Exploring and Utilizing Pattern Imbalance

Shibin Mei, Chenglong Zhao, Shengchao Yuan, Bingbing Ni*
Shanghai Jiao Tong University, Shanghai 200240, China
{adair327, cl-zhao, sc_yuan, nibingbing}@sjtu.edu.cn

Abstract

In this paper, we identify pattern imbalance from several aspects, and further develop a new training scheme to avert pattern preference as well as spurious correlation. In contrast to prior methods which are mostly concerned with category or domain granularity, ignoring the potential finer structure that existed in datasets, we give a new definition of seed category as an appropriate optimization unit to distinguish different patterns in the same category or domain. Extensive experiments on domain generalization datasets of diverse scales demonstrate the effectiveness of the proposed method.

1. Introduction

Over the past decade, the rise of deep neural networks (DNNs) has promoted the rapid development of various artificial intelligence communities [13, 20, 22]. Despite the remarkable success, DNNs tend to take shortcuts to learn spurious features [24, 27]. The causal correlation between these spurious features and ground truth only exists in the training set, which hinders the generalization of DNN models. This phenomenon is also known as domain shift. Moreover, due to the incomplete distribution of training data, the learned model may have a preference for gender, race, and skin color, which will lead to serious ethical problems.

To tackle these problems, various methods have been proposed to discuss the failure modes of out-of-distribution (OOD) generalization [18, 30, 32, 43]. Some researchers focus on encouraging the model to learn domain invariant features. Ganin et al. [9] simultaneously optimize a standard classifier and a domain classifier through adversarial training, where the features extracted by DNN can be used for original classification but failed on domain recognition to inhibit domain characteristics learning. Arjovsky et al. [1] restrict the learned representations to be classified by similar optimal linear classifiers in different domains. Other researchers start by avoiding spurious features. Zhang et al. [42] argue that there exist sub-networks with preferable domain generalization ability in the model and represent the sub-network through a learnable mask. Nam et al. [28] assume that the spurious features are generally embodied in the texture or style of the image. They design SagNet to decouple the content and style of the image, impelling the feature extractor to pay more attention to the content information. Most of the above methods manually design specific model structures to handle domain generalization.

Instead of designing specific networks, we are more concerned about solving domain generalization by exploring the character of the dataset. In particular, suppose a simple handwritten digit recognition scenario, where a large amount of digit 0 possesses the red background and digit 1 possesses the green background. The dataset with only the above two patterns cannot be effectively learned, since the model has no idea whether the task is to classify the digits or the background color. Therefore, in a given learnable data set, there must exist a minority of digit 0 with green background and digit 1 with red background. These samples play a significant role in establishing the true causal relationship between images and labels but have not been paid enough attention. We call pattern imbalance the phenomenon that different patterns in the same class appear imbalanced, thus leading to model learning preference. Based on the above observations, we attribute the domain generalization problem to the mining of hard or minority patterns under imbalanced patterns. First of all, we identify the pattern imbalance in the dataset from several perspectives. We note that even though a model has achieved favorable performance on average, Achilles'heel still exists on some weak patterns. To alleviate the influence caused by imbalance patterns, we pay more attention to these samples of minority patterns and propose a training scheme based on dynamic distribution. To this end, we define a new concept, seed category, that is, the inherent pattern to distinguish, to promote model training by paying full attention to various patterns in the data set. Specifically, for samples of the same class, the seed category is divided based on the distance of the samples in the embedding space as a more fine-grained weight allocation unit than previous methods [19, 32, 39].

*Corresponding author.

In this paper, this dynamic and fine-grained training scheme enables our method to obtain excellent domain generalization performance.

We argue that it is effective to apply more detailed weight allocation on out-of-distribution generalization tasks, that is, the patterns that are crucial but laborious to be learned by the model deserve special treatments, which is the most significant difference between our method and the previous methods. Prior methods, e.g., GroupDRO [32], minimize the worst-case loss over domains to treat different domains differently, and the performance will be limited by the coarse granularity of grouped distribution. On the contrary, the flexibility of our method is revealed in two aspects, that is, the weight allocation unit is more detailed and the seed category can be dynamically adjusted during the training process. Our contributions can be summarized as follows:

- We identify pattern imbalance generally existed in classification tasks and give a new definition of seed category, that is, the inherent pattern to recognize.
- We further develop a dynamic weight distribution training strategy based on seed category to facilitate out-of-distribution performance.
- Extensive experiments on several domain generalization datasets well demonstrate the effectiveness of the proposed method.

