

DomainDrop: Suppressing Domain-Sensitive Channels for Domain Generalization

Jintao Guo^{1,2} Lei Qi^{3,*} Yinghuan Shi^{1,2,*}

¹ State Key Laboratory for Novel Software Technology, Nanjing University

² National Institute of Healthcare Data Science, Nanjing University

³ School of Computer Science and Engineering, Southeast University

guojintao@smail.nju.edu.cn, qilei@seu.edu.cn, syh@nju.edu.cn

Abstract

Deep Neural Networks have exhibited considerable success in various visual tasks. However, when applied to unseen test datasets, state-of-the-art models often suffer performance degradation due to domain shifts. In this paper, we introduce a novel approach for domain generalization from a novel perspective of enhancing the robustness of channels in feature maps to domain shifts. We observe that models trained on source domains contain a substantial number of channels that exhibit unstable activations across different domains, which are inclined to capture domain-specific features and behave abnormally when exposed to unseen target domains. To address the issue, we propose a DomainDrop framework to continuously enhance the channel robustness to domain shifts, where a domain discriminator is used to identify and drop unstable channels in feature maps of each network layer during forward propagation. We theoretically prove that our framework could effectively lower the generalization bound. Extensive experiments on several benchmarks indicate that our framework achieves state-of-the-art performance compared to other competing methods. Our code is available at <https://github.com/lingeringlight/DomainDrop>.

1. Introduction

Deep neural networks (DNNs) have shown impressive performance in computer vision tasks over the past few years. However, this performance often degrades when the test data follows a different distribution from the training data [29]. This issue, known as domain shift [39], has

*Corresponding authors: Yinghuan Shi and Lei Qi. The work is supported by NSFC Program (62222604, 62206052, 62192783), China Postdoctoral Science Foundation Project (2023T160100), Jiangsu Natural Science Foundation Project (BK20210224), and CCF-Lenovo Bule Ocean Research Fund.

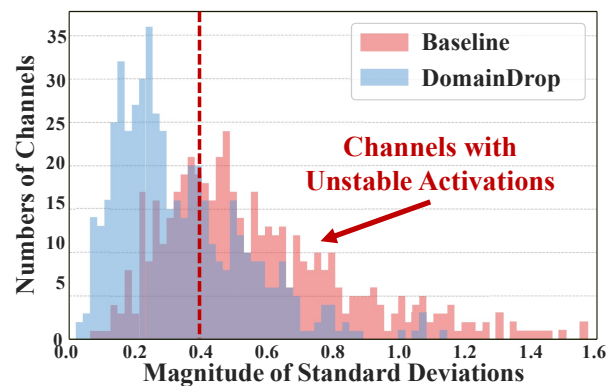


Figure 1. The robustness of channel to domain shifts. We investigate channel robustness using the histogram of activations based on their standard deviation across different domains. We experiment on PACS [29] with sketch as the target domain and analyze the representations from the last residual block of ResNet-18. For each channel, averaged activations are computed across all samples from each domain, and the standard deviation is calculated on domain dimension to indicate its robustness to domain shifts.

greatly impaired the applications of DNNs [32, 61], as training and test data often come from different distributions in reality. To address this issue, domain adaptation (DA) has been widely studied under the assumption that some labeled or unlabeled target domain data can be observed during training [13, 58]. Despite their success, DA models cannot guarantee their performance on unknown target domains that have not been seen during training, which makes them unsuitable for some real-world scenarios where target data are not always available [57, 47]. Therefore, domain generalization (DG) is proposed as a more challenging yet practical setting, which aims to utilize multiple different but related source domains to train a model that is expected to generalize well on arbitrary unseen target domains [71, 55].

The core idea of existing DG methods is to learn domain-invariant feature distribution $P(F(X))$ across domains for the robustness of conditional distribution $P(Y|F(X))$,

where $F(X)$ denotes the extracted features from input X , and Y is the corresponding label. Traditional DG methods primarily impose constraints on the whole network (*i.e.*, the prediction layer) to supervise the model to learn domain-invariant features [15, 74, 27, 43]. However, these methods do not explicitly guide the model to remove domain-specific features in middle network layers, which could lead to the model learning excessive domain-related information. In this paper, we revisit DG issue from a novel perspective of feature channels, which indicates that model generalization could be related to the robustness of feature channels to domain shifts. Specifically, we quantify the robustness of each channel to domain shifts by computing the standard deviation of activations across different domains. As shown in Fig. 1, we observe that models trained on source domains often contain numerous non-robust channels that exhibit unstable activations for different domains, indicating that they are likely to capture domain-specific features. When domain shifts, these unstable channels are likely to produce abnormal activations on the unseen target domain, leading to a shift in the conditional distribution. These unstable channels are dubbed as “domain-sensitive channels”.

Based on the above observation, we propose *Domain-Drop*, a simple yet effective framework that explicitly mutes unstable channels across domains. Unlike previous DG methods that seek to directly distill domain-invariant features, our method aims to continuously guide the model to remove domain-specific features during forward propagation. To this end, we introduce a domain discriminator to assign each channel a specific dropout rate according to its effectiveness for the domain discrimination task. The more the channel contributes to domain prediction, the more likely it contains domain-specific features, and the greater the probability of it being discarded. Moreover, we discover that unstable channels exist at both shallow and deep network layers, which contrasts with existing dropout methods that drop either high-level or low-level features. Thus, we propose a *layer-wise training strategy* that inserts Domain-Drop at a random middle layer of the network at each iteration, which can sufficiently narrow domain gaps in multiple network layers. To further enhance the robustness of channels against domain shifts, we adopt a *dual consistency loss* to regularize the model outputs under various perturbations of DomainDrop. Furthermore, we provide theoretical evidence that our method could effectively lower the generalization error bound of the model on unseen target domains.

Our contributions can be summarized as follows:

- We propose a novel dropout-based framework for DG, which explicitly suppresses domain-sensitive channels to enhance the robustness of channels to domain shifts.
- We theoretically prove that removing domain-sensitive channels during training could result in a tighter gen-

eralization error bound and better generalizability.

- We evaluate our method on four standard datasets. The results demonstrate that our framework achieves state-of-the-art performance on all benchmarks.

2. Related Works

Domain generalization. Domain generalization (DG) aims to extract knowledge from source domains that is well-generalizable to unseen target domains. One prevalent approach is to align the distributions of source domains by *learning domain-invariant representations* via adversarial learning [15, 62, 74], causality learning [49, 31] or meta-learning methods [11, 67, 58]. Another important method is *domain augmentation*, which empowers the model with generalization ability by enriching the diversity of source data at image-level [72, 47, 8] or feature-level [56, 30]. Although demonstrating promising results, these methods may still learn excessive domain-specific features, as they rely on the hope that domain-specific features would be implicitly removed by achieving the final goal of learning domain-invariant features via image-level augmentations or model-level constraints. Recently, some works reveal that CNNs tend to classify objects based on features from superficial local textures that are likely to contain domain-specific features [53, 54]. They propose to penalize the model from learning local representation and make the CNNs rely on global representations for classification [53, 54, 46]. However, the local features may be only one kind of domain-specific feature, and there could exist other forms of domain-specific features that lead to the overfitting issue. Different from these methods, our framework is proposed to continuously drop domain-sensitive channels during forward propagation, which could explicitly suppress the model learning of generic domain-specific features.

