Discrepancy-guided Channel Dropout for Domain Generalization

Seonggyeom Kim*
Byeongtae Park*
Hanyang University
Seoul, South Korea
{canino,byeongtae}@hanyang.ac.kr

Harim Lee
Hanyang University
Seoul, South Korea
hrimlee@hanyang.ac.kr

Dong-Kyu Chae
Hanyang University
Seoul, South Korea
dongkyu@hanyang.ac.kr

ABSTRACT

Deep Neural Networks (DNNs) tend to perform poorly on unseen domains due to domain shifts. Domain Generalization (DG) aims to improve the performance on such scenarios by minimizing the distribution discrepancy between source domains. Among many studies, dropout-based DG approaches which remove domain-specific features have gained attention. However, they are limited in minimizing the upper bound of generalization risk because they do not explicitly consider the distribution discrepancy when discarding features. In this paper, we propose a novel **Discrepancy-guided Channel Dropout (DgCD)** for DG that explicitly derives the discrepancy between domains and drops the channels with significant distribution discrepancy. Given a training batch, we perform two ways of standardization: (1) based on the variance/mean of the batch (i.e., sampled from all source domains) and (2) based on the variance/mean of domain-wise samples in the batch. With the two normal distributions, we explicitly derive the discrepancy using KL-divergence and backpropagate it towards each channel. A channel with a higher contribution to the discrepancy is more likely to be dropped. Experimental results show the superiority of DgCD over the state-of-the-art DG baselines, demonstrating the effectiveness of our dropout strategy which is directly coupled to reducing the domain discrepancy. Our code is available at: https://github.com/gyeomo/DgCD

CCS CONCEPTS

• **Computing methodologies → Learning latent representations**.

KEYWORDS

Domain generalization, Dropout, Distribution discrepancy

ACM Reference Format:
Seonggyeom Kim, Byeongtae Park, Harim Lee, and Dong-Kyu Chae. 2024. Discrepancy-guided Channel Dropout for Domain Generalization. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3627673.3679539

*Both authors contributed equally to this research.

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679539

1 INTRODUCTION

Recently a large number of studies have demonstrated the success of deep neural networks (DNNs) in various vision tasks. However, recent discussions have raised concerns about performance degradation of DNNs due to differences between training and test distributions. This performance degradation is particularly significant in unseen domains affected by domain shift or covariate shift [39]. Domain generalization (DG) is an important area of research that addresses this problem. The goal of DG is to learn on a training dataset consisting of multiple source domains, each from different distributions while sharing the same label, to achieve good performance on unseen target domains [54].

It has been well established that the upper bound of the generalization risk for unseen domains can be expressed as the sum of the risks for each source domain and the pairwise differences [4] between the domains [1]. Following this theory, approaches such as discrepancy minimization [10, 57] and feature alignment [46, 47] reduced discrepancies between different source domains, leading models to learn comprehensive domain-invariant features and thereby achieve generalization to unseen domains. However, these studies are limited in that they do not explicitly remove domain-specific features in intermediate layers, which could lead to models learning unnecessary domain-specific information [23]. Furthermore, such over-fitting on source domains could degrade performance on unseen domains [24].

To mitigate these limitations, dropout-based DG methods [23, 24, 27, 49] have been explored to discard features that hinder generalization in domain shift. Originally, dropout [50] deactivates randomly selected neurons during training to prevent overfitting and promotes diverse feature learning, and this idea has been adapted for DG: Dropout-based DG methods aim to identify and mask domain-specific features that disturb generalization in domain shifts. However, these methods have a fundamental challenge in generalization because they do not account for the distribution discrepancy among source domains, which are essential for the risk minimization. For example, methods such as RSC [27], InfoDrop [49], and PLACE dropout [24] lack clear criteria for suppressing domain-specific features when training on multiple source domains, potentially leading to over-dependence on domain-related information. DomainDrop [23] introduces a domain discriminator to account for discrepancies between source domains, but faces potential conflicts between the domain discriminator objective and the label classifier objective [37]. In addition, its generalization performance can be dependent on the learnable domain discriminator.

To overcome these challenges, this paper proposes **D**iscrepancy-**g**uided **C**hannel **D**ropout (**DgCD**) for DG, which explicitly derives the distribution discrepancy between source domains, and removes features that highly contribute to this discrepancy. To explicitly derive the discrepancy, we induce two normal distributions for the output (batch) of a randomly selected layer during training, one based on the mean and variance of the batch, and the other based on the mean and variance of the samples from each domain within the batch. We then compute their KL-divergence, and backpropagate the contribution of each channel to the discrepancy using the gradient of the KL-divergence. Finally, we calculate the Euclidean distance between the contributions of the two distributions propagated to each channel, and the larger this distance is, the more likely each channel is to be deleted.

DgCD has several advantages over existing dropout-based DG methods: (1) It explicitly captures the distribution discrepancy between source domains without requiring additional learning parameters like the domain discriminator. It drops channels that lead to large domain discrepancy, thereby reducing the discrepancy for the generalization risk minimization. (2) It is working based on the statistics of samples from each domain, which allows to well-capture domain characteristics [36]. In addition, it removes features with significant domain discrepancy based on batch statistics, and thereby is robust to covariate shifts [5]. (3) In the widely-used DG benchmark DomainBed [22], our DgCD shows superior performance to state-of-the-arts. Additionally, it tends to reduce the domain discrepancy in the latent space more effectively and to be more robust to distribution shifts than the baselines including DomainDrop [23] and PLACE dropout [24].

2 RELATED WORK

2.1 Minimizing Domain Discrepancy

Several DG studies have focused on capturing the distribution discrepancy between multiple source domains. To improve generalization performance, these studies attempt to reduce the discrepancy among source domains, which are drawn from different distributions, using various distance functions such as optimal transport [14, 48], maximum mean discrepancy (MMD) [10, 33], mutual information (MI) [8], and KL-divergence [34, 57]. In particular, LDDG [34] theoretically demonstrated that using KL-divergence, when a domain's latent variable is within the manifold of the standard normal distribution, can lead to the upper bound of the generalization risk on an unseen (but related) target domain.

2.2 Feature Alignment for DG

This subsection reviews DG methods that try to capture the statistics of each source domain and use them to normalize and explicitly align features. AdaBN [36] assumes that domain-related knowledge can be expressed in batch normalization statistics, and proposes adaptive batch normalization to modulate unseen domain statistics. IBN-Net [42] adopts additional instance/batch normalization layers to learn spatial invariance properties. DSON [47] aligns the statistics of multiple source domains by learning affine parameters for per-domain normalization. Core [59] proposes a normalization technique that mixes up batch statistics of the same class from source and target domains to reduce the gap between their statistics. BNE [46] presents a strategy to map the features of each domain into a latent space using domain-specific batch normalization statistics. Unlike these methods, our approach does not require additional normalization layers and exploits normalization to capture the domain discrepancy and drop domain-specific features.

2.3 Dropout-based Approaches

The methods introduced here adopt the idea of dropout in DG to drop units (e.g., neurons, blocks, channels) that hinder generalization performance. RSC [27] discards neurons with high gradients, assuming that they are biased towards specific domains. InfoDrop [49] estimates patch-wise information to measure texture bias and removes highly biased patches. GSNR-guided dropout [38] eliminates model parameters with high gradient signal-to-noise ratios. CDG [19] uses domain-specific feature extractors to create gating masks for dropping channels by exploiting the gradients of mutual information. PLACE-dropout [24] introduces a strategy for random layer-wise and channel-wise dropout and a progressive scheme that adjusts the dropout rate. DomainDrop [23] distinguishes domain-sensitive channels using a learnable domain discriminator and drops them. Our method is different in that our dropout strategy is directly connected to reducing the distribution discrepancy among source domains, without additional parameters like domain discriminators for learning domain-specific features.