2. Related Work

Algorithms. Many methods have been proposed to enhance domain generalization ability on out-of-distribution (OOD) datasets. Peter et al. [29] and Carulla et al. [31] demonstrate that in the linear model, learning invariant features in the training set is conducive to finding invariant features in the test set. Therefore, a common strategy is to invariant features in visible domains, including minimizing the maximum average difference [26] and adversarial feature alignment [19]. Ganin et al. [9] utilize adversarial learning to train a domain classifier jointly with the original classifier, and apply the gradient inversion layer to reduce the domain information embedded in extracted representations. Arjovsky et al. [1] manage to learn domain invariant features that can be classified by an optimal linear classifier in all domains. Sagawa et al. [32] assume that the key to learning out-of-distribution samples is to minimize the worst-case group loss, and they also show stronger regularization should be applied to narrow the generalization gap. Yan et al. [40] enhance the domain generalization ability of the model through the mixup of inter-domain samples. Wang et al. [38] propose to use knowledge distillation to improve the generalization ability through a smoother model, where the student network learns richer features by learning the soft labels crafted from the teacher network.

Interpretation. Based on spurious features, many arts have proposed insightful explanations and effective algorithms. Zhang et al. [42] believe that even if the model learns spurious features, there still exist sub-networks that have favorable domain generalization performance. They call this hypothesis lottery theory. Nagarajan et al. [27] analyze the reasons why the network can learn spurious features in detail. They experimentally demonstrated that even if invariant features are fully predictive of the label while the spurious features can not, or the invariant features can easily accomplish the classification through an ordinary classifier, the network will still learn spurious features. They argue that learning spurious features is an inherent nature of optimization algorithms.

Learnability. In addition to studying domain generalization algorithms, quite a few works are committed to exploring the learnability of domain generalization problems. A more direct approach is to measure the distance between the training domain and the test domain [2, 5]. Another feasible framework is causal analysis, which has high robustness to the test domain shift caused by variable intervention [12]. Ye et al. [41] propose a unified framework to analyze the domain generalization problem. They define the domain generalization problem theoretically through a series of concepts such as variation, informativeness, and extension function, and further give the learnability of the domain generalization problem. They also present the upper and lower bounds of the generalization error through thorough theoretical analysis.

3. Methodology

In this section, we will discuss in detail the identification of pattern imbalance and the definition of seed category. We first revisit the basic concept of domain generalization problem, and then verify the existence of pattern imbalance from the perspective of representations, activation path, and model optimization. Several prior methods to deal with data imbalance are introduced to compare with our views. A dynamic distribution algorithm based on seed category is developed to boost domain generalization performance and we further give related theoretical analysis.

3.1. Preliminary

We give the concept of domain generalization following prior works [27, 41, 42]. The basic paradigm of the domain generalization problem originates from multi-class classification task $X \to Y = \{1, 2, ..., K\}$, where K represents the number of total categories. Assume that the data available in the training stage is $\mathcal{E}_{seen}$ while the inaccessible data during model training is $\mathcal{E}_{unseen}$. The performance of the model is evaluated on all domains $\mathcal{E}_{all}$, where

$\mathcal{E}_{seen} \subset \mathcal{E}_{all}$. For a certain visible domain e in $\mathcal{E}_{seen}$, the data point (X^e, Y^e) in e can be seen as an independent identically distributed sample from the corresponding dataset $D_e = \{X_i^e, Y_i^e\}_{e=1}^{n^e}$. For a given neural network $f_\theta : X \to Y$ which has learned the mapping from samples to ground truth to some extent, where $\theta \in \Theta$ represent model parameters, we can thus define the loss on a domain e as,

$$R^e(\theta) = \mathbb{E}[\mathcal{L}(f_\theta(X^e), Y^e)] \tag{1}$$

, where $\mathcal{L}$ denotes loss function, e.g., cross-entropy loss. The objective of the domain generalization problem is minimizing the worst case (worst domain) classification loss, i.e.,

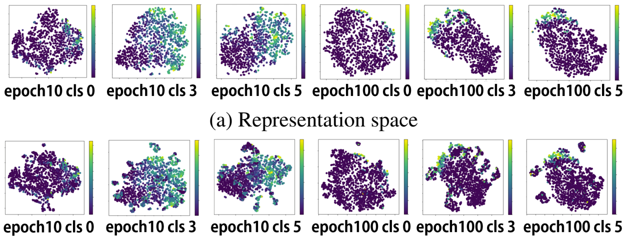
$$\min_{\theta\in\Theta} \max_{e\in\mathcal{E}} R^e(\theta). \tag{2}$$

Since we can only obtain the data in the visible domain $\mathcal{E}_{seen}$ in the training process, there exists an unknown domain shift between the visible domain $\mathcal{E}_{seen}$ and the invisible domain $\mathcal{E}_{unseen}$. Domain generalization, as a challenging problem, has attracted the attention of many researchers.

3.2. Identifying Pattern Imbalance

We assume that some patterns that the model is difficult to capture hinder the generalization to out-of-distribution samples. A learnable domain generalization dataset must contain causal correlations between samples and labels, and the model may neglect the causality presented by some key patterns, thus leading to the susceptibility to domain shift. Previous works propose to improve the generalization ability based on the noticeable imbalance on categories [11] and domains [32], and we experimentally confirm that a more fine-grained pattern imbalance exists. The model shows poor performance on these modes, even if it has high accuracy on average. In this section, we will identify the pattern imbalance of neural networks on datasets from three aspects.