Dropout regularization. Since our method builds on discarding domain-specific features during training, we here compare our method with dropout-based methods. As one of the most widely used regularization methods, dropout [48] aims to fight the overfitting issue by randomly discarding neurons. SpatialDropout [50] is proposed to randomly drop features across channels that capture different patterns. DropBlock [17] is designed to mask random contiguous regions within a feature map. Recently, a series of structure-information dropout methods have also emerged, which utilize feature-level structure information to guide dropout operations [22, 64, 19]. DAT [28] uses adversarial dropout based on cluster assumption to help the model learn discriminative features. RSC [23] attempts to mute the most predictive parts of feature maps for learning comprehensive information. PLACE [19] seeks to activate diverse channels by randomly dropping feature channels for mitigating model overfitting. However, these methods may not be ef-

fective in the DG task where there exists a large distribution gap between source and unseen target domains [10, 12]. Due to the lack of guidance on suppressing domain-specific information, these methods still inevitably learn excessive domain-specific features and suffer from the overfitting issue on source domains. As a test-time adaptation method, SWR [7] is proposed to enable rapid adaptation to target domains during test, which reduces the update of shift-agnostic parameters identified by specific transformations, but could not address other sensitive parameters unaffected by the transformations. To address these problems, we explore a novel domain discriminator guided dropout for DG, which explicitly identifies and discards unstable channels that are non-robust to domain shifts using domain discriminators. The proposed method works only during training, relying on no target data or specific transformations. Besides, different from the most related works that solely employ dropout on either low-level [12, 46] or high-level features [23, 31], our method involves muting domain-specific information across all network layers, which effectively combats the overfitting issue of the model.

3. Methodology

3.1. Setting and Overview

Assuming that we are given K observed source domains $\mathcal{D}_s = \{D_s^1, D_s^2, \dots, D_s^K\}$ that follow different distributions. For each domain, $D_s^k = \{(x_i^k, y_i^k)\}_{i=1}^{n_k}$, where n_k is the number of samples in D_s^k , and (x_i^k, y_i^k) denotes the sample-label pair for the i -th sample in the k -th domain. The goal of domain generalization is to use the source domains \mathcal{D}_s to train a model F that is expected to perform well on unseen target domain \mathcal{D}_t . In general, the model F can be divided into a feature extractor F_f and a classifier F_c , respectively.

With training data of source domains, our framework trains the model with the standard classification loss. However, unlike previous DG methods, we aim to explicitly remove domain-specific features while enhancing domain-invariant ones. To this end, we propose a novel technique, called DomainDrop, which uses a domain discriminator to distinguish and remove domain-sensitive channels. Besides, we observe that domain-sensitive channels exist in each network layer, thus designing a layer-wise training scheme that applies DomainDrop to both high-level and low-level layers. Moreover, we add a dual consistency loss to reach consensus between predictions derived by the model under different perturbations of DomainDrop, which could further reduce channel instability to domain shifts and enhance the model learning of domain-invariant features. The overview of our framework is illustrated in Fig. 2. Below we introduce the main components of our framework and provide a theoretical analysis of its effectiveness.

3.2. Suppressing Domain-Sensitive Channels

To clearly inform the model to remove domain-specific features during training, we introduce domain discriminators to multiple middle layers for locating domain-sensitive channels, which consists of a Global Average Pooling (GAP) layer and a Fully-Connected (FC) layer. Given an input x_i^k from source domain D_k and its label y_i^k , we first extract the feature $F_l(x_i^k) \in \mathbb{R}^{C \times H \times W}$ that is yielded by the l -th middle layer, where C is the number of channels, H and W denote the height and width dimensions, respectively. The feature map $F_l(x_i^k)$ is fed to domain discriminator F_d^l to predict domain labels and compute discrimination loss. To avoid the negative impact of domain discriminators on the main network, we use a gradient reversal layer (GRL) [33] before the domain discriminator to truncate the gradients of minimizing discrimination loss:

$$\mathcal{L}_{domain}^l = -\frac{1}{K} \sum_{k=1}^K \left[\frac{1}{n_k} \sum_{i=1}^{n_k} \sum_{j=1}^K \mathbb{1}_{[j=k]} \log F_d(F_l(x_i^k)) \right]. \quad (1)$$

Distinguish domain-sensitive channels. To determine which channels contain domain-specific information, we use the performance of the domain discriminator in the middle layer as an indicator of channel importance. Specifically, we hypothesize that *channels that contribute the most to domain prediction are likely to contain domain-specific information*. We quantify the correlation between each channel and domain-specific information by computing the weighted activations for the correct domain prediction. For an input x_i^k and the feature extractor $F_l(\cdot)$, we define the score of the j -th channel in the feature map $F_l(x_i^k)$ as:

$$s_j = W_j^{y^d} \cdot \text{GAP}(F_l(x_i^k))_j, \quad (2)$$

where $W_j^{y^d} \in \mathbb{R}^C$ is the FC layer weight of the domain discriminator $F_d(\cdot)$ for the true domain y^d , and C is the channel number. The higher the weighted activation value, the more contribution of the channel to the true domain prediction. Thus, we aim to reduce the impact of domain-sensitive channels on the classification task by constraining domain-specific information used by the domain discriminator.

Dropping domain-sensitive channels. To explicitly reduce domain-specific information in the feature maps, we propose to select and drop the most domain-sensitive channels during training. Specifically, with score s_j , we first calculate the probability p_j of discarding the j -th channel:

$$p_j = s_j / \sum_{c=1}^C s_c. \quad (3)$$

Subsequently, we generate a binary mask $m \in \mathbb{R}^C$ based on the probability of each channel, *i.e.*, m_j has the probability p_j of being set to 1, with channels with relatively high

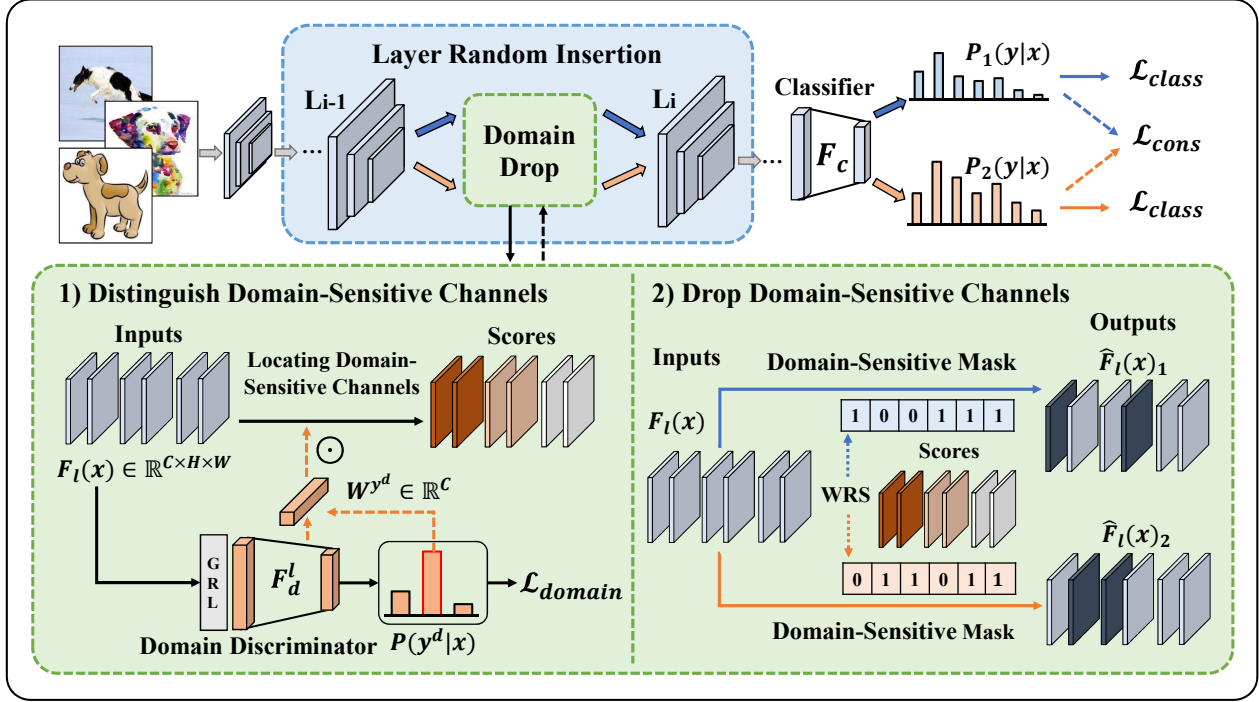


Figure 2. An overview of the proposed framework. Our framework contains three key components, including the DomainDrop, the layer-wise training scheme, and the dual consistency loss. At each iteration, we randomly select one middle layer to apply DomainDrop, which uses a domain discriminator to locate and drop domain-sensitive channels. To further enhance channel stability to domain shifts, we utilize the dual consistency loss that aligns the model predictions for the same sample under different perturbations generated by DomainDrop.