3 PRELIMINARY

3.1 Generalization Risk Minimization

We build our DgCD framework on top of Empirical Risk Minimization (ERM) [52] for multiple source domains. Let $\mathcal{D}_S = \{D_S^1, D_S^2, \ldots, D_S^K\}$ be a training dataset including K source domains. Each source domain D_S^k is associated with its own distribution $\mathbb{P}^k$, defined over a set of i.i.d samples $\{(x_i^k, y_i^k)\}_{i=1}^{N_k}$ where N_k is the total number of samples in k-th domain. Given a classification model $F(\cdot)$ parameterized by θ and a classification loss $\ell(\cdot)$ like cross entropy, the ERM objective for K source domains is defined as:

$$\min_{F:x\to\hat{y}} \mathcal{L}_{\text{ERM}} = \frac{1}{K}\sum_{k\in K}\frac{1}{N_k}\sum_{i=1}^{N_k}\ell(F(x_i^k;\theta), y_i^k)\ . \tag{1}$$

ERM might be effective for minimizing risk in datasets ($\mathcal{D}_s$) composed of i.i.d. samples, however, this i.i.d. assumption is often unrealistic in practical scenarios, especially in unseen domains $\mathcal{D}_U$ where generalization becomes challenging due to domain or covariate shifts. An alternative objective [1] is to minimize discrepancy between all pairs of source domains, which eventually leads to an upper bound for the generalization risk over unseen target domains, contingent on certain assumptions [15]. For a model F, let $R_S^k[F]$ and $R_U[F]$ be the risk for each source domain $\mathcal{D}_S^k$ and the risk for an unseen target domain $\mathcal{D}_U$, respectively. Then, if $\mathcal{D}_U$ is part of the overall training data distribution, the upper bound of $R_U[F]$ can be defined as below, based only on the given K source domains:

$$R_U[F] \leq \sum_{k=1}^{K}\pi_k R_S^k[F] + \zeta\ , \tag{2}$$

where ζ denotes pairwise divergence between the K source domains, obtained from a statistical distance like KL-divergence [34], and π_k denotes a non-negative coefficient. This upper bound indicates that even without observing $\mathcal{D}_U$, reducing the risk in source

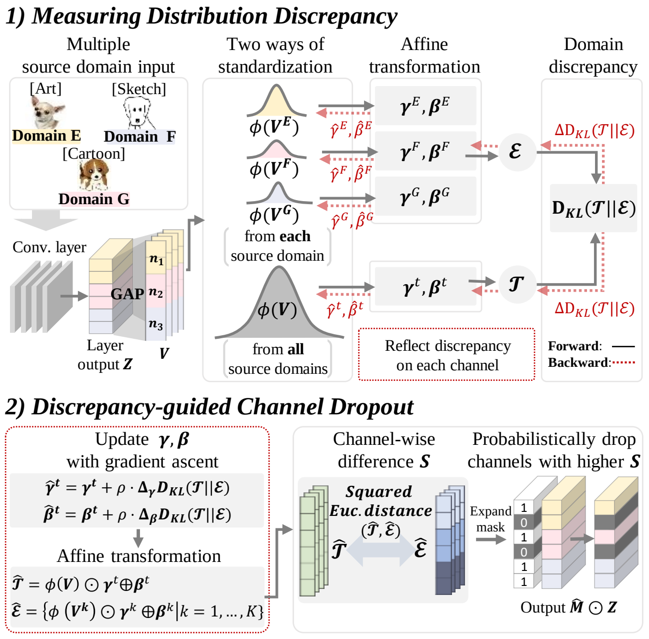

1) Measuring Distribution Discrepancy
Multiple source domain input
[Art]
[Sketch]
Domain E
Domain F
[Cartoon]
Domain G
Conv. layer
GAP
n_1
n_2
n_3
Layer output Z
V
Two ways of standardization
$\phi(V^E)$
$\phi(V^F)$
$\phi(V^G)$
from each source domain
$\phi(V)$
from all source domains
Affine transformation
γ^E, β^E
γ^F, β^F
γ^G, β^G
$\hat{\gamma}^E, \hat{\beta}^E$
$\hat{\gamma}^F, \hat{\beta}^F$
$\hat{\gamma}^G, \hat{\beta}^G$
γ^t, β^t
$\hat{\gamma}^t, \hat{\beta}^t$
$\mathcal{E}$
$\mathcal{T}$
Reflect discrepancy on each channel
Domain discrepancy
$\Delta D_{KL}(\mathcal{T}||\mathcal{E})$
$D_{KL}(\mathcal{T}||\mathcal{E})$
$\Delta D_{KL}(\mathcal{T}||\mathcal{E})$
Forward:
Backward:
2) Discrepancy-guided Channel Dropout
Update γ, β with gradient ascent
$\hat{\gamma}^t = \gamma^t + \rho \cdot \Delta_\gamma D_{KL}(\mathcal{T}||\mathcal{E})$
$\hat{\beta}^t = \beta^t + \rho \cdot \Delta_\beta D_{KL}(\mathcal{T}||\mathcal{E})$
Affine transformation
$\hat{\mathcal{T}} = \phi(V) \odot \gamma^t \oplus \beta^t$
$\hat{\mathcal{E}} = \{\phi(V^k) \odot \gamma^k \oplus \beta^k | k = 1, ..., K\}$
Channel-wise difference S
Squared Euc. distance $(\hat{\mathcal{T}}, \hat{\mathcal{E}})$
$\hat{\mathcal{T}}$
$\hat{\mathcal{E}}$
Expand mask
1
0
1
0
1
1
Probabilistically drop channels with higher S
Output $\hat{M} \odot Z$

Figure 1: Overview of the proposed DgCD.

domains and the pairwise divergence between them (ζ) helps generalization. Based on this theory, our study focuses on minimizing the discrepancy between source domains.

3.2 Channel-wise Dropout

Channel-wise dropout probabilistically drops a certain percentage of channels in the output of a given convolution layer. This can be used by dropout-based DG methods to explicitly discard domain-specific features in the latent space during training. Let the latent representation $\mathbf{Z} \in \mathbb{R}^{B \times C \times H \times W}$ be the output of any convolution layer of the model F, where B is the batch size, C is the number of channels, and H and W represent the height and width, respectively. For an instance $\mathbf{Z}_i$, its channel-wise dropout result $\hat{\mathbf{Z}}_i \in \mathbb{R}^{C \times H \times W}$ can be computed using a mask:

$$\hat{\mathbf{Z}}_i = \mathbf{M}_i \odot \mathbf{Z}_i, \\ s.t., \mathbf{M}_i \neq \mathbf{M}_j, \forall i, j = \{1, \dots, B\}, i \neq j, \tag{3}$$

where $\mathbf{M} \in \mathbb{R}^{B \times C \times H \times W}$ indicates the mask composed of zeros and ones and $\odot$ indicates element-wise multiplication.

3.3 Motivation

The upper bound serves as a theoretical guidance to reduce the risk for unseen domains, and based on it, there has been active research on minimizing the discrepancy between source domains for DG. However, these methods pose the problem that a model might learn unnecessary domain-specific features during training [23]. Dropout-based DG methods could shed light on solving this issue by discarding such features. However, their dropout strategy does not directly connected to reducing the discrepancy between source domains. We therefore design a novel DG approach that can minimize the risk for new domains by identifying and dropping channels that explicitly contribute to increasing the discrepancy between source domains.