Data view. The dataset displays pattern imbalance in the representation space of the neural network. Similar to the previous works [7, 21], we consider that the model can be decoupled into a feature extractor and a classifier. From the perspective of representation learning [3], the level of inter-class dispersion and intra-class compactness of the feature space determines the quality of representations. Therefore, we investigate the pattern imbalance from the output representations. We choose a full-trained model on the CIFAR10 dataset to conduct experiments. It can be observed from Fig. 1(a) that the representations differ in optimization extent of identical categories, despite that the model possesses high accuracy on average (e.g., epoch 100). We use the color of data points to represent its classification loss (the loss has been re-scaled for convenience of visualization) and use TSNE [35] to facilitate visualization.



(a) Representation space

(b) Activation path

Figure 1. Dimension reduction results of representation space and activation path on CIFAR10 dataset (class 0-airplane, 3-cat, 5-dog) of different training epochs. We can observe from the figure that pattern imbalance appears in the representation space and activation path, which indicates the widespread of this phenomenon, despite that the model possesses favorable performance on average. The data point color represents its classification loss.

Model view. Different patterns present different activation paths in the neural network. Taking the ResNet34 model trained on the CIFAR10 dataset as an example, we extract the activation map output by the network respectively after the initial convolution layer, four basic block stages, and fully-connected layers. For each activation map, we select the index of the corresponding channel with the highest average activation as a component of the activation path. In this way, we utilize a 6-dimension vector to characterize the activation path of data flow in the network. Fig. 1(b) displays that after dimension reduction, the pattern imbalance also exists in the activation path.

Optimization view. Different patterns have different training extent. As mentioned above, we use the classification loss to characterize the optimization extent of corresponding data samples.

It is easy to notice that the above discussion cannot fully explain the pattern imbalance, since there is no explicit alignment among different views. In order to further prove the existence of pattern imbalance, we explore the approach to depicting the alignment of these views. We apply clustering results to explicitly portray the pattern imbalance. We respectively cluster the representations, sample activation paths and classification loss from three views using K-means algorithm [23] of the same group data (i.e. same classes and training epoch), and specify the number of clustering centers as K. We subsequently define the Cluster Consistency (CCT) of two clustering results A and B as the conditional probability that for any two data samples, they are in the same cluster in results B on the condition that they are also in the same cluster in results A, i.e.,

$$CCT_{AB} = \mathbb{E}_{g\in\mathcal{G}}\mathbb{E}_{s_i,s_j\in g,R^A_{s_i}=R^A_{s_j}}[\mathbb{1}(R^B_{s_i} = R^B_{s_j})] \tag{3}$$

, where R^A represents a clustering result and $R^A_{s_i} = R^A_{s_j}$ represents sample s_i, s_j are in the same cluster. g represents one data group and $\mathcal{G}$ represents all data groups. The

Pair	Random	Rep-Path	Rep-Loss	Path-Loss
CCT	0.118	0.241	0.577	0.508

Table 1. The cluster consistency metric conducted on the CIFAR10 dataset. Rep-Path represents the CCT of clustering results of representation and activation path. Random represents the CCT of two randomly divided data groups. We set the number of clusters K as 10.

cluster consistency metric CCT quantifies the alignment degree of clustering results. We conduct experiments on the CIFAR10 dataset. It can be observed from Tab. 1 that the consistency metric among three clustering results of three views is much higher compared with randomly grouped results, which further verifies the existence of pattern imbalance in the dataset. Details about the implementations can be found in the supplementary materials.

3.3. Fine Grained Perspective

Previous data-driven generalization algorithms show bias on some specific groups based on category imbalance or domain imbalance. The Distributionally Robust Optimization (DRO) based algorithm [15,32] is to minimize the worse domain loss. Traditional class-imbalance-based algorithms [6, 11], conduct over-sampling on long-tailed classes using mixup to generate new samples or under-sample majority classes. In the previous section, we find more fine-grained pattern imbalance and this phenomenon can not be alleviated simply by taking special treatment on domain granularity or category granularity. Moreover, previous arts on Online Hard Example Mining (OHEM) [33] directly select samples with top K large loss in a batch for gradient backpropagation. However, we argue that the model should not learn the knowledge reflected by a single difficult sample, but the knowledge contained in a group of special samples with similar characteristics, that is, more macro patterns. Therefore, the fine-grained perspective introduced by pattern imbalance, as a compromise between the above two granularity, not only solves the imbalance problem in a superior way but also avoids their shortcomings.

3.4. Seed Category

In this section, we define the concept of seed category to more intuitively reveal the phenomenon of pattern imbalance. In the above analysis, the seed category can be simply obtained by clustering the samples in the embedding space. However, in order to facilitate the theoretical analysis and the interpretability of the algorithm, we do not choose to obtain seed categories by clustering.