s_j more likely to be discarded. To ensure the regularization effect of dropout, we attempt to drop a certain number of domain-specific channels by probability p_j . The naive algorithm to achieve this has been analyzed in [14], but it suffers a high time complexity of $O(C \times M)$, where C is the number of channels and M is the number of discarded channels. To reduce computation cost, we employ weighted random selection (WRS) algorithm [14, 22] to generate the binary mask m , which enjoys the time complexity as $O(C)$. Specifically, for the j -th channel with s_j , we first generate a random number $r_j \in (0, 1)$ and compute a key value $k_j = r_j^{1/s_j}$. Then we select M items with the largest key values and set the corresponding m_j of the mask m to 0:

$$m_j = \begin{cases} 0, & \text{if } j \in \text{TOP}(\{k_1, k_2, \dots, k_C\}, M) \\ 1, & \text{otherwise} \end{cases}, \quad (4)$$

where j is the channel index, $\text{TOP}(\{k_1, k_2, \dots, k_C\}, M)$ denotes the M items with the largest key value k . Additionally, the hyper-parameter $P_{drop} = \frac{M}{C}$ indicates the number of channels to discard. In practice, we use a hyperparameter P_{active} to control the activation probability of DomainDrop in the forward pass, which can narrow the domain gap by preserving the original feature maps while meantime reducing domain-specific information. During inference, DomainDrop is closed as conventional dropout [48].

Remark. Note that DomainDrop differs significantly from previous DG methods that focus on constraining the network to extract domain-invariant information during the backpropagation stage. In contrast, our DomainDrop operates during the feature forward propagation stage, which continuously filters out domain-sensitive channels to prevent the model from retaining too many domain-related features. The process could also be regarded as a guided feature-level augmentation that effectively disturbs domain-specific features while preserving domain-invariant features. Compared to previous dropout methods, we address the DG issue from a novel perspective, considering enhancing channel robustness to domain shifts. By dropping generic domain-sensitive channels during training, our method reduces channel instability caused by domain shifts and promotes the learning of domain-invariant features.

3.3. Layer-wise Training Scheme

Prior research has revealed that CNNs encode information across multiple stages, with shallow layers capturing more generic patterns and deep ones encoding more semantic features [63, 66]. Traditional dropout techniques focus on removing information from high-level semantic layers that contain crucial classification information [28, 23, 25, 64, 31]. Recently, some studies suggest that features ex-

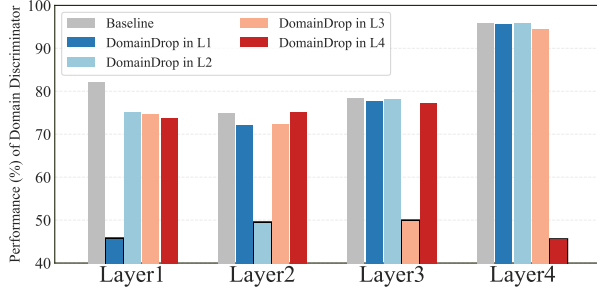


Figure 3. The accuracy of domain discriminator in each network layer. We test the discrimination accuracies of the baseline and the models with DomainDrop in different layers. The experiment is conducted on the PACS dataset with ResNet-18 backbone.

tracted from shallow layers exhibit more distinct domain-specific characteristics, and thus propose gating low-level information to mitigate domain gaps [46, 12]. However, previous methods are typically designed for a specific single layer, either low-level or high-level, which may not be optimal for DG task. In contrast, we propose gating feature maps from all network layers, recognizing that features extracted from each layer may contain domain-specific characteristics. To validate the statement, an experiment is conducted on the PACS dataset, in which we insert multiple domain discriminators with GRL (*i.e.*, truncate gradients) in each layer and train them with the backbone network. The accuracy of each discriminator indicates how many domain-specific features are contained in the corresponding layer.

As shown in Fig. 3, there are several observations: (1) The domain discriminator in each layer has relatively good performance, indicating that considerable domain gaps exist in every network layer. (2) The discrimination accuracy drops while inserting DomainDrop into the corresponding layer, suggesting that DomainDrop can effectively narrow the domain gap in the inserted layer. Hence, the proposed DomainDrop focuses on removing domain-specific features in both high-level and low-level layers. Since discarding features at multiple layers simultaneously may lead to excessive information absence and hinder model convergence [40], we propose a layer-wise training scheme that randomly selects a middle layer of the network and performs DomainDrop on its feature maps at each iteration. This scheme enables DomainDrop to effectively reduce channel sensitivity to domain shifts in multiple network layers.

3.4. Enhancing Domain-Invariant Channels

In the training stage, DomainDrop is employed to remove domain-sensitive channels by probability, which can be approximated as applying Gaussian multiplicative noise to domain-specific features [48, 40]. In light of the output inconsistency that arises from different perturbations of DomainDrop, we propose a dual consistency loss that enhances

model robustness to domain-specific feature perturbations and facilitates the learning of domain-invariant features.

Specifically, for a given input data x_i^k , denoted as x for simplicity, we apply DomainDrop twice at each training step to obtain two sets of class predictions denoted as $\hat{F}(x)_1$ and $\hat{F}(x)_2$, respectively. According to Eqs. (3) and (4), the masks m generated by DomainDrop could be different for the same input, leading to different predictions $\hat{F}(x)_1$ and $\hat{F}(x)_2$. To ensure consistency between these two outputs for the same input, we introduce the following constraint:

$$\mathcal{L}_{cons} = \frac{1}{2} (\text{KL}[\sigma(\hat{F}(x)_1/T) || \sigma(\hat{F}(x)_2/T)] + \text{KL}[\sigma(\hat{F}(x)_2/T) || \sigma(\hat{F}(x)_1/T)]), \quad (5)$$

where σ denotes a softened softmax at a temperature T , and KL is Kullback-Leibler divergence [21]. With this consistency constrain, our framework encourages the model to improve the robustness of channels to domain shifts and extract domain-invariant features from perturbed representations. Moreover, Since DomainDrop only operates during training, the loss could also reduce the inconsistency existing in the training and inference stages [60], thus improving the generalization ability of the model to target domains.

3.5. Theoretical Analysis of DomainDrop

Previous DG methods mainly train the model to directly distill domain-invariant features, which could still retain excessive domain-specific features and perform inferiorly when domain shifts. We here theoretically prove that *removing the domain-sensitive channels during training can improve the generalizability in DG task*. Given a hypothesis $h : \mathcal{X} \rightarrow \mathcal{Y}$, where h comes from the space of the candidate hypothesis \mathcal{H} , \mathcal{X} and \mathcal{Y} are the input space and the label space, respectively. Let $\phi(\cdot) : \mathcal{X} \rightarrow \mathbb{R}^n$ be the feature extractor that maps the input images into the n -dimensional feature space. Following the popular Integral Probability Metrics (IPMs) [38, 16, 65], we first define the channel-level maximum mean discrepancy (CMMD) that estimates the distribution gap between different domains by channels.