4 METHODOLOGY

This section presents our method, named *Discrepancy-guided Channel Dropout* (DgCD). To explicitly capture the discrepancy between multiple source domains, we derive two normal distributions (one based on the total samples and the other based on the domain-wise samples) from the input batch during training. We then obtain the discrepancy through the KL-divergence between these two distributions. The computed KL-divergence is backpropagated to each channel to perform an affine transformation on the two normal distributions. Finally, we drop channels based on the squared Euclidean distance between the transformed distributions. Figure 1 shows the overview of our DgCD.

4.1 Discrepancy among Source Domains

This subsection explains how we capture the discrepancy between source domains using their statistics (mean and variance). Let $\mathcal{B}$ be an input batch for model training, composed of data randomly sampled from source domains, with an equal number of data points sampled from each domain, *i.e.*, $B = n_1 + \ldots + n_K$ and $n_1 = \ldots = n_K$, where n_k represents the number of samples from k-th source domain. Given an arbitrary convolution layer of a model F and its output representation $\mathbf{Z} \in \mathbb{R}^{B \times C \times H \times W}$, we perform Global Average Pooling (GAP) to produce $\mathbf{V} \in \mathbb{R}^{B \times C}$. Then, by standardizing $\mathbf{V}$ as follows, we can obtain a distribution that reflects the characteristics of all source domains [36]:

$$\phi(\mathbf{V}) = \frac{\mathbf{V} - \mathbb{E}[\mathbf{V}]}{\sqrt{\mathrm{Var}[\mathbf{V}] + \epsilon}} . \tag{4}$$

where $\mathbb{E}[\cdot] \in \mathbb{R}^{1 \times C}$ and $\mathrm{Var}[\cdot] \in \mathbb{R}^{1 \times C}$ represent the mean and variance, respectively. ϵ is a value added to the denominator for numerical stability.

Here, we consider two ways of standardization: one ($\mathcal{T}$) is the standardization of samples from all source domains $\mathbf{V}$ and the other ($\mathcal{E}$) is the standardization of samples from each domain $\mathbf{V}^k \in \mathbb{R}^{n_k \times C}$. We then consider the domain discrepancy as the difference between the two normal distributions. This is motivated by previous claims that (1) standardizing a batch in latent space makes the batch likely to follow a standard normal distribution [5] and (2) the difference in statistics between samples normalized per domain can be seen as the difference in distribution between domains [36, 47].

Specifically, we derive the two distributions $\mathcal{T} \in \mathbb{R}^{B \times C}$ and $\mathcal{E} \in \mathbb{R}^{B \times C}$ used to measure the discrepancy as follows (see Algorithm 1 for better understanding):

$$\mathcal{T} = \phi(\mathbf{V}) \odot \boldsymbol{\gamma}^t \oplus \boldsymbol{\beta}^t , \\ \mathcal{E} = \{\phi(\mathbf{V}^k) \odot \boldsymbol{\gamma}^k \oplus \boldsymbol{\beta}^k \mid k = 1, \dots, K\} , \tag{5}$$

where each $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are affine parameters, initialized to $\mathbf{1}$ and $\mathbf{0}$ respectively, and $\oplus$ stands for element-wise addition. The affine parameters $\boldsymbol{\gamma}^t, \boldsymbol{\beta}^t \in \mathbb{R}^{B \times C}$ are related to $\mathcal{T}$, and there are K sets of affine parameters associated with $\mathcal{E}$, each denoted by $\boldsymbol{\gamma}^k, \boldsymbol{\beta}^k \in \mathbb{R}^{n_k \times C}$. Throughout this section, we simply use $\boldsymbol{\gamma}, \boldsymbol{\beta}$ to denote the affine parameters where appropriate.

Algorithm 1 Calculation of $\mathcal{T}$ and $\mathcal{E}$

```
Input: V (representation obtained by Global Average Pooling), $\boldsymbol{\gamma}^t, \boldsymbol{\beta}^t$ (Affine parameters for $\mathcal{T}$), $\boldsymbol{\gamma}^k, \boldsymbol{\beta}^k$ (Affine parameters for $\mathcal{E}$), $K$ (# source domains)
Output: $\mathcal{T}, \mathcal{E}$ (Normal distributions)
1: Standardize V: $\phi(\mathbf{V}) = \frac{\mathbf{V}-\mathbb{E}[\mathbf{V}]}{\sqrt{\mathrm{Var}[\mathbf{V}]+\epsilon}}$
2: $\mathcal{T} \leftarrow \phi(\mathbf{V}) \odot \boldsymbol{\gamma}^t \oplus \boldsymbol{\beta}^t$
3: Initialize empty set $\mathcal{E}$
4: for all $k = 1$ to $K$ do
5:     Extract samples for $k$-th domain: $\mathbf{V}^k \leftarrow$ Samples from $k$-th domain contained in V
6:     Standardize $\mathbf{V}^k$: $\phi(\mathbf{V}^k) = \frac{\mathbf{V}^k-\mathbb{E}[\mathbf{V}^k]}{\sqrt{\mathrm{Var}[\mathbf{V}^k]+\epsilon}}$
7:     $\mathcal{E} \leftarrow \mathcal{E} \cup \{\phi(\mathbf{V}^k) \odot \boldsymbol{\gamma}^k \oplus \boldsymbol{\beta}^k\}$
8: end for
```

As a result, $\mathcal{T}$ is normalized with respect to the characteristics of all source domains, while $\mathcal{E}$ is normalized according to the characteristics of each domain. Although the two normal distributions ($\mathcal{T}$ and $\mathcal{E}$) have identical statistical parameters (mean of 0 and standard deviation of 1), they are normalized using different statistics, resulting in different values for the corresponding i-th sample. Specifically, $\mathcal{T}_i$ is normalized by the statistic of $\mathbf{V}$, while $\mathcal{E}_i$ is normalized by the statistic of $\mathbf{V}^k$, causing the i-th sample of each distribution to follow different distributional characteristics. Therefore, aggregating the gap between each sample of $\mathcal{T}$ and $\mathcal{E}$ can represent the distribution discrepancy across the source domains, which we interpret as the upper bound $\boldsymbol{\zeta}$ we aim to minimize. To estimate $\boldsymbol{\zeta}$, we compute the KL-divergence between $\mathcal{T}$ and $\mathcal{E}$ by:

$$D_{KL}(\mathcal{T}||\mathcal{E}) = \sum_i \mathcal{T}_i \log\left(\frac{\mathcal{T}_i}{\mathcal{E}_i}\right). \tag{6}$$

We consider this D_{KL} as $\boldsymbol{\zeta}$, representing the discrepancy between source domains.

Our next step is to separate the D_{KL} value by channel (in other words, to compute the discrepancy with respect to each channel) in order to find out channels with higher domain discrepancy and drop them. This task is not straightforward because of the interdependencies between the channels. The next subsection describes how we approach this task.

4.2 Affine Transformation via Gradients

According to recent studies [23, 24], channel-wise dropout can effectively disrupt the domain-specific patterns learned by the model, providing a strong regularization effect that improves the generalization performance. Based on these existing theoretical and empirical findings, we also adopt channel-wise dropout. To reflect the individual channel influence on the distribution discrepancy, we perform an affine transformation of $\mathcal{T}$ and $\mathcal{E}$ through the gradient for D_{KL}.