Intuitively, the grouping criterion of seed categories should be based on the distance of samples in certain embedding space, that is, the farther the distance between two samples is, the more likely they are in the different seed categories, while the closer they are, the more likely they are in the same seed categories. For any two samples s_i, s_j, we define the distance between them as $D(s_i, s_j) = D_m(\Psi(s_i), \Psi(s_j))$, where D_m represents a distance metric, i.e., L_2 distance and Ψ represents a representation embedding function. We then define two points s_i, s_j are adjacent as they satisfy $D(s_i, s_j) < \xi$. Directly obtaining seed categories from the above definition is difficult since all samples may be indirectly or directly adjacent from a macro point of view. To avoid this ambiguity, we regard all samples and their adjacent relationships as a structure of a graph, where the samples represent the vertices of the graph and the adjacent relationships between samples represent the edges between vertices. In this way, we can conveniently and intuitively give the definition of seed,

Definition 1 *(Seed) For a given dataset $\mathcal{D}$ and the above concept of adjacent samples, we define seed set $\mathcal{S}$ if and only if for any sample $x \in \mathcal{D} - \mathcal{S}$, there exists at least one $s \in \mathcal{S}$ that satisfies x and s are adjacent samples. We define each sample in the smallest seed set $\mathcal{S}$ as seed.*

We presume that each seed represents a possible pattern in the dataset. We can control the number of seeds by adjusting the threshold ξ. With the definition of seed, we can readily define the seed category in the dataset. For each sample that does not belong to the smallest seed set $\mathcal{S}$, we randomly select a seed adjacent to it. We call that this sample is subordinate to the corresponding seed. We can then get the definition of seed category:

Definition 2 *(Seed Category) For any seed s, we define the union of this seed and the samples $\mathcal{B}_s$ subordinate to it as seed category, i.e.,*

$$\mathcal{C}_{seed} = \{s\} \cup \mathcal{B}_s. \tag{4}$$

In this way, the dataset can be divided into several seed categories, and the seed categories correspond to the seeds one by one. We consider that each seed category represents a pattern in the original dataset, so we can regard the seed category as the basic optimization unit of the algorithm. We further develop a dynamic distribution allocation algorithm based on seed category.

The dynamic distribution allocation algorithm is illustrated as follows. Firstly, considering the training stability and reducing the computational cost, we perform the redistribution, i.e., update the seed categories, every t_0 steps. Specifically, we recalculate the seed categories according to representations or classification loss. Suppose the number of seed categories is $|\mathcal{C}|$. During two rounds of redistribution, we will maintain a distribution weight vector $q \in \mathbb{R}^{|\mathcal{C}|}$ initialized to all one. Secondly, in each normal step, we randomly select a data batch from each seed category with

Algorithm 1 Dynamic Distribution Based on Seed Category

```
Require: Original data $\mathcal{D}$, total training steps $T$, re-distribution cycle $t_0$
Ensure: Model parameters $\theta$
for t= 1 to $T$ do
    Conduct re-distribution and update seed category for $\mathcal{D}$ every $t_0$ steps.
    Maintain a weight distribution vector $q$ on these seed categories.
    Sample data from each seed category as a batch.
    Calculate average losses and update weights for each seed category.
    Weight normalization.
    Update model parameters $\theta$ according to distribution weight vector $q$.
end for
```

batch size B. The merged batch ($|\mathcal{C}| \times B$ samples) collected from all seed categories is then sent to the model for evaluation to obtain the average classification loss of each seed category. The corresponding position of the distribution vector q is then updated according to the exponent of classification loss of each seed category. The distribution vector q is normalized and then served as the weight of the loss of each seed category. Finally, we back-propagate the loss and update the model parameters, as shown in Alg.1.

Implementation details. In graph theory, calculating seed or seed categories of a dataset is an NP-hard problem ($\mathcal{O}(2^{0.61n})$) [8] and there is no effective algorithm to deal with large-scale data at present. Therefore, we consider simplifying the above problems. From the above illustration, we know that the seed category is divided based on the distance of samples in the embedding space. We can of course apply a clustering algorithm to obtain seed categories. However, we provide another convenient approach. We hope the grouping of the seed category will bring as much discrimination between any two distribution of seed categories as possible. For two seed category distributions P and Q, JS divergence [25] can be used to measure their distance as a symmetry criterion, i.e., $JS(P||Q) = \frac{1}{2}KL(P||M) + \frac{1}{2}KL(Q||M)$, where $M = \frac{1}{2}(P+Q)$. The overlap between these two distributions can thus be measured by $O(PQ) = -\mathbb{E}_{x\sim P}[\log(G(F(x)))] - \mathbb{E}_{x\sim Q}[\log(1-G(F(x)))]$, where $G(x) = \frac{1}{1+e^{-x}}$ and $F(x)$ is a feature extractor. This criterion is in the form of binary cross-entropy loss, which also explains the rationality of using classification loss as a sign of hard samples in previous work [33]. The conclusion can be naturally extended to multi-label classification tasks. We now want to seek out a grouping strategy to make the distribution of different seed categories more and more scattered. Since we can further obtain seed categories based on original categories or domains, the classification loss can be sorted and used to obtain seed categories according to their order. In our experiments, we further divided each domain of out-of-distribution datasets into four seed categories. Details about proofs can be found in the supplementary materials.