Definition 1. Let n denote the number of channels in the extracted features of $\phi(\cdot)$. Given two different distribution of D_s and D_t , the **channel-level maximum mean discrepancy (CMMD)** between $\phi(D_s)$ and $\phi(D_t)$ is defined as:

$$d_{\text{CMMD}}(D_s, D_t) = \frac{1}{n} \sum_{i=1}^n \sup_{\phi_i \in \Phi_i} \left\| \int_{\mathcal{X}} k(x, \cdot) d\phi_i(D_s) - \phi_i(D_t) \right\|_{\mathcal{H}_k}, \quad (6)$$

where Φ is the space of candidate hypothesis for each channel, $\phi_i(D)$ is the distribution of the i -th channel for the domain D , and \mathcal{H}_k is a RKHS with its associated kernel k .

Based on the above definition, we derive the theorem that provides an upper bound on the generalization risk of the hypothesis h on the target domain D_t . First, we define the risk of h on a domain D as: $\mathcal{R}[h] = \mathbb{E}_{x \sim D} \ell[h(x), f(x)]$, where $\ell_{h,f} : x \rightarrow \ell[h(x), f(x)]$ is a convex loss-function defined for $\forall h, f \in \mathcal{H}$, and assume that ℓ obeys the triangle inequality. Following [10, 70], for the source domains $\mathcal{D}_s = \{D_s^1, D_s^2, \dots, D_s^K\}$, we define the convex hull Λ_s as a set of mixture of source domain distributions: $\Lambda_s = \{\bar{D} : \bar{D}(\cdot) = \sum_{i=1}^K \pi_i D_s^i(\cdot), \pi_i \in \Delta_K\}$, where π is non-negative coefficient in the K -dimensional simplex Δ_K . We define $\bar{D}_t \in \Lambda_s$ as the closest domain to the target domain D_t .

Theorem 1 (Generalization risk bound). *With the previous settings and assumptions, let S^i and T be two samples of size m drawn i.i.d from D_s^i and D_t , respectively. Then, with the probability of at least $1 - \delta$ ($\delta \in (0, 1)$) for all $h \in \mathcal{F}$, the following inequality holds for the risk $\mathcal{R}_t[h]$:*

$$\begin{aligned} \mathcal{R}_t[h] &\leq \sum_{i=1}^N \pi_i \mathcal{R}_s^i[h] + d_{\text{CMMD}}(\bar{D}_t, D_t) \\ &\quad + \sup_{i,j \in [K]} d_{\text{CMMD}}(D_s^i, D_s^j) + \lambda + \epsilon, \end{aligned} \quad (7)$$

where $\lambda = 2\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} + \frac{2}{m} (\sum_{i=1}^N \pi_i E_{x \sim D_s^i} [\sqrt{\text{tr}(K_{D_s^i})}] + E_{x \sim D_t} [\sqrt{\text{tr}(K_{D_t})}])$, $K_{D_s^i}$ and K_{D_t} are kernel functions computed on samples from D_s^i and D_t , respectively. $\text{tr}(\cdot)$ is the trace of input matrix and ϵ is the combined error of ideal hypothesis h^* on D_t and \bar{D}_t . Let $\gamma = d_{\text{CMMD}}(\bar{D}_t, D_t)$ and $\beta = \sup_{i,j \in [K]} d_{\text{CMMD}}(D_s^i, D_s^j)$, respectively.

Proof. See in Supplementary Material. \blacksquare

Theorem 1 indicates that the upper bound of generalization risk on target domain depends mainly on: 1) γ that measures the maximum discrepancy between different activations for the same channel in source and target domains; 2) β that presents the maximum pairwise activation discrepancy among source domains at channel level. Recall that our DomainDrop actively removes domain-sensitive channels during training, which can effectively restrict the size of the hypothesis space Φ and promote the model learning of domain-invariant channels. Consequently, the features extracted by the DomainDrop model from source domains would exhibit a smaller distribution gap than the original model, *i.e.*, effectively reducing β in Eq. (7). Moreover, after removing domain-sensitive channels, the features extracted from target domain would become more similar to those of source domains, thus potentially reducing γ in Eq. (7). Concerning Theorem 1, *by explicitly removing domain-sensitive channels, we can obtain a lower error bound and expect a better generalization ability.* The conclusion is also proved in the experiment section (Sec. 4.4).

4. Experiment

4.1. Datasets

We evaluate our method on four conventional DG benchmarks: (1) **PACS** [29] consists of images from 4 domains: Photo, Art Painting, Cartoon, and Sketch, including 7 object categories and 9,991 images total. We adopt the official split provided by [29] for training and validation. (2) **Office-Home** [52] contains around 15,500 images of 65 categories from 4 domains: Artistic, Clipart, Product and Real-World. As in [2], we randomly split each domain into 90% for training and 10% for validation. (3) **VLCS** [51] comprises of 5 categories selected from 4 domains, VOC 2007 (Pascal), LabelMe, Caltech and Sun. We use the same setup as [2] and divide the dataset into training and validation sets based on 7 : 3. (4) **DomainNet** [42] is a large-scale dataset, consisting of about 586,575 images with 345 categories from 6 domains, *i.e.*, Clipart, Infograph, Painting, Quickdraw, Real, and Sketch. Following [18], we split source data into 80% for training and 20% for validation.

4.2. Implementation Details

Basic details. For PACS and OfficeHome, we use the ImageNet pre-trained ResNet-18 and ResNet-50 as our backbones following [67, 34]. For VLCS, we follow [68, 67] and use the ResNet-18 as backbone. The batch size is 128. We train the network using SGD with momentum of 0.9 and weight decay of 0.0005 for 50 epochs. The initial learning rate is 0.002 and decayed by 0.1 at 80% of the total epochs. For the large-scale DomainNet, we use ResNet-50 pre-trained on ImageNet as backbone and train the network using Adam optimizer for 5000 iterations following [4, 18]. The initial learning rate is $5e - 5$ and the batch size is 64.

Method-specific details. For all experiments, we the dropout ratio P_{drop} of DomainDrop to 0.33. We set the weight of gradient reversal layers [33] before domain discriminators to 0.25. The weight of the dual consistency loss is set to 1.5 and the temperature T in Eq. (5) is set to 5 for all datasets. For layer-wise training scheme, we select all the middle layers of the network (*i.e.*, the 1st, 2nd, 3rd, and 4-th residual blocks) as the candidate set. At each iteration, we randomly select a residual block from the set and perform DomainDrop on its feature maps. We apply the leave-one-domain-out protocol for all benchmarks. We select the best model on the validation splits of all source domains and report the top-1 accuracy. All results are reported based on the averaged accuracy over five repetitive runs.

4.3. Comparison with SOTA Methods

Evaluation on PACS. We evaluate our framework on PACS with ResNet-18 and ResNet-50 as our backbones. We compare with several dropout-based methods (*i.e.*, I²-drop [46], RSC [23], PLACE [19], CDG [12]), augmenta-

Table 1. Performance (%) comparisons with the start-of-the-art DG approaches on the PACS dataset with ResNet-18 and ResNet-50 backbones. The best performance is marked as **bold**.