Initially, the affine parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are set to $\mathbf{0}$ and $\mathbf{1}$, respectively, so that the two normal distributions $\mathcal{T}$ and $\mathcal{E}$ are not affected. We assign the gradients calculated from D_{KL} to $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, and through this, we affine-transform $\mathcal{T}$ and $\mathcal{E}$, assigning the contribution of each channel to D_{KL} in terms of the two distributions [60]. Formally, we implement this process via the gradient ascending fashion, as follows:

$$\hat{\boldsymbol{\gamma}} = \boldsymbol{\gamma} + \rho \cdot \frac{\nabla_{\boldsymbol{\gamma}} D_{KL}(\mathcal{T}||\mathcal{E})}{\sqrt{\|\nabla_{\boldsymbol{\gamma}} D_{KL}(\mathcal{T}||\mathcal{E})\|_{2,C} + \epsilon}}, \tag{7}$$

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\beta} + \rho \cdot \frac{\nabla_{\boldsymbol{\beta}} D_{KL}(\mathcal{T}||\mathcal{E})}{\sqrt{\|\nabla_{\boldsymbol{\beta}} D_{KL}(\mathcal{T}||\mathcal{E})\|_{2,C} + \epsilon}}, \tag{8}$$

where $\nabla_{\boldsymbol{\gamma}} D_{KL}(\mathcal{T}||\mathcal{E}) \in \mathbb{R}^{B\times C}$ (because $\boldsymbol{\gamma} \in \mathbb{R}^{B\times C}$) is the gradient of D_{KL} with respect to $\boldsymbol{\gamma}$, and ρ is the update rate. $\|\cdot\|_{2,C}$ denotes the channel-wise L2-norm. By dividing the gradient by its L2-norm, we regulate the gradient's magnitude to prevent it from being biased toward a specific instance [61]. Note that the affine parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are initialized to zeros and ones at each iteration, and are only used to assign contributions of channels to the discrepancy, so they are not the parameters trained by the model F's objective.

Through the above gradient ascent method, $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are updated in a direction leading to an increase in D_{KL}. In this case, a large magnitude in the gradient of the i-th sample indicates a significant difference between the corresponding samples in $\mathcal{T}$ and $\mathcal{E}$. The magnitude of the gradient of D_{KL} can be expressed as the sum of the partial derivatives of each sample with respect to D_{KL}:

$$\begin{aligned}\|\nabla D_{KL}(\mathcal{T}||\mathcal{E})\|^2 &= \sum_i \left(\frac{\partial D_{KL}(\mathcal{T}||\mathcal{E})}{\partial \mathcal{T}_i}\right)^2 \\ &= \sum_i \left(\log\left(\frac{\mathcal{T}_i}{\mathcal{E}_i}\right) + 1\right)^2. \end{aligned}\tag{9}$$

This formula implies that the magnitude of the gradient of D_{KL} is proportional to the contribution of each sample to D_{KL}. In other words, the magnitude of the gradient represents the chance of a sample inducing the distribution discrepancy.

This idea can also be applied to channels, because a sample is composed of channels. We reflect the contribution to the discrepancy on the channels by updating the affine parameters $\boldsymbol{\gamma}, \boldsymbol{\beta}$ corresponding to the two normalized distributions $\mathcal{T}$ and $\mathcal{E}$ through Eq. (7). The affine parameters updated by gradient ascent create larger gaps for channels where the difference between the two distributions is significant, and smaller gaps where the difference is minor. Next, we compute the channel-wise difference between the two distributions, transformed by the affine parameters reflecting the discrepancy, and drop channels with higher differences.

4.3 Discrepancy-guided Channel Dropout

First, in order to reflect the discrepancy on each channel, we transform the two distributions using the updated $\hat{\boldsymbol{\gamma}}$ and $\hat{\boldsymbol{\beta}}$:

$$\begin{aligned}\hat{\mathcal{T}} &= \phi(\mathbf{V}) \odot \hat{\boldsymbol{\gamma}}^t \oplus \hat{\boldsymbol{\beta}}^t, \\ \hat{\mathcal{E}} &= \{\phi(\mathbf{V}^k) \odot \hat{\boldsymbol{\gamma}}^k \oplus \hat{\boldsymbol{\beta}}^k \mid k = 1, \ldots, K\}. \end{aligned}\tag{10}$$

Then, we obtain the channel-wise difference between $\hat{\mathcal{T}}$ and $\hat{\mathcal{E}}$ ($\mathrm{S} \coloneqq (\hat{\mathcal{T}} - \hat{\mathcal{E}})^2 \in \mathbb{R}^{B\times C}$), reflecting the discrepancy in terms of each channel, using the Squared Euclidean Distance (SED):

$$\begin{aligned} \mathbf{G} &= diag(\hat{\mathcal{T}}^\top \cdot \hat{\mathcal{E}})\,, \\ \mathbf{S} &= (\Phi_C(\hat{\mathcal{T}}/\mathbf{G}) - \Phi_C(\hat{\mathcal{E}}/\mathbf{G}))^2\,, \end{aligned} \tag{11}$$

where the $diag(\cdot)$ function extracts the diagonal elements. Here, to prevent channels with little information (channels with zero values) from hindering model training [49], we introduce the diagonal elements of the Gram matrix, $\mathbf{G} \in \mathbb{R}^{1\times C}$, representing the information (importance) of the channels. In addition, Φ_C in Eq. (11) is the min-max scaler with respect to the channel, introduced into both $\hat{\mathcal{T}}$ and $\hat{\mathcal{E}}$ to ensure that $\mathbf{S}$ should not be sensitive to the scale of the data. With these strategies, $\mathbf{S}$ increases for channels with large discrepancies and little information.

Based on the obtained $\mathbf{S}$, we now introduce the Weighted Random Selection (WRS) algorithm [20, 23] to probabilistically drop channels with high $\mathbf{S}$. We first draw a random sample $\mathbf{R} \in \mathbb{R}^{B\times C}$ of the same size as $\mathbf{S}$ from a uniform distribution $\text{Uniform}(0, 1)$. Then we derive $\mathbf{K}$, which serves as a criterion for channel selection in the top-k fashion, as follows:

$$\mathbf{K} = \mathbf{R}^{1/\Phi_C(\mathbf{S})},\ s.t., \mathbf{R} \in [0, 1], \tag{12}$$

where both $\mathbf{R}$ and $\Phi_C(\mathbf{S})$ have a range between 0 and 1. $\mathbf{K}$ is designed to have a higher probability of being larger as $\Phi_C(\mathbf{S})$ increases.

Next, we select the top channels from $\mathbf{K}$ according to a predefined dropout ratio (r), and derive a mask $\mathbf{M} \in \mathbb{R}^{B\times C}$ where the element corresponding to the selected channels are 0 and the rest are 1, as follows:

$$\mathbf{M}_i = \begin{cases} 0, & \text{if } i \in \text{Top}(\{\mathbf{K}_1, \ldots, \mathbf{K}_C\}, r\times C) \\ 1, & \text{otherwise} \end{cases}, \tag{13}$$

where i is the channel index and $\text{Top}(\{\mathbf{K}_1, \ldots, \mathbf{K}_C\}, r\times C)$ represents the top $r \times C$ elements in $\mathbf{K}$. Here, the mean of $\mathbf{M}$ is $(1 - r)$, which is less than 1. Multiplying a mask with a mean less than 1 with $\mathbf{Z}$ can reduce the magnitude of the input and potentially destabilize the training. For stability, we use $\hat{\mathbf{M}} = \mathbf{M} \times (B \times C)/\sum \mathbf{M}$, which has a mean of 1. Finally, we expand $\hat{\mathbf{M}}$ to the same shape as $\mathbf{Z}$ ($\mathbb{R}^{B\times C\times H\times W}$) and multiply it by $\mathbf{Z}$:

$$\mathbf{Z}_{drop} = \hat{\mathbf{M}} \odot \mathbf{Z} \tag{14}$$

As a result, a total of $r \times C$ channels inducing domain discrepancy per sample are masked to 0. Similar to the traditional dropout, the proposed dropout is also used only during the training phase, and not during the inference phase. Algorithm 2 shows the overall process of DgCD.