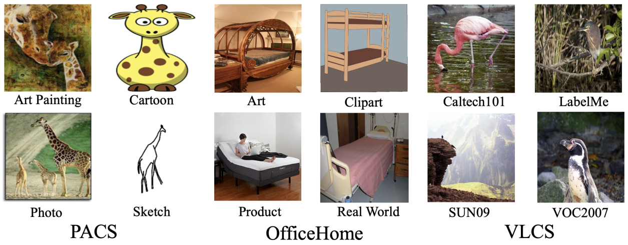


Figure 2. Sample visualization of out-of-distribution datasets (PACS, OfficeHome, VLCS).

Detailed theoretical analysis can be found in the supplementary materials.

4. Datasets and Model Evaluation

In this section, we illustrate the datasets we used and the methods for model selection in detail.

4.1. Datasets

We employ several widely used benchmark out-of-distribution (OOD) datasets to evaluate our methods, i.e., ColoredMNIST [16], RotatedMNIST, PACS [17], OfficeHome [37], VLCS [34]. The original MNIST dataset is handwriting digits of 0-9, and has a training set of 60000 examples, and a test set of 10000 examples. For ColoredMNIST, we first assign a binary label y based on digits ($y = 0$ for digits 0-4, $y = 1$ for digits 5-9) and flip the label with probability 0.25. The images are colored in either red or green according to their label and we flip the color with probability e. We construct three domains where flip probability e are set as 0.1, 0.2, 0.9. For RotatedMNIST, the original images are rotated by an angle and we construct six domains where the rotated angles are set as $0°$, $15°$, $30°$, $45°$, $60°$, $75°$. For PACS, there exist four domains, i.e., Art Painting(2048 images), Cartoon(2344 images), Photo(1670 images) and Sketch(3929 images), and each domain contains seven categories. For OfficeHome dataset, it consists of four domains, i.e., Art, Clipart, Product and Read-World, and each domain consists of 65 categories. For VLCS dataset, it includes four domains, that is, Caltech101, LabelMe, SUN09, VOC2007, where each domain consists of five categories. Some examples in these datasets can be seen in Fig. 2.

Algorithm	ERM	DANN	IRM	GDRO	MLDG	MMD	MTL	ARM	SagNet	VREx	Ours
Train-domain validation set											
A	81.1	81.9	82.7	86.2	81.8	84.9	82.5	82.1	83.0	81.4	82.0
C	81.6	77.8	78.5	80.5	80.0	81.0	79.9	82.9	78.6	81.4	80.3
P	97.0	95.1	96.4	96.2	95.3	95.7	95.5	93.6	95.3	96.0	96.8
S	74.1	75.4	74.3	75.3	69.5	73.3	79.6	76.0	80.7	77.8	80.8
Avg.	83.5	82.6	83.0	84.5	81.6	83.7	84.4	83.6	84.4	84.2	**85.0**
Leave-one-domain-out cross validation											
A	82.7	79.0	82.7	81.3	82.6	81.6	80.0	77.2	80.7	81.8	83.5
C	80.0	76.1	78.5	76.8	79.5	80.8	80.3	82.9	78.1	79.9	79.9
P	95.3	95.1	96.4	94.8	97.7	95.1	96.7	93.1	95.7	95.4	96.2
S	75.3	72.4	74.3	80.3	70.5	71.7	74.6	73.2	63.6	72.8	83.2
Avg.	83.3	78.6	83.0	83.3	82.5	82.3	82.9	81.6	79.6	82.4	**85.7**
Test-domain validation set											
A	80.9	74.0	72.4	72.4	82.6	81.2	84.5	73.5	77.5	81.8	80.2
C	81.2	75.6	77.1	79.0	81.3	81.7	76.1	76.7	78.6	79.7	79.6
P	95.1	91.0	90.9	94.8	94.9	95.1	94.2	94.7	95.7	95.3	94.4
S	78.1	76.1	74.1	72.6	73.2	78.8	74.6	70.6	77.4	76.3	84.3
Avg.	83.8	79.2	78.9	79.7	83.0	84.2	82.3	78.9	82.3	83.3	**84.6**

Table 2. Evaluation of the domain generalization ability on PACS dataset. We present the results of three model selection methods for each domain (A, C, P, S) and average accuracy (Avg.) for a comprehensive assessment. All experiments are conducted on the ResNet50 model. We compare our method with recently proposed OOD algorithms, i.e., ERM, DANN, IRM, GDRO(GroupDRO), MLDG, MMD, MTL, ARM, SagNet, and VREx. All these OOD algorithms are implemented as their official settings. All values are shown in percentages.