Methods	Art	Cartoon	Photo	Sketch	Avg.
ResNet-18					
DeepAll [72] (AAAI'20)	80.31	76.65	95.38	71.67	81.00
I ² -Drop [46] (ICML'20)	80.27	76.54	96.11	76.38	82.33
DMG [5] (ECCV'20)	80.38	76.90	93.35	75.21	81.46
RSC [23] (ECCV'20)	83.43	80.31	95.99	80.85	85.15
NAS-OoD [1] (ICCV'21)	83.74	79.69	96.23	77.27	84.23
MixStyle [73] (ICLR'21)	84.10	78.80	96.10	75.90	83.70
LDSGD [57] (ICCV'21)	81.44	79.56	95.51	80.58	84.27
PLACE [19] (arXiv'21)	82.60	78.33	95.65	81.47	84.51
FACT [61] (CVPR'21)	85.37	78.38	95.15	79.15	84.51
StableNet [68] (CVPR'21)	81.74	79.91	96.53	80.50	84.69
EFDMix [69] (CVPR'22)	83.90	79.40	96.80	75.00	83.90
StyleNeophile [24] (CVPR'22)	84.41	79.25	94.93	83.27	85.47
COMEN [6] (CVPR'22)	82.60	81.00	94.60	84.50	85.70
I ² -ADR [34] (ECCV'22)	82.90	80.80	95.00	83.50	85.60
CDG [12] (IJCV'22)	83.50	80.10	95.60	83.80	85.80
ALOFT [20] (CVPR'23)	84.81	79.05	96.11	80.55	85.13
DomainDrop (Ours)	84.47 ± 0.77	80.50 ± 0.56	96.83 ± 0.21	84.83 ± 0.67	86.66
ResNet-50					
DeepAll [72] (AAAI'20)	81.31	78.54	94.97	69.76	81.15
RSC [23] (ECCV'20)	87.89	82.16	97.92	83.85	87.83
FACT [61] (CVPR'21)	89.63	81.77	96.75	84.46	88.15
PLACE [19] (arXiv'21)	87.55	83.11	97.19	83.48	87.83
COPA [59] (ICCV'21)	83.30	79.80	94.60	82.50	85.10
SWAD [3] (NeurIPS'21)	89.30	83.40	97.30	82.50	88.10
EFDMix [69] (CVPR'22)	90.60	82.50	98.10	76.40	86.90
StyleNeophile [24] (CVPR'22)	90.35	84.20	96.73	85.18	89.11
I ² -ADR [34] (ECCV'22)	88.50	83.20	95.20	85.80	88.20
PTE [35] (ECCV'22)	87.90	78.40	98.20	75.70	85.10
CDG [12] (IJCV'22)	88.90	83.50	97.60	84.90	88.70
ALOFT [20] (CVPR'23)	89.26	83.11	97.96	84.04	88.59
DomainDrop (Ours)	89.82 ± 0.44	84.22 ± 0.37	98.02 ± 0.24	85.98 ± 1.14	89.51

tion based methods (*i.e.*, MixStyle [73], LDSGD [57], EFD-Mix [69], StyleNeophile [24], ALOFT [20]), feature decorrelation methods (*i.e.*, StableNet [68], I²-ADR [34]), meta-learning method (*i.e.*, COMEN [6]) and neural search (*i.e.*, NAS-OoD [1]). As presented in Tab. 6, our framework obtains the highest average accuracy among all the compared methods on both backbones. Specifically, compared with existing dropout-based DG methods, our DomainDrop can surpass the SOTA approach CDG by a considerable margin by 0.86% (86.66% vs. 85.80%) on ResNet-18 and 0.81% (89.51% vs. 88.70%) on ResNet-50. It is because DomainDrop can explicitly reduce the domain-specific features in every middle layer of the network, instead of only applying dropout in a single layer and narrowing the domain gap implicitly. Besides, our method achieves the best performance on most domains and our overall performance exceeds other SOTA DG methods. The encouraging results demonstrate the superiority of our method in fighting the overfitting issue of the model on source domains.

Evaluation on OfficeHome. We also compare our method with SOTA DG methods on OfficeHome to demonstrate the adaptation of our method to the dataset with a large number of categories and samples. We perform the evaluation on both ResNet-18 and ResNet-50 backbones. The results are reported in Tab. 2. Our method can still achieve the best performance among the compared

Table 2. Performance (%) comparisons with the SOTA DG methods on the OfficeHome dataset with ResNet-18 and ResNet-50 backbones. The best performance is marked as **bold**.

Methods	Art	Clipart	Product	Real	Avg.
ResNet-18					
DeepAll [72] (AAAI'20)	56.98	50.14	72.85	74.39	63.59
RSC [23] (ECCV'20)	57.70	48.58	72.59	74.17	63.26
MixStyle [73] (ICLR'21)	58.70	53.40	74.20	75.90	65.50
SagNet [37] (CVPR'21)	60.20	45.38	70.42	73.38	62.34
COPA [59] (ICCV'21)	59.40	55.10	74.80	75.00	66.10
FACT [61] (CVPR'21)	60.34	54.85	74.48	76.55	66.56
StyleNeophile [24] (CVPR'22)	59.55	55.01	73.57	75.52	65.89
CDG [12] (IJCV'22)	59.20	54.30	74.90	75.70	66.00
DomainDrop (Ours)	59.62 ± 0.40	55.60 ± 0.31	74.50 ± 0.53	76.64 ± 0.35	66.59
ResNet-50					
DeepAll [72] (AAAI'20)	61.30	52.40	75.80	76.60	66.50
RSC [23] (ECCV'20)	57.70	51.40	74.80	75.10	65.50
SelfReg [26] (ICCV'21)	63.60	53.10	76.90	78.10	67.90
SagNet [37] (CVPR'21)	63.40	54.80	75.80	78.30	68.10
SWAD [3] (NeurIPS'21)	66.10	57.70	78.40	80.20	70.60
Fishr [44] (ICML'22)	63.40	54.20	76.40	78.50	68.20
PTE [35] (ECCV'22)	66.30	55.80	78.20	80.40	70.10
DomainDrop (Ours)	67.33 ± 0.45	60.39 ± 0.48	79.05 ± 0.29	80.22 ± 0.22	71.75

Table 3. Performance (%) comparisons with the SOTA DG methods on VLCS with ResNet-18 backbone. The best is **bolded**.

Methods	Caltech	LabelMe	Pascal	Sun	Avg.
DeepAll [72] (AAAI'20)	96.98	62.00	73.83	68.66	75.37
JiGen [2] (CVPR'19)	96.17	62.06	70.93	71.40	75.14
MMLD [33] (AAAI'20)	97.01	62.20	73.01	72.49	76.18
RSC [23] (ECCV'20)	95.83	63.74	71.86	72.12	75.89
StableNet [68] (CVPR'21)	96.67	65.36	73.59	74.97	77.65
DomainDrop (Ours)	98.94 ± 0.19	63.97 ± 1.33	76.36 ± 0.93	73.74 ± 1.17	78.25

DG methods, *e.g.*, outperforming the state-of-the-art DG method COPA [59] by 0.49% (66.59% vs 66.10%). Besides, our method precedes the best method SWAD [3] on ResNet-50, which proposes an ensemble learning method that seeks flat minima for DG, with a significant improvement of 1.14% (71.75% vs. 70.60%). The above results further demonstrate the effectiveness of our framework.

Evaluation on VLCS. To verify the trained model can also generalize to unseen target domains with a relatively small domain gap to source domains, we conduct the experiments on VLCS with ResNet-18. The results are portrayed in Tab. 3. Among all the competitors, our DomainDrop achieves the best performance, exceeding the second-best method StableNet [68] by 0.60% (78.25% vs. 77.65%) on average. Our method also exceeds the advanced dropout method RSC [23], which removes over-dominant features according to gradients, by 2.36% (78.25% vs. 75.89%). The results prove the effectiveness of our method again.

Evaluation on DomainNet. Tab. 4 shows the results on the large-scale DomainNet. On the challenging benchmark, our method performs better in the averaged accuracy than existing methods and improves top-1 accuracy by 3.87% (44.37% vs. 40.50%) from ResNet-50 baseline, proving the effectiveness of our method on the large-scale dataset. All the comparisons prove that suppressing domain-sensitive channels can effectively improve model generalizability.

Table 4. Performance (%) comparisons with the SOTA DG methods on DomainNet with ResNet-50 backbone. The best is **bolded**.