4.4 Theoretical Basis of Distribution Discrepancy

We further elaborate on the motivation behind the use of KL divergence to model the discrepancy (more details can be found in our supplementary material). DG works that aim to reduce the discrepancy between source distributions can face the problem of overfitting to the source domains [10, 48]. This is because the distribution of the unseen target domain may not align with the distributions of the multiple source domains. Consequently, simply minimizing the discrepancy between the source domains may not generalize well to the unseen domain.

Algorithm 2 Discrepancy-guided Channel Dropout

```
Input: Z (output representation), C (# channels), K (# source domains), r (dropout ratio), ρ (update rate)
Output: Z_drop (Output after dropout)
1: Initialize $\boldsymbol{\gamma}, \boldsymbol{\beta}$ to 0 and 1, respectively
2: Perform Global Average Pooling on $\mathbf{Z}$ to get $\mathbf{V}$
3: Compute $\phi(\mathbf{V})$ using Eq. (4)
4: Derive $\mathcal{T}$ and $\mathcal{E}$ using Eq. (5)
5: Compute $D_{KL}(\mathcal{T}\|\mathcal{E})$ using Eq. (6)
6: Update $\boldsymbol{\gamma}, \boldsymbol{\beta}$ using Eq. (7-8) for both $\mathcal{T}$ and $\mathcal{E}$
7: Derive $\hat{\mathcal{T}}$ and $\hat{\mathcal{E}}$ using Eq. (10)
8: Compute $\mathbf{S}$ using Eq. (11)
9: Apply Weighted Random Selection to get $\mathbf{M}$ using Eq. (12-13)
10: Expand $\mathbf{M}$ to the shape of $\mathbf{Z}$ to obtain $\hat{\mathbf{M}}$
11: Apply dropout: $\mathbf{Z}_{drop} = \hat{\mathbf{M}} \odot \mathbf{Z}$
```

To overcome this challenge, in a previous study, the authors of [34] aimed to fit the latent features of multiple source domains to a pre-defined prior distribution, $q_*(z)$, and adopted KL-divergence for this objective. Let the prior $q_*(z)$ be a computationally tractable normal distribution $\mathcal{N}(0, 1)$ computable via a reparameterization trick. In addition, assume that the latent features belonging to the unseen domain $\mathcal{D}_U$ with label y can also be represented by the latent features of the source domain with label y. The relation between the latent features of the unseen domain ($q_U(z)$) and those of the source domain ($q_k(z), k \in K$) can be expressed as follows:

$$\begin{aligned} D_{KL}(q_U(z)||q_*(z)) &= \sum_{k=1}^{K} \alpha_k \int_z q_k(z) \log \frac{q_U(z)}{q_*(z)} dz \\ &= \sum_{k=1}^{K} \alpha_k \int_z q_k(z) \log \frac{q_k(z)\,[1 + (q_U(z)/q_k(z) - 1)]}{q_*(z)} dz \\ &\leq \sum_{k=1}^{K} \alpha_k D_{KL}(q_k(z)||q_*(z)) \end{aligned} \tag{15}$$

where each α_k is a non-negative weight applied to the result of KL-divergence w.r.t. the corresponding k-th domain. This formulation implies that by regularizing the latent features of source domains to follow the prior distribution, the latent features of the unseen domain are also forced to follow the prior distribution.

Next, we explain our modeling of the discrepancy based on the aforementioned study. DgCD standardized the samples from source domains to transform them into a normal distribution. In this process, we created a normal distribution for samples from all source domains ($\mathcal{T}$) and the set of mixtures of distributions for samples from each source domain ($\mathcal{E}$). Following research on batch statistics [5, 29], we assume that the standardized distributions ($\mathcal{T}$ and $\mathcal{E}$) are likely to fall into $\mathcal{N}(0, 1)$. Under this assumption, both distributions are considered to have a mean of 0 and a standard deviation of 1 by standardization, and thus have the same parameters (mean and standard deviation) as the prior distribution, which can be represented as follows:

$$D_{KL}(\mathcal{T}||q_*(z)) \approx D_{KL}(\mathcal{E}||q_*(z)). \tag{16}$$

Table 1: Performance comparison of our DgCD with SOTA DG methods in the five benchmarks. Results are shown in accuracy (%), which are the average with three random seed runs. We also report their standard deviations. Best results are in bold. The results of all baselines except PLACE (†) are borrowed from their original papers (note that DomainDrop's results on the DomainBed protocol can be found in the Appendix of the DomainDrop paper [23]). The performance of PLACE [24] on DomainBed has not been reported yet, so we reproduced its performance.

Method	Venue	PACS	VLCS	OfficeHome	TerraIncognita	DomainNet	Avg.
ERM [22]	ICLR'20	85.5 ± 0.2	77.5 ± 0.4	66.5 ± 0.3	46.1 ± 1.8	40.9 ± 0.1	63.3
SagNet [40]	CVPR'21	86.3 ± 0.2	77.8 ± 0.5	68.1 ± 0.1	48.6 ± 1.0	40.3 ± 0.1	64.2
SelfReg [30]	ICCV'21	86.5 ± 0.3	77.5 ± 0.0	69.4 ± 0.2	51.0 ± 0.4	44.6 ± 0.1	65.8
MIRO [8]	ECCV'22	85.4 ± 0.4	79.0 ± 0.0	70.5 ± 0.4	50.4 ± 1.1	44.3 ± 0.2	65.9
PGrad-B [56]	ICLR'23	87.0 ± 0.1	78.9 ± 0.3	69.6 ± 0.1	49.4 ± 0.8	41.4 ± 0.1	65.3
DAC-SC [31]	CVPR'23	87.5 ± 0.1	78.7 ± 0.3	70.3 ± 0.2	46.5 ± 0.3	44.9 ± 0.1	65.6
SAGM [55]	CVPR'23	86.6 ± 0.2	**80.0** ± 0.3	70.1 ± 0.2	48.8 ± 0.9	45.0 ± 0.2	66.1
RSC [27]	ECCV'20	85.2 ± 0.9	77.1 ± 0.5	65.5 ± 0.9	46.6 ± 1.0	38.9 ± 0.5	62.7
PLACE †[24]	TOMM'23	86.3 ± 0.5	78.6 ± 0.4	68.1 ± 0.7	48.7 ± 0.5	42.1 ± 0.4	64.8
DomainDrop [23]	ICCV'23	87.9 ± 0.3	79.8 ± 0.3	68.7 ± 0.1	51.5 ± 0.4	44.4 ± 0.5	66.5
DgCD	Ours	**88.5** ± 0.2	79.9 ± 0.1	**70.6** ± 0.3	**52.9** ± 0.3	**45.4** ± 0.2	**67.5**

Table 2: Ablation on each component of DgCD, performed under the same environment as the comparative experiment.