4.2. Model Selection

Since the data distribution of the test set and validation set is not identical in the domain generalization problem, the optimal model selection method is not as straightforward as traditional supervised learning. Following [10], we employ three model selection methods stated below.

Train-domain validation set. Each training domain is further divided into a training subset and a validation subset. All the validation subsets are put together to form the validation set and all the training subsets are put together to form the training set. We finally select the model that performs best in the validation set.

Leave-one-domain-out cross validation. Each time we leave one training domain as the validation domain. Assuming there are N domains, we want to employ each training domain as the leave-out domain each training round, and finally select the model with the best average performance in all leave-out domains. We then conduct model training in all training domains with the optimal hyper-parameter and regard this model as the final selected model.

Test-domain validation set. The selected model is expected to possess favorable performance in the test domain, so here we use samples in the test domain for model evaluation. Note that the model should not have reached the test domain, it is necessary to strictly control the number of test samples the model can assess, such as several limited queries. Since each validation consumes queries, this method does not allow early stopping and only performs model evaluation in the last step.

5. Experiments

In this section, we conduct extensive experiments to evaluate the performance of the proposed method, which well demonstrates the effectiveness of our method compared with recent works.

Settings. We choose ResNet50 as our model architecture as [10,41]. For each model training, we run 5000 steps with an initial learning rate of 1×10^{-3} for MNIST datasets and 5×10^{-5} for large-scale datasets. We re-calculate the seed category every 1000 steps and the batch size of each seed category is adjusted to satisfy a fixed total batch size. For large-scale datasets, we apply data augmentations, such as random resized crop, random horizontal flip, color jitter and random grayscale. All images are resized to a fixed size of 224×224 and normalized to facilitate faster convergence. For training stability, before we apply the exponential operation to the classification loss, we multiply the loss with $\eta = 0.01$ following [32]. For seed category calculation, we conduct experiments using K-means clustering [23] and loss ranking as illustrated in implementation details, and we display results of the latter setting in Sec. 5.1 and Sec. 5.2 and compare these two settings in Sec. 5.3.

OOD Algorithms. We compare our proposed method with recent OOD algorithms, including ERM (Empirical Risk

Algorithm	ERM	DANN	IRM	GDRO	MLDG	MMD	MTL	ARM	SagNet	VREx	Ours
Train-domain validation set											
0.1	71.6	71.5	61.0	72.5	72.4	49.3	71.8	72.8	71.4	72.3	72.8
0.2	72.6	73.2	67.8	72.2	72.6	63.4	71.6	72.6	73.5	73.0	73.2
0.9	9.8	10.0	9.8	10.2	10.1	10.8	10.2	10.1	10.3	10.2	10.7
Avg.	51.3	51.6	46.2	51.6	51.7	41.2	51.2	51.8	51.7	51.8	**52.2**
Leave-one-domain-out cross validation											
0.1	61.4	50.9	48.8	50.9	49.2	49.5	47.8	47.4	50.7	72.2	55.6
0.2	50.5	50.1	50.2	51.0	53.3	50.6	66.8	50.6	49.4	50.7	71.8
0.9	9.8	10.0	9.8	10.2	10.1	9.8	10.2	10.1	10.3	10.0	10.2
Avg.	40.6	37.0	40.6	37.3	37.5	36.7	41.6	36.0	36.8	44.3	**45.9**
Test-domain validation set											
0.1	64.6	69.7	51.0	67.1	70.1	50.5	67.2	77.8	64.6	72.2	71.8
0.2	68.5	70.6	62.3	68.4	71.0	50.6	68.8	70.3	68.7	71.7	72.5
0.9	26.5	13.5	50.6	38.5	23.8	10.3	20.1	17.5	28.4	29.4	37.1
Avg.	53.2	51.3	54.7	58.0	54.9	37.1	52.1	55.2	53.9	57.8	**60.5**

Table 3. Evaluation of the domain generalization ability of the proposed method compared with recent OOD algorithms on the ColoredMNIST dataset. 0.1, 0.2, 0.9 represent the color flip probability of three domains. All values are shown in percentages.

Minimization) as baseline [36], IRM (Invariant Risk Minimization) [1], GroupDRO (Group Distributionally Robust Optimization) [32], MLDG (Meta Learning Domain Generalization) [18], MMD (Maximum Mean Discrepancy) [19], DANN (Domain Adversarial Neural Network) [9], ARM (Adaptive Risk Minimization) [43], SagNet (Style Agnostic Networks) [28], MTL (Marginal Transfer Learning) [4], VREx (Variance Risk Extrapolation) [14]. All these OOD algorithms are implemented as their official settings.