Methods	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	Avg.
DeepAll [72] (AAAF'20)	62.20	19.90	45.50	13.80	57.50	44.40	40.50
RSC [23] (ECCV'20)	55.00	18.30	44.40	12.20	55.70	47.80	38.90
SagNet [37] (CVPR'21)	57.70	19.00	45.30	12.70	58.10	48.80	40.30
SelfReg [26] (ICCV'21)	60.70	21.60	49.40	12.70	60.70	51.70	42.80
I ² -ADR [34] (ECCV'22)	64.40	20.20	49.20	15.00	61.60	53.30	44.00
PTE [35] (ECCV'22)	62.40	21.00	50.50	13.80	64.60	52.40	44.10
DomainDrop (Ours)	62.93 \pm 0.25	21.63\pm0.07	50.67\pm0.15	14.83 \pm 0.30	62.70 \pm 0.08	53.46\pm0.59	44.37

Table 5. Comparisons with the SOTA dropout-based methods on PACS dataset with ResNet-18 backbone. The best is **bolded**.

Methods	Art	Cartoon	Photo	Sketch	Avg.
Baseline (AAAF'20)	80.31	76.65	95.38	71.67	81.00
Cutout [9] (ArXiv'17)	79.60	77.20	95.90	71.60	80.60
C-Drop [36] (ICCV'17)	79.64	76.49	95.93	72.37	81.11
DropBlock [17] (NeurIPS'18)	80.30	77.50	95.60	76.40	82.50
AdvDrop [41] (AAAF'18)	82.40	78.20	96.10	75.90	83.00
WCD [22] (AAAF'19)	81.56	78.24	94.99	75.53	82.58
DMG [5] (ECCV'20)	80.38	76.90	93.35	75.21	81.46
I ² -Drop [46] (ICML'20)	80.27	76.54	96.11	76.38	82.33
RSC [23] (ECCV'20)	83.43	80.31	95.99	80.85	85.15
PLACE [19] (arXiv'21)	82.60	78.33	95.65	81.47	84.51
CDG [12] (ICV'22)	83.50	80.10	95.60	83.80	85.80
DomainDrop (Ours)	84.47\pm0.77	80.50\pm0.56	96.83\pm0.21	84.83\pm0.67	86.66

Comparison with SOTA dropout methods. We here compare DomainDrop with the SOTA dropout-based methods, including the traditional approaches designed for supervised learning (*i.e.*, Cutout [9], C-Drop [36], DropBlock [17], AdvDrop [41], WCD [22]) and the methods proposed for domain generalization (*i.e.*, DMG [5], I²-Drop [46], RSC [23], PLACE [19], CDG [12]). The results are summarized in Tab. 5. We observe that the traditional dropout methods cannot adequately combat the overfitting issue of the model on source domains, which is caused by the large discrepancy between source domains and unseen target domains. Besides, DMG [5] that assumes some domain-specific features provide useful information for the target domain achieves a slight improvement, which may be because the assumption cannot always be guaranteed since the target domain is unseen. I²-Drop [46] aims to penalize the model from learning superficial local textures, but there could also exist other forms of domain-specific features. PLACE [19] attempts to activate diverse channels by randomly discarding channels, but it inevitably activates channels that capture domain-specific information, impeding the model generalization. Although RSC [23] and CDG [12] have shown promising performance, these methods are only suitable for a single network layer, which prevents them from sufficiently reducing domain-specific information. In general, our framework obtains the best performance among existing dropout methods, which proves its effectiveness in enhancing model generalization.

4.4. Analytical Experiments

In this paragraph, we conduct extensive ablation studies of our framework, including the impact of each com-

Table 6. Ablation study on different components of our framework: DomainDrop (DD), layer-wise training scheme (LT), and dual consistency loss (CL). The experiment is conducted on PACS and OfficeHome with ResNet-18. The best is **bolded**.

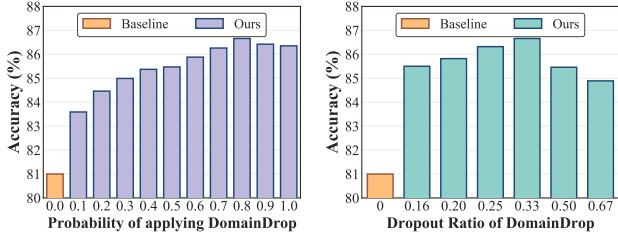
PACS								
Methods	DD	LT	CL	Art	Cartoon	Photo	Sketch	Avg.
Baseline	-	-	-	80.31 \pm 1.54	76.65 \pm 0.48	95.38 \pm 0.12	71.67 \pm 1.49	81.00
Variant 1	✓	-	-	82.37 \pm 0.78	79.27 \pm 0.26	95.45 \pm 0.27	83.15 \pm 0.73	85.06
Variant 2	✓	✓	-	83.84 \pm 0.12	79.69 \pm 0.29	96.47 \pm 0.36	83.74 \pm 0.39	85.94
Variant 3	✓	-	✓	83.32 \pm 0.76	80.09 \pm 0.91	95.85 \pm 0.33	83.71 \pm 0.61	85.74
DomainDrop	✓	✓	✓	84.47\pm0.77	80.50\pm0.56	96.83\pm0.21	84.83\pm0.67	86.66

OfficeHome								
Methods	DD	LT	CL	Art	Clipart	Product	Real	Avg.
Baseline	-	-	-	56.98 \pm 0.47	50.14 \pm 0.88	72.85 \pm 0.35	74.39 \pm 0.32	63.59
Variant 1	✓	-	-	57.49 \pm 0.37	53.69 \pm 0.58	73.82 \pm 0.14	76.18 \pm 0.20	65.30
Variant 2	✓	✓	-	59.46 \pm 0.28	53.86 \pm 0.57	74.23 \pm 0.29	76.31 \pm 0.24	65.96
Variant 3	✓	-	✓	58.17 \pm 0.29	54.96 \pm 0.40	73.95 \pm 0.35	76.42 \pm 0.49	65.88
DomainDrop	✓	✓	✓	59.62\pm0.40	55.60\pm0.31	74.50\pm0.53	76.64\pm0.35	66.59

ponent and the parameter sensitivity. We also analyze the discrepancy among source and target domains to prove that our framework can effectively reduce domain divergence. Moreover, we validate the orthogonality of our DomainDrop on other DG SOTA methods. More experiments can be found in the Supplementary Material.

Ablation study on each component. We here conduct the ablation study to investigate the efficacy of DomainDrop (DD), Layer-wise Training Scheme (LT), and Consistency Loss (CL) in our framework. Tab. 6 presents the results of different variants of DomainDrop with ResNet-18 on both PACS and OfficeHome. Notably, for Variant 1, we apply DomainDrop to all layers, activating it with a probability of 0.5 for each layer, which aligns with [73, 69]. As shown in Tab. 6, variant 1 exhibits significant improvement over the baseline, *i.e.*, 4.06% (85.06% vs. 81.00%) on PACS and 1.71% (65.30% vs. 63.59%) on OfficeHome, proving the effectiveness of DomainDrop in reducing channel sensitivity to domain changes and narrowing domain gap. Besides, comparing variant 1 with variant 2, we observe that combining both DomainDrop and layer-wise training scheme leads to better performance than using DomainDrop alone, indicating that alternating use of DomainDrop at multiple layers can maximize regularization effect while avoiding losing too much information. Furthermore, the improved performance of variant 3 over variant 1 suggests that the consistency loss contributes to enhancing domain-invariant features. Finally, DomainDrop performs the best, verifying that the three modules complement and promote mutually, and none of them is dispensable for achieving the superior generalization ability of the model.

Parameter sensitivity. We conduct experiments to investigate the sensitivity of DomainDrop to hyper-parameter P_{active} and P_{drop} as shown in Fig. 4(a) and 4(b). Specifically, P_{apply} denotes the probability of applying DomainDrop to the network at each iteration, varying from 0.1 to 1.0. The results in Fig. 4(a) show that DomainDrop achieves competitive performance robustly with a relatively



(a) Effect of dropout probability. (b) Effects of dropout ratio.

Figure 4. Effects of hyper-parameters including the probability of applying DomainDrop and the dropout ratio in DomainDrop. The experiments are conducted on PACS with ResNet-18 backbone.

Table 7. Comparisons of source and source-target domain divergences ($\times 10$) for different methods on PACS with ResNet-18.