Method	PACS	VLCS	OfficeHome	TerraInc.	DomainNet	Avg.
ERM †	83.7	78.1	65.1	46.4	42.4	63.1
Bernoulli	86.8	78.8	69.4	50.2	44.5	65.9
DgCD	88.5	79.9	70.6	52.9	45.4	67.5
w/o AT	88.2	79.0	69.7	51.4	45.2	66.7
GD	86.7	79.6	70.3	51.9	44.9	66.7
Rand	87.7	79.0	69.5	49.9	44.3	66.1
Last	84.5	78.6	68.4	48.8	44.3	64.9

If all source domains had the same distribution, the difference between samples from different source domains would be small, and the above Eq. (16) would hold. However, under the assumption of domain shift where source domains have different distributions, which is common in the field of DG, the discrepancy between $D_{KL}(\mathcal{T}||q_*(z))$ and $D_{KL}(\mathcal{E}||q_*(z))$ would be significant, and it would increase with the distributional difference between the source domains. Therefore, considering that both $\mathcal{T}$ and $\mathcal{E}$ are standardized and the prior distribution is $\mathcal{N}(0,1)$, we define $D_{KL}(\mathcal{T}||\mathcal{E})$, omitting the prior, as the discrepancy between source domains.

5 EXPERIMENTS

5.1 Settings

Datasets. Following the DomainBed settings [22], we employ five datasets: **PACS** [32] (9,991 images, 7 classes, and 4 domains), **VLCS** [21] (10,729 images, 5 classes, and 4 domains), **OfficeHome** [53] (15,588 images, 65 classes, and 4 domains), **TerraIncognita** [3] (24,788 images, 10 classes, and 4 domains), and **DomainNet** [22] (586,575 images, 345 classes, and 6 domains).

For a fair comparison, we also follow the dataset splitting method and evaluation protocol from DomainBed. The training/validation set is split in a ratio of 8:2. For the evaluation protocol, we adopt the leave-one-out cross-validation design: a single domain is used as the unseen domain (test set), and the remaining domains are used as the source domains (training set). Each domain is used once as the unseen domain, and the corresponding image classification performance is measured. All experiments are repeated three times using the random seed suggested by DomainBed.

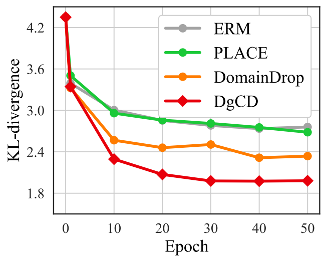
Implementation details. We use ResNet-50 [25] pre-trained on ImageNet [17] as the backbone. The model is then fine-tuned using the Adam optimizer. During training, the input batches to the model include samples from all source domains, with 32 samples per domain (e.g., for 3 source domains, the batch size is 96). We follow the hyperparameter search protocol proposed by the authors of DNA [11]. The model was trained for 20,000 steps on DomainNet and 5,000 steps on other datasets. At each training step, DgCD works on top of a randomly selected bottleneck layer of the ResNet backbone. In addition, the original dropout layer located in the last pooling layer of the backbone (the default setting of DomainBed) was replaced by our DgCD. More details can be found in our code and supplementary material.

5.2 Comparative Results

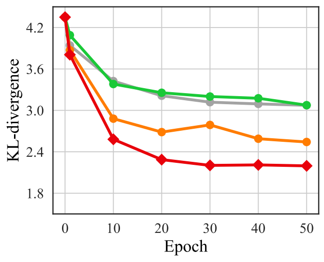
Table 1 compares the performance of various state-of-the-art DG methods with our DgCD. DgCD demonstrated the most superior performance on average. Notably, it achieved the best results on the PACS, OfficeHome, TerraIncognita, and DomainNet datasets, and exhibited performance nearly equivalent to the previous best on the VLCS dataset. Compared to other dropout-based DG methods, such as RSC [27], PLACE [24], and DomainDrop [23] which do not explicitly consider the discrepancy, our method outperformed in all datasets. Furthermore, DgCD generally showed lower standard errors, suggesting relatively stable model training. These results validate the effectiveness of DgCD's strategy in DG, which explicitly derives differences between source domains by estimating their statistical gap, and performs dropout based on such discrepancy.

5.3 Ablation Study

Here, we evaluate the effectiveness of each idea that constitutes DgCD by evaluating the following ablations: (1) computing channel-wise gaps using only SED, without the KL-divergence computation



(a) Source - Source



(b) Source - Target

Figure 2: (a): The avg. KL-divergence between pairs of the source domains in PACS, derived by the output embedding of the ResNet50 backbone. (b): The avg. KL-divergence between the source domains and the unseen target domain.

and affine transformation (denoted by **w/o AT**), (2) applying gradient descent instead of gradient ascent when updating $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ (**GD**), (3) applying dropout only at randomly selected bottleneck layers (**Rand**), and (4) applying dropout only at the last pooling layer of the backbone (**Last**). Additionally, as a baseline, we compare with methods trained without dropout (**ERM**) and with channel-wise dropout with completely random probability (**Bernoulli**).

Table 2 reports the experimental results. On average, ERM showed the lowest accuracy with 63.1%, followed by Bernoulli with 65.9%. The results of DgCD without applying affine transformation (w/o AT) and using gradient descent (GD) both recorded 66.7%, showing a decrease in accuracy compared to the original DgCD. This suggests that backpropagating the KL-divergence via gradient ascent to transform the statistics of $\mathcal{T}$ and $\mathcal{E}$ allows each channel to reflect the discrepancy effectively. In addition, Rand showed higher performance than Last. Unlike the previous study [24], which was unstable when applying dropout to multiple layers simultaneously, our DgCD, which performs dropout based on discrepancy, shows stable and superior performance even when applied to multiple layers. This indicates that DgCD is capable of successfully removing channels that are likely to hinder training.

5.4 In-depth Analysis of DgCD

In this section, we further analyze whether our DgCD effectively reduces the domain discrepancy and whether it is robust to covariate shift, in order to support the effectiveness of DgCD in generalization. For a fair comparison, we conducted the experiments in the same setting as DomainDrop (for this purpose, we did not exploit our last layer dropout).

5.4.1 *Domain Discrepancy Reduction.*

Evaluation in terms of KL-Divergence. To assess how effectively DgCD reduces the domain discrepancy, we measured the KL-divergence between the source domains, as well as between the source domains and the unseen domain. The outputs (embedding) of the last pooling layer of the ResNet backbone trained with our DgCD framework were used for calculating the KL-divergence. Figure 2(a) presents the results between the source domains, while (b) shows the results between the source domains and the unseen domain. Every domain in the PACS was set to unseen domain one by one, and the figure reports the average of all results. We observe that ERM and PLACE, which do not consider domain discrepancy, exhibit high

Table 3: Domain discrepancy measured by channel-level maximum mean discrepancy (CMMD), instead of KL-divergence. Following DomainDrop [23], we employed the ResNet18 backbone here. We report the results on PACS.

Method	Source-Source CMMD	Source-Target CMMD
ERM [22]	9.38	7.25
InfoDrop [49]	7.11	6.44
RSC [27]	8.81	7.20
PLACE [24]	6.45	6.17
DomainDrop [23]	5.15	4.09
DgCD	**4.85**	**3.79**

KL-divergence, and DomainDrop, which discards domain-specific features, shows lower KL-divergence than them. DgCD achieved the lowest KL-divergence than any of the baselines employed. These results confirm that DgCD not only effectively reduces the discrepancy between source domains (ζ in Eq. (2)), but also has the ability to significantly reduce the discrepancy between source domains and unseen domains. This ability might come from DgCD's dropout scheme that is directly related to reducing the domain discrepancy for minimizing the generalization risk bound.

