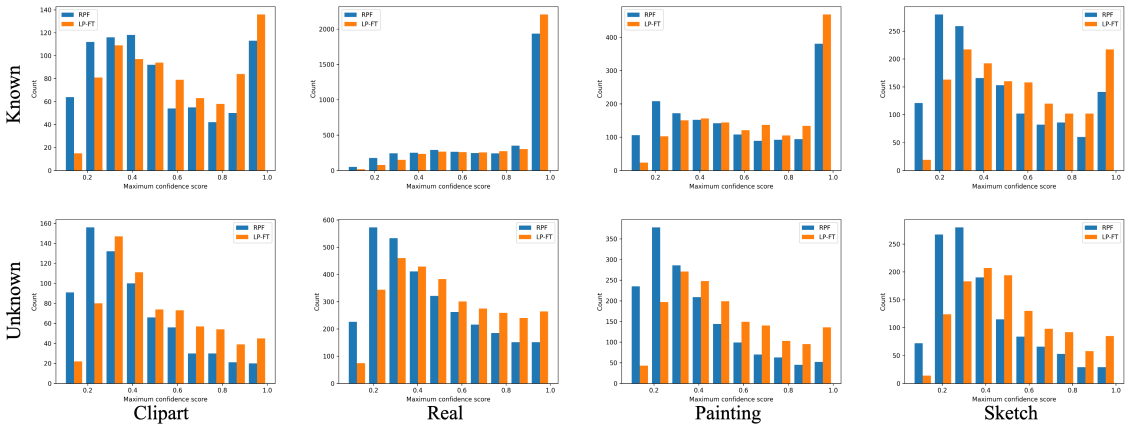

Figure 8: Histograms of known and unknown classes on the four target domains of the Multi-Datasets scenario.

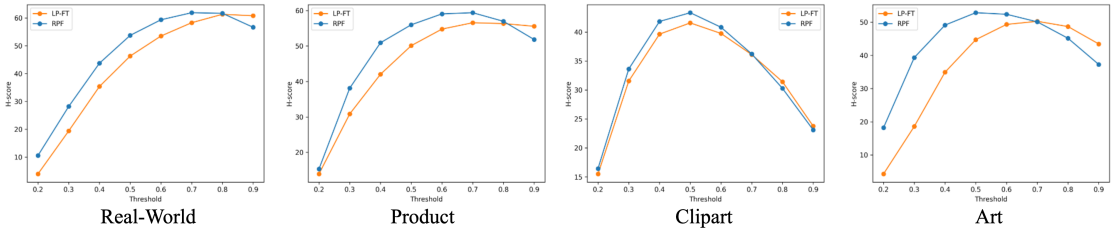

Figure 9: Threshold robustness analysis on the the Office-Home dataset.

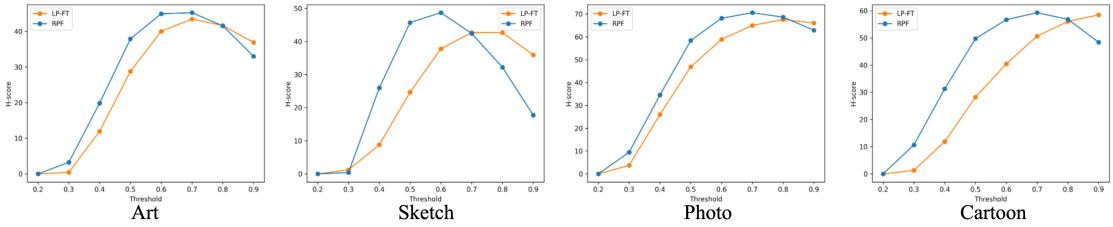

Figure 10: Threshold robustness analysis on the the PACS dataset.

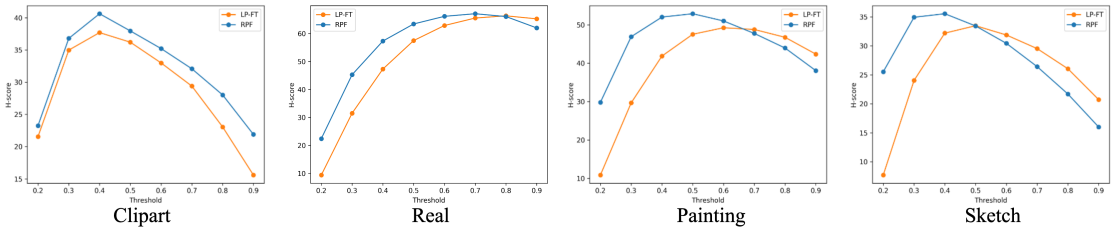

Figure 11: Threshold robustness analysis on the the Multi-Datasets scenario.