Methods	Source Divergence	Source-Target Divergence
Baseline	9.38	7.25
MixStyle [73]	7.53	6.23
FACT [61]	9.00	7.27
I ² -Drop [34]	7.11	6.44
RSC [23]	8.81	7.20
PLACE [19]	6.45	6.17
DomainDrop (Ours)	5.15	4.09

large probability (*i.e.*, $P_{active} \geq 0.7$), verifying the stability of our method. Moreover, we also examine the impacts of the dropout ratio on model performance, which vary $\{0.16, 0.20, 0.25, 0.33, 0.50, 0.67\}$ as in [23]. Fig. 4(b) shows that our framework consistently exceeds the baseline by a large margin (*i.e.*, more than 4%) with different dropout ratios, and the highest accuracy is reached at $P_{drop} = 0.33$, which is adopted as default in all our experiments.

Discrepancy among source and target domains. To verify the effectiveness of our method to channel deviation across domains, we here measure the estimated channel-level maximum mean discrepancy (CMMD) among source and target domains. Based on the definition of CMMD in Eq. (6), given a trained feature extractor $\phi: \mathcal{X} \rightarrow \mathbb{R}^n$, we estimate the CMMD between pairwise domains D_s^i and D_s^j :

$$\hat{d}_{CMMD}(D_s^i, D_s^j) = \frac{1}{n} \sum_{k=1}^n \|\phi_k(D_s^i) - \phi_k(D_s^j)\|_2. \quad (8)$$

Then we estimate the source domain divergence β by $\hat{\beta} = \sup_{i,j \in [K]} \hat{d}_{CMMD}(D_s^i, D_s^j)$. We utilize the averaged divergence between target domain and each source domain to estimate the source-target domain gap γ , defined as $\hat{\gamma} = \frac{1}{K} \sum_{k=1}^K \hat{d}_{CMMD}(D_s^k, D_t)$. The results are shown in Tab. 7. Notably, both the SOTA DG methods, MixStyle [73] and FACT [61], present lower source domain divergence due to their ability to diversify the source data. However, they do not explicitly remove domain-specific features, leading to limited reduction of the source-target domain divergence. RSC [23] regularizes the model to learn comprehensive representations but still cannot adequately narrow the domain

Table 8. Effect (%) of DomainDrop on other SOTA DG methods. The experiments are conducted on the PACS dataset. We select ResNet-18 as the backbone architecture for RandAug [8] and FACT [61]. We also validate the effectiveness of our DomainDrop on the SOTA MLP-like model, *i.e.*, GFNet [45].

Methods	Art	Cartoon	Photo	Sketch	Avg.
RandAug [8]	82.90 \pm 0.49	76.89 \pm 0.75	96.17 \pm 0.46	78.55 \pm 0.54	83.63
+ DomainDrop	84.62\pm0.59	80.25\pm0.51	96.27\pm0.40	85.62\pm0.60	86.69
FACT [61]	85.64 \pm 0.57	77.94 \pm 0.83	95.45 \pm 0.78	79.41 \pm 1.30	84.61
+ DomainDrop	86.52\pm0.83	80.86\pm0.49	95.81\pm0.23	85.75\pm1.02	87.24
GFNet [45]	89.37 \pm 0.60	84.74 \pm 0.59	97.94 \pm 0.25	79.01 \pm 0.77	87.76
+ DomainDrop	92.38\pm0.69	87.24\pm0.71	98.26\pm0.36	86.18\pm1.02	91.02

gap across source and target domains. I²-Drop [46] penalizes superficial local features that may contain domain-specific information, but it only applies to shallow layers and cannot suppress other forms of domain-specific features. PLACE [19] generates a regularization effect to fight the overfitting, but it cannot prevent channels from learning domain-specific information. In contrast, the lowest source and source-target domain gaps achieved by DomainDrop prove its effectiveness in reducing domain-specific features, highlighting its superiority over existing methods. Moreover, the results align well with the theoretical analysis in Sec. 3.5 and prove that DomainDrop effectively lowers the generalization risk bound by reducing β and γ .

Orthogonality to other DG methods. We investigate the effectiveness of DomainDrop in improving the performance of other DG methods. We selected two representative DG methods, *i.e.*, RandAug [8] and FACT [61], both using ResNet-18 as the backbone. We also verify the effectiveness of DomainDrop on the SOTA MLP-like model GFNet [45]. As shown in Tab. 8, our method can significantly improve the SOTA DG methods, *e.g.*, boosting RandAug by 3.06% (87.24% vs. 84.18%) and FACT by 2.63% (87.24% vs. 84.61%). These experiments demonstrate that our DomainDrop is orthogonal to other SOTA DG methods, and a new SOTA performance can be achieved by combining our approach with existing methods. Moreover, our DomainDrop also achieves a considerable improvement on the MLP-like model, enhancing the performance of GFNet by 3.26% (91.02% vs. 87.76%), which indicates the generalization of our methods on different network architectures.

5. Conclusion

In this paper, we study the model generalizability from a novel channel-level perspective and find that the overfitting issue could be caused by substantial domain-sensitive channels in the model. To tackle this issue, we propose a novel DG framework that explicitly suppresses domain-specific features by removing the domain-sensitive channels. We also theoretically prove the effectiveness of our framework to generate a tight generalization error bound. Experiments show that our framework achieves strong performance on various datasets compared with existing DG methods.