5.1. Performance on Large Scale Dataset

In this section, we evaluate our method on several large-scale out-of-distribution datasets, i.e., PACS, OfficeHome and VLCS. Tab. 2 shows our experimental results on the PACS dataset. We evaluate our method compared with ERM, DANN, IRM, GDRO(GroupDRO), MLDG, MMD, MTL, ARM, SagNet, VREx. For a comprehensive evaluation, we employ the above three model selection methods, train-domain validation set, leave-one-domain-out cross-validation and test-domain validation set. We display the results for each environment setting (A, C, P, S) and calculate the average accuracy for the final evaluation. To facilitate observation, all results are shown in percentages. As observed in the table, our method presents outstanding performance compared with other methods, and the average accuracy have achieved 85.0%, 85.7%, 84.6% on three model selection methods, which well verified the effectiveness of using seed categories as distribution allocation unit to mine the pattern imbalance. We also display the domain generalization performance on the OfficeHome dataset and VLCS dataset, as shown in Tab. 6 and Tab. 7. For OfficeHome and VLCS datasets, we only display the results of the leave-one-domain-out cross-validation model selection method due to space limitation.

5.2. Performance on ColoredMNIST and RotatedMNIST

We then conduct experiments on ColoredMNIST and RotatedMNIST datasets. For ColoredMNIST, we evaluate the domain generalization ability for each environment, that is, different color flip probabilities of 0.1, 0.2, 0.9. We displayed results on three model selection methods. As shown in Tab. 3, our method shows favorable domain generalization performance compared with recently proposed methods. Especially, our method shows superior performance under the test-domain validation set model selection setting, which is also used in [42]. This phenomenon demonstrates that our method can sensitively capture the unbalanced patterns in the training data set, which is easily ignored in the original training scheme. For RotatedMNIST, we also only display the results of the leave-one-domain-out cross-validation model selection method, as shown in Tab. 8.

5.3. Redistribution Method Analysis

We further explore how different seed category calculation methods will affect the domain generalization performance. As illustrated above, we provide two approaches, i.e., using ranking classification loss or employing clustering. We also explore the number of seed categories that one original domain should be divided (We test 2,3,4 since more divided groups will damage the parallelism of the algorithm). As shown in Tab. 4, the loss-based redistribution method with four seeds per domain performs best.

Datasets	ColoredMNIST						PACS					
Methods	Loss			Cluster			Loss			Cluster		
Seeds per domain	2	3	4	2	3	4	2	3	4	2	3	4
Model Selection 1	51.5	51.5	**52.2**	51.6	51.6	51.9	82.9	83.0	**85.0**	82.5	82.6	82.6
Model Selection 2	38.5	41.8	**45.9**	37.3	37.3	37.5	82.8	84.2	**85.7**	83.5	83.5	83.8
Model Selection 3	59.2	57.4	**60.5**	54.8	59.3	53.0	83.1	79.6	**84.6**	82.2	81.8	80.9

Table 4. Performance vs. seed category calculation methods.

Datasets	CMNIST		RMNIST		PACS		OfficeHome		VLCS	
Methods	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
PCMA(G)	0.186	0.186	0.437	0.437	8.112	8.124	8.113	8.128	8.117	8.125
TTR(min)	6.18	7.14	7.75	9.62	44.15	48.14	66.53	71.19	76.57	81.71

Table 5. Efficiency of our method compared with baseline. PCMA represents peak CUDA memory allocated and TTR represents the time of a training round.

Algo.	A	C	P	R	Avg.
ERM	54.7	47.3	72.7	74.0	62.2
DANN	54.3	51.1	73.0	67.4	61.5
IRM	55.4	49.1	68.2	75.0	61.9
GDRO	55.7	52.0	71.4	74.7	63.4
MTL	53.0	47.1	70.5	76.3	61.7
ARM	51.9	46.8	69.8	71.0	59.9
SagNet	53.1	49.0	72.5	73.4	62.0
Ours	57.5	50.4	73.2	74.0	**63.8**

Table 6. Evaluation on OfficeHome datasets. All values are shown in percentages. A, C, P, R represent four domains of OfficeHome, i.e., Art, Clipart, Product, Real World.

Algo.	C	L	S	V	Avg.
ERM	98.1	59.0	70.1	74.6	75.4
DANN	98.5	63.0	56.9	74.6	73.2
IRM	95.4	59.3	74.2	76.0	76.2
GDRO	94.9	66.3	69.7	71.3	75.6
MTL	96.1	59.5	70.0	73.0	74.6
ARM	94.6	62.9	74.0	70.3	75.4
SagNet	93.4	60.4	75.1	75.0	76.0
Ours	97.4	66.4	70.1	72.8	**76.7**

Table 7. Evaluation on VLCS datasets. All values are shown in percentages. C, L, S, V represent four domains of VLCS, i.e., Caltech101, LabelMe, SUN09, VOC2007.