Evaluation with regard to CMMD. We measured channel-level maximum mean discrepancy (CMMD) [23] for the embeddings used in DomainDrop. Let $Q : \mathcal{X} \to \mathbb{R}^C$ be the feature extractor of model F. Then, CMMD between an arbitrary pair of source domains (namely, $k1$ and $k2$) can be represented as follows:

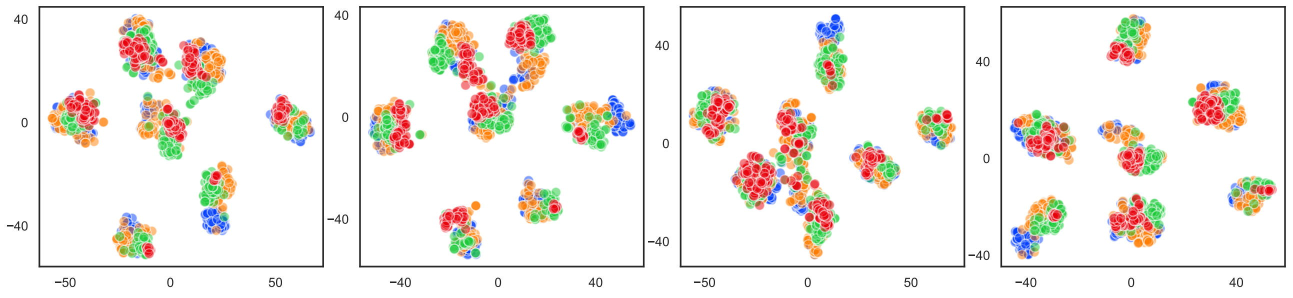
$$d_{CMMD}(D_S^{k1}, D_S^{k2}) = \frac{1}{C}\sum_{c=1}^{C} ||Q_c(D_S^{k1}) - Q_c(D_S^{k2})||_2 \qquad (17)$$

We calculated the CMMD between source domains (validation sets) and between source domains and the unseen target domain (test set). The experimental setup was constructed identically to that in DomainDrop [23]. Table 3 reports the results. The results for other methods were taken from the values reported in [23]. Similar to when domain discrepancy was measured using KL-divergence, DgCD exhibited the lowest CMMD compared to other methods.

5.4.2 *Covariate Shift.*

The experiments shown in this subsection aim to verify that DgCD, which works based on batch statistics, is robust to covariate shifts. We consider covariate shifts due to differences in distribution between source domains, and assume that training and validation data may have different distributions.

Embedding Cluster Analysis. First, for qualitative evaluation, we used T-SNE to visualize the embeddings (the ResNet backbone's output) of validation (source domains) and test (unseen domain) samples from PACS [51]. If clusters can be clearly formed according to the classes (PACS comprises 7 classes) even in domain shift or covariate shift, it means that the model well-captures the class-specific information across different distributions, indicating robustness to covariate shift [6, 59]. Figure 3 displays the results, where the red dots represent the test data from the unseen domain, and the other colors indicate the validation data from the source domains. The baselines generally show some overlap between several clusters. This overlap can be attributed to the increased variance of the features due to the dropout applied in the intermediate layers of



(a) ERM (b) PLACE (c) DomainDrop (d) DgCD

Figure 3: T-SNE visualization of the embeddings, where the red points correspond to the unseen domain (Photo) and the other colored points represent the source domains. The seven clusters formed by each model correspond to each class label in PACS.

Table 4: Quantitative evaluation on clustering embeddings.

Method	Adjusted Rand ↑	Silhoutte ↑	Davies Bouldin ↓
ERM	0.860	0.716	0.389
PLACE	0.882	0.703	0.410
DomainDrop	0.858	0.689	0.434
DgCD	**0.892**	**0.721**	**0.385**

the backbone during training, leading to covariate shift [29]. On the contrary, even though DgCD applied dropout to the randomly-selected intermediate layers, it formed the most distinct 7 clusters. Since DgCD derives discrepancy based on batch statistics, channels with high discrepancy indicate a significant difference between the mean and variance across source domains. Thus, DgCD has a minimal impact on the variance of the features during training, resulting in clearer clustering and thus the most robustness to covariate shift.

Furthermore, we provide a quantitative analysis of the clustering results. Table 4 shows a quantitative comparison of embedding quality using three metrics: *Adjusted Rand Score* [28] (similarity between actual and predicted clusters), *Silhouette Score* [45] (degree of overlap between clusters), and *Davies Bouldin Score* [16] (ratio of intra-cluster to inter-cluster distances). Our findings indicate that DgCD has better separation of instances by class and clustering within the same domain compared to PLACE and DomainDrop.

Robustness against Noise. We analyzed robustness to perturbations as another approach to assess robustness to covariate shift. The sensitivity of a model to noise can be measured via Model Robustness Score (MRS), which is based on the KL-divergence between the data with noise and the original data, as follows [12]:

$$\mathrm{MRS}(x, \delta) = D_{KL}(F(x;\theta)||F(x+\delta;\theta)). \quad (18)$$

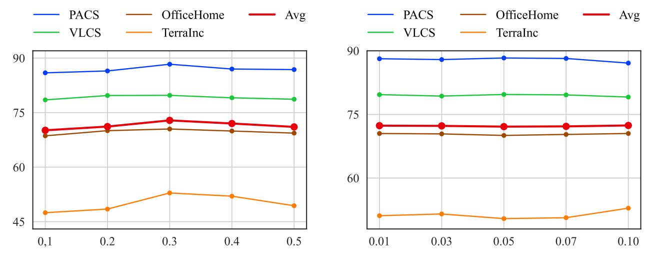
Following this equation, we calculated the KL-divergence between the embeddings of the validation and test data and the embeddings of the same data with various perturbations added. We analyzed the robustness to noise by adding the following four perturbations: '**Blur**' applies Gaussian blur, '**Noise**' adds Gaussian noise, '**Compression**' compresses and restores with JPEG, and '**Weather**' perturbs the data with snow (i.e., modifying a normal background image to look like it's snowing). Tables 5 and 6 show the results for the source domain (validation set) and unseen domain (test set), respectively. On average, DgCD showed the lowest KL-divergence,

Table 5: Robustness against perturbations. The validation set of PACS is used here.

Method	Blur	Noise	Compression	Weather	Avg.
ERM	2.876	2.645	2.922	2.775	2.804
PLACE	2.640	2.605	2.886	2.782	2.728
DomainDrop	2.395	2.386	2.697	2.366	2.461
DgCD	**1.903**	**1.877**	**2.637**	**1.943**	**2.090**

Table 6: Robustness against perturbations. The test (unseen target domain) set of PACS is used here.

Method	Blur	Noise	Compression	Weather	Avg.
ERM	2.833	2.765	2.938	2.913	2.862
PLACE	2.702	2.635	2.857	2.912	2.776
DomainDrop	2.422	2.421	**2.646**	2.480	2.492
DgCD	**2.077**	**2.057**	2.749	**2.343**	**2.306**



(a) Dropout ratio r (b) Update rate ρ

Figure 4: Hyperparameter analysis results.

which we interpret that DgCD is the most robust to perturbations than the baselines.

5.4.3 *Hyperparameter Analysis.* Figure 4 visualizes our hyperparameter analysis in PACS, VLCS, OfficeHome, and TerraIncognita. Overall, the performance was highest when the dropout ratio was around 0.3. Regarding the update rate, there was no significant difference overall; however, in PACS and VLCS, there was a tendency for performance to improve with a smaller update rate, while in OfficeHome and TerraIncognita, performance tended to improve with a larger update rate.

6 DISCUSSION

Applicability to Backbones without Channels. Although DgCD can be applied to any backbone based on CNNs, it may face minor challenges when applied to other architectures that do not have channels. For instance, the Vision Transformer (ViT) [18] doesn't have the concept of channels and instead introduces the concept of tokens. Therefore, adapting our DgCD to ViTs may require slight modifications. To explore such applicability while maintaining DgCD's key idea, we design an approach to drop *features per token* from the encoder output of ViTs, motivated by our observation that DgCD has shown effectiveness on ResNet block output (without a residual connection). To implement this, we take the average of the tokens from the encoder output to obtain a shape of (*batch size*, *feature dimension*), similar to how global average pooling (GAP) is used on a feature map of an image.

Table 7 shows the results of applying DgCD to the ViT-B backbone (pre-trained on CLIP [44]), compared to ERM [52], SWAD [7], SMA [2], and MIRO [8], all of which employ the same ViT-B backbone. We observe that DgCD outperforms ERM, SWAD, and SMA by a wide margin, but slightly underperforms MIRO. This might be due to the fact that the features we removed from the encoder output are actually a subset of image patches (tokens), rather than a subset of an entire image. Hence, applying our DgCD to ViT backbones needs further study on identifying suitable locations where DgCD can be applied to learn a lot of domain-invariant but class-specific features.

Single-source Domain Generalization. DgCD is basically designed to generalize to unseen domains across *multiple* source domains, but it can also be performed based on a *single* source domain by reducing distribution discrepancies *within batches* of the given single domain. To verify this, we designed a single-source DG task using the PACS dataset. We considered one domain of PACS as the source domain, and the rest domains as the unseen domains. Following [43], we applied the SGD optimizer with momentum 0.9, defined the batch size as 64, and set the learning rate to 2e-3 when training. We divided the input batch into *four pieces*, and applied DgCD to reduce the discrepancy between them.

We compared our performance with several data augmentation techniques, which are mainly studied in single-source DG research, such as Augmix [26], Mixstyle [62], DSU [35], and ACVC [13]. We also employed several dropout-based DG methods such as RSC [27], PLACE [24], and Bernoulli (drop channels with equal probability), as baselines. We also compared our method with pAdaIn [41], an adaptive normalization layer method, and MAD [43], a feature-modality alignment framework.

Table 8 shows the results. Even though it has been known that multiple-source DG approaches often face challenges in generalization based only on a single source domain [9, 58], our DgCD achieved meaningful performance in average, which is quite noteworthy. Compared to the dropout-based methods, DgCD outperforms them by a wide margin. Compared with the data augmentation strategies, DgCD beats most of them, but underperforms ACVC [13] when it is adopted by MAD [43]. However, if our DgCD employs ACVC (note that our DgCD can be used orthogonally with data augmentation), it performs the best among all the compared methods. These results indicate that DgCD is not only competitive enough for single-source DG tasks, but also has good synergy with nice data augmentation methods like ACVC. Furthermore, these results imply that DgCD can perform well without dividing the batches per domain. Therefore, even when the number of domains is very large, it is possible to divide the batches into a manageable number, which can improve generalization performance.

Table 7: Results of DgCD and several baselines using the ViT-B backbone, instead of ResNet-50.

Method	PACS	VLCS	OfficeHome	TerraInc.	DomainNet	Avg.
ERM	83.4	75.9	66.4	35.3	44.4	61.1
SWAD [7]	91.3	79.4	76.9	45.4	51.7	68.9
SMA [2]	92.1	79.7	78.1	48.3	55.9	70.8
MIRO [8]	**95.6**	**82.2**	**82.5**	**54.3**	54.0	**73.7**
DgCD	93.6	80.6	80.1	53.0	**56.2**	72.7

Table 8: Single-source DG performance on the PACS dataset using ResNet-18 as the backbone. The model is trained with the corresponding source domain and then evaluated on the remaining three domains. The average accuracy on the three domains is reported. The results are borrowed from MAD [43] except Bernoulli, RSC, and PLACE.

Method	photo	art painting	cartoon	sketch	Avg.
ERM	33.65	65.38	64.20	34.15	49.34
ERM w/ MAD [43]	32.32	66.47	69.80	34.54	50.78
Augmix [26]	38.30	66.54	70.16	52.48	56.87
pAdaIn [41]	33.66	64.96	65.24	32.04	48.98
Mixstyle [62]	37.44	67.60	70.38	34.57	52.50
DSU [35]	42.10	71.54	74.51	47.75	58.97
DSU+MAD [43]	44.15	72.41	74.47	49.60	60.16
ACVC [13]	48.05	73.68	**77.39**	55.30	63.61
ACVC+MAD [43]	52.95	75.51	77.25	57.75	65.87
ACVC+**DgCD**	**55.02**	**78.80**	77.09	**59.89**	**67.70**
Bernoulli	46.32	75.81	75.43	44.36	60.48
RSC [27]	43.03	74.20	75.27	**51.03**	60.88
PLACE [24]	46.54	75.85	75.57	45.23	60.80
DgCD	**51.37**	**76.96**	**76.67**	49.81	**63.70**

7 CONCLUSION

In this paper, we propose a novel dropout-based DG method, named DgCD, which controls the upper bound of generalization risk via channel dropout based on domain discrepancy to ensure generalization performance in domain shift. DgCD uses KL-divergence to explicitly capture domain discrepancy, and by using domain statistics, it identifies and removes channels that significantly contribute to the discrepancy, thereby reducing the upper bound of the generalization risk for unseen domains. DgCD has achieved state-of-the-art performance on five widely-used DG benchmarks, outperforming SOTA by 1% in average, which is definitely a meaningful achievement in the field of DG research. Our extensive analysis confirms the effectiveness of considering discrepancy in dropout-based DG, demonstrating superior performance in covariate shifts not only in unseen domains, but also in observed domains.

ACKNOWLEDGMENTS

This work was partly supported by (1) the National Research Foundation of Korea (NRF) grant funded by the Korea government (*MSIT) (No.2018R1A5A7059549) and (2) the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2020-II201373,Artificial Intelligence Graduate School Program (Hanyang University)). *Ministry of Science and ICT

REFERENCES

[1] Isabela Albuquerque, João Monteiro, Mohammad Darvishi, Tiago H Falk, and Ioannis Mitliagkas. 2019. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804* (2019).

[2] Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. 2022. Ensemble of averages: Improving model selection and boosting performance in domain generalization. *Advances in Neural Information Processing Systems* 35 (2022), 8265–8277.

[3] Sara Beery, Grant Van Horn, and Pietro Perona. 2018. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*. 456–473.

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning* 79 (2010), 151–175.

[5] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. 2018. Understanding batch normalization. *Advances in neural information processing systems* 31 (2018).

[6] João B. S. Carvalho, Mengtao Zhang, Robin Geyer, Carlos Cotrini, and Joachim M. Buhmann. 2023. Invariant Anomaly Detection under Distribution Shifts: A Causal Perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=F1mv2L7Rkb

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

[8] Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. 2022. Domain Generalization by Mutual-Information Regularization with Pre-trained Models. In *European Conference on Computer Vision*. 440–457.

[9] Liang Chen, Yong Zhang, Yibing Song, Ying Shan, and Lingqiao Liu. 2023. Improved test-time adaptation for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24172–24182.

[10] Mathieu Chevalley, Charlotte Bunne, Andreas Krause, and Stefan Bauer. 2022. Invariant causal mechanisms through distribution matching. *arXiv preprint arXiv:2206.11646* (2022).

[11] Xu Chu, Yujie Jin, Wenwu Zhu, Yasha Wang, Xin Wang, Shanghang Zhang, and Hong Mei. 2022. DNA: Domain generalization with diversified neural averaging. In *International Conference on Machine Learning*. PMLR, 4010–4034.

[12] WeiQin Chuah, Ruwan Tennakoon, Reza Hoseinnezhad, David Suter, and Alireza Bab-Hadiashar. 2024. Single Domain Generalization via Normalised Cross-correlation Based Convolutions. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 1752–1761.

[13] Ilke