References

- [1] Haoyue Bai, Fengwei Zhou, Lanqing Hong, Nanyang Ye, S-H Gary Chan, and Zhenguo Li. Nas-ood: Neural architecture search for out-of-distribution generalization. In *ICCV*, 2021. 7
- [2] Fabio M Carlucci, Antonio D’Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, 2019. 6, 7
- [3] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *NeurIPS*, 2021. 7
- [4] Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. Domain generalization by mutual-information regularization with pre-trained models. In *ECCV*, 2022. 6
- [5] Prithvijit Chattopadhyay, Yogesh Balaji, and Judy Hoffman. Learning to balance specificity and invariance for in and out of domain generalization. In *ECCV*, 2020. 7, 8
- [6] Chaoqi Chen, Jiongcheng Li, Xiaoguang Han, Xiaoqing Liu, and Yizhou Yu. Compound domain generalization via meta-knowledge encoding. In *CVPR*, 2022. 7
- [7] Sungha Choi, Seunghan Yang, Seokeon Choi, and Sungrack Yun. Improving test-time adaptation via shift-agnostic weight regularization and nearest source prototypes. In *ECCV*, 2022. 3
- [8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020. 2, 9
- [9] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 8
- [10] Yu Ding, Lei Wang, Bin Liang, Shuming Liang, Yang Wang, and Fang Chen. Domain generalization by learning and removing domain-specific features. In *NeurIPS*, 2022. 3, 6
- [11] Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *NeurIPS*, 2019. 2
- [12] Dapeng Du, Jiawei Chen, Yuexiang Li, Kai Ma, Gangshan Wu, Yefeng Zheng, and Limin Wang. Cross-domain gated learning for domain generalization. *IJCV*, 2022. 3, 5, 6, 7, 8
- [13] Zhekai Du, Jingjing Li, Hongzu Su, Lei Zhu, and Ke Lu. Cross-domain gradient discrepancy minimization for unsupervised domain adaptation. In *CVPR*, 2021. 1
- [14] Pavlos S Efraimidis and Paul G Spirakis. Weighted random sampling with a reservoir. *Information processing letters*, 2006. 4
- [15] Xinjie Fan, Qifei Wang, Junjie Ke, Feng Yang, Boqing Gong, and Mingyuan Zhou. Adversarially adaptive normalization for single domain generalization. In *CVPR*, 2021. 2
- [16] Bo Geng, Dacheng Tao, and Chao Xu. Daml: Domain adaptation metric learning. *TIP*, 2011. 5
- [17] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *NeurIPS*, 2018. 2, 8
- [18] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2020. 6
- [19] Jintao Guo, Lei Qi, Yinghuan Shi, and Yang Gao. Domain generalization via progressive layer-wise and channel-wise dropout. *arXiv preprint arXiv:2112.03676*, 2021. 2, 6, 7, 8, 9
- [20] Jintao Guo, Na Wang, Lei Qi, and Yinghuan Shi. Aloft: A lightweight mlp-like architecture with dynamic low-frequency transform for domain generalization. In *CVPR*, 2023. 7
- [21] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 5
- [22] Saihui Hou and Zilei Wang. Weighted channel dropout for regularization of deep convolutional neural network. In *AAAI*, 2019. 2, 4, 8
- [23] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *ECCV*, 2020. 2, 3, 4, 6, 7, 8, 9
- [24] Juwon Kang, Sohyun Lee, Namyup Kim, and Suha Kwak. Style neophile: Constantly seeking novel styles for domain generalization. In *CVPR*, 2022. 7
- [25] Zhiwei Ke, Zhiwei Wen, Weicheng Xie, Yi Wang, and Linlin Shen. Group-wise dynamic dropout based on latent semantic variations. In *AAAI*, 2020. 4
- [26] Daehee Kim, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. In *ICCV*, 2021. 7, 8
- [27] Kyungmoon Lee, Sungyeon Kim, and Suha Kwak. Cross-domain ensemble distillation for domain generalization. In *ECCV*, 2022. 2
- [28] Seungmin Lee, Dongwan Kim, Namil Kim, and Seong-Gyun Jeong. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *ICCV*, 2019. 2, 4
- [29] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017. 1, 6
- [30] Pan Li, Da Li, Wei Li, Shaogang Gong, Yanwei Fu, and Timothy M Hospedales. A simple feature augmentation for domain generalization. In *ICCV*, 2021. 2
- [31] Fangrui Lv, Jian Liang, Shuang Li, Bin Zang, Chi Harold Liu, Ziteng Wang, and Di Liu. Causality inspired representation learning for domain generalization. In *CVPR*, 2022. 2, 3, 4
- [32] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *ICML*, 2021. 1
- [33] Toshihiko Matsuura and Tatsuya Harada. Domain generalization using a mixture of multiple latent domains. In *AAAI*, 2020. 3, 6, 7
- [34] Rang Meng, Xianfeng Li, Weijie Chen, Shicai Yang, Jie Song, Xinchao Wang, Lei Zhang, Mingli Song, Di Xie, and Shiliang Pu. Attention diversification for domain generalization. In *ECCV*, 2022. 6, 7, 8, 9
- [35] Seonwoo Min, Nokyung Park, Siwon Kim, Seunghyun Park, and Jinkyu Kim. Grounding visual representations with texts for domain generalization. In *ECCV*, 2022. 7, 8
- [36] Pietro Morerio, Jacopo Cavazza, Riccardo Volpi, René Vidal, and Vittorio Murino. Curriculum dropout. In *ICCV*, 2017. 8

- [37] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *CVPR*, 2021. 7, 8
- [38] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *TNN*, 2010. 5
- [39] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE*, 2009. 1
- [40] Sungheon Park and Nojun Kwak. Analysis on the dropout effect in convolutional neural networks. In *ACCV*, 2016. 5
- [41] Sungrae Park, JunKeon Park, Su-Jin Shin, and Il-Chul Moon. Adversarial dropout for supervised and semi-supervised learning. In *AAAI*, 2018. 8
- [42] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019. 6
- [43] Jiaxin Qi, Kaihua Tang, Qianru Sun, Xian-Sheng Hua, and Hanwang Zhang. Class is invariant to context and vice versa: on learning invariance for out-of-distribution generalization. In *ECCV*, 2022. 2
- [44] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. In *ICML*, 2022. 7
- [45] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification. In *NeurIPS*, 2021. 9
- [46] Baifeng Shi, Dinghuai Zhang, Qi Dai, Zhanxing Zhu, Yadong Mu, and Jingdong Wang. Informative dropout for robust representation learning: A shape-bias perspective. In *ICML*, 2020. 2, 3, 5, 6, 7, 8, 9
- [47] Yang Shu, Zhangjie Cao, Chenyu Wang, Jianmin Wang, and Mingsheng Long. Open domain generalization with domain-augmented meta-learning. In *CVPR*, 2021. 1, 2
- [48] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014. 2, 4, 5
- [49] Xinwei Sun, Botong Wu, Xiangyu Zheng, Chang Liu, Wei Chen, Tao Qin, and Tie-Yan Liu. Recovering latent causal factor for generalization to distributional shifts. In *NeurIPS*, 2021. 2
- [50] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *CVPR*, 2015. 2
- [51] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR*, 2011. 6
- [52] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 6
- [53] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019. 2
- [54] Haohan Wang, Zexue He, Zachary C Lipton, and Eric P Xing. Learning robust representations by projecting superficial statistics out. In *ICLR*, 2019. 2
- [55] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. Generalizing to unseen domains: A survey on domain generalization. *TKDE*, 2022. 1
- [56] Yue Wang, Lei Qi, Yinghuan Shi, and Yang Gao. Feature-based style randomization for domain generalization. *TCSVT*, 2022. 2
- [57] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *ICCV*, 2021. 1, 7
- [58] Guoqiang Wei, Cuiling Lan, Wenjun Zeng, and Zhibo Chen. Metaalign: Coordinating domain alignment and classification for unsupervised domain adaptation. In *CVPR*, 2021. 1, 2
- [59] Guile Wu and Shaogang Gong. Collaborative optimization and aggregation for decentralized domain generalization and adaptation. In *ICCV*, 2021. 7
- [60] Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. R-drop: Regularized dropout for neural networks. In *NeurIPS*, 2021. 5
- [61] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A fourier-based framework for domain generalization. In *CVPR*, 2021. 1, 7, 9
- [62] Fu-En Yang, Yuan-Chia Cheng, Zu-Yun Shiau, and Yu-Chiang Frank Wang. Adversarial teacher-student representation learning for domain generalization. In *NeurIPS*, 2021. 2
- [63] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014. 4
- [64] Yuyuan Zeng, Tao Dai, Bin Chen, Shu-Tao Xia, and Jian Lu. Correlation-based structural dropout for convolutional neural networks. *PR*, 2021. 2, 4
- [65] Chao Zhang, Lei Zhang, and Jieping Ye. Generalization bounds for domain adaptation. In *NeurIPS*, 2012. 5
- [66] Dinghuai Zhang, Kartik Ahuja, Yilun Xu, Yisen Wang, and Aaron Courville. Can subnetwork structure be the key to out-of-distribution generalization? In *ICML*, 2021. 4
- [67] Jian Zhang, Lei Qi, Yinghuan Shi, and Yang Gao. Mvdg: A unified multi-view framework for domain generalization. In *ECCV*, 2022. 2, 6
- [68] Xingxuan Zhang, Peng Cui, Renzhe Xu, Linjun Zhou, Yue He, and Zheyang Shen. Deep stable learning for out-of-distribution generalization. In *CVPR*, 2021. 6, 7
- [69] Yabin Zhang, Minghan Li, Ruihuang Li, Kui Jia, and Lei Zhang. Exact feature distribution matching for arbitrary style transfer and domain generalization. In *CVPR*, 2022. 7, 8
- [70] Xingchen Zhao, Chang Liu, Anthony Sicilia, Seong Jae Hwang, and Yun Fu. Test-time fourier style calibration for domain generalization. In *IJCAI*, 2022. 6
- [71] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *TPAMI*, 2022. 1
- [72] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. In *AAAI*, 2020. 2, 7, 8
- [73] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *ICLR*, 2021. 7, 8, 9
- [74] Wei Zhu, Le Lu, Jing Xiao, Mei Han, Jiebo Luo, and Adam P Harrison. Localized adversarial domain generalization. In *CVPR*, 2022. 2