Algo.	0	15	30	45	60	75	Avg.
ERM	88.2	98.4	99.1	99.0	98.9	95.8	96.6
IRM	78.9	77.4	89.6	93.3	93.3	82.7	85.9
GDRO	95.5	97.9	97.6	98.6	98.8	96.1	97.4
ARM	91.3	98.8	98.9	99.1	98.7	95.9	97.1
Ours	94.8	99.6	99.0	99.1	98.8	96.6	**98.0**

Table 8. Experimental results on RotatedMNIST datasets. 0, 15, 30, 45, 60, 75 represent the angle that images will rotate in corresponding domains.

5.4. Algorithm Efficiency

Since our proposed method introduces extra computation about re-calculating seed categories, we thus investigate the algorithm efficiency regarding training time and CUDA memory consumption, experimental results are summarized as Tab. 5. All training rounds run 5000 steps on a Tesla V100 32GB GPU with the same domain left out. We can observe that our method consumes little extra time and has almost no increase in CUDA memory.

6. Conclusions

In this paper, we identify the pattern imbalance that generally existed in datasets from three views and then compare this fine-grained perspective with prior works. We further give a new definition of seed category to concretize the concept of pattern imbalance. We finally provide an effective algorithm based on periodically calculating seed categories to boost domain generalization. We firstly propose this fine-grained optimization perspective, where dynamic distribution allocation is conducted based on seed category to solve the problem of domain generalization by exploring the character of the dataset. Extensive experiments on several benchmark out-of-distribution datasets well demonstrate the superior performance of our proposed method.

Acknowledgement. This work was supported by National Science Foundation of China (U20B2072, 61976137). This work was also partly supported by SJTU Medical Engineering Cross Research Grant YG2021ZD18.

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 1, 2, 7

[2] J Andrew Bagnell. Robust supervised learning. In *AAAI*, pages 714–719, 2005. 2

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. 3

[4] Gilles Blanchard, Aniket Anand Deshmukh, Urun Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *arXiv preprint arXiv:1711.07910*, 2017. 7

[5] Jose Blanchet, Yang Kang, Karthyek Murthy, and Fan Zhang. Data-driven optimal transport cost selection for distributionally robust optimization. In *2019 winter simulation conference (WSC)*, pages 3740–3751. IEEE, 2019. 2

[6] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002. 4

[7] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019. 3

[8] Fedor V Fomin, Fabrizio Grandoni, Artem V Pyatkin, and Alexey A Stepanov. Bounding the number of minimal dominating sets: a measure and conquer approach. In *International Symposium on Algorithms and Computation*, pages 573–582. Springer, 2005. 5

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. 1, 2, 7

[10] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020. 6

[11] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009. 3, 4

[12] Christina Heinze-Deml and Nicolai Meinshausen. Conditional variance penalties and domain shift robustness. *arXiv preprint arXiv:1710.11469*, 2017. 2

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[14] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021. 7

[15] Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations research & management science in the age of analytics*, pages 130–166. Informs, 2019. 4

[16] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[17] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017. 5

[18] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 1, 7

[19] 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, 2018. 1, 2, 7

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 1

[21] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017. 3

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[23] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967. 3, 6

[24] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning*, pages 7313–7324. PMLR, 2021. 1

[25] ML Menéndez, JA Pardo, L Pardo, and MC Pardo. The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318, 1997. 5

[26] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2013. 2

[27] Vaishnavh Nagarajan, Anders Andreassen, and Behnam Neyshabur. Understanding the failure modes of out-of-distribution generalization. *arXiv preprint arXiv:2010.15775*, 2020. 1, 2

[28] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8690–8699, 2021. 1, 7

[29] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):947–1012, 2016. 2

[30] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. *arXiv preprint arXiv:2109.02934*, 2021. 1

[31] Mateo Rojas-Carulla, Bernhard Schölkopf, Richard Turner, and Jonas Peters. Invariant models for causal transfer learning. *The Journal of Machine Learning Research*, 19(1):1309–1342, 2018. 2

[32] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019. 1, 2, 3, 4, 6, 7

[33] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016. 4, 5

[34] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011. 5

[35] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 3

[36] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999. 7

[37] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017. 5

[38] Yufei Wang, Haoliang Li, Lap-pui Chau, and Alex C Kot. Embracing the dark knowledge: Domain generalization using regularized knowledge distillation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2595–2604, 2021. 2

[39] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6502–6509, 2020. 1

[40] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020. 2

[41] Haotian Ye, Chuanlong Xie, Tianle Cai, Ruichen Li, Zhenguo Li, and Liwei Wang. Towards a theoretical framework of out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 6

[42] Dinghuai Zhang, Kartik Ahuja, Yilun Xu, Yisen Wang, and Aaron Courville. Can subnetwork structure be the key to out-of-distribution generalization? In *International Conference on Machine Learning*, pages 12356–12367. PMLR, 2021. 1, 2, 7

[43] Marvin Zhang, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn. Adaptive risk minimization: Learning to adapt to domain shift. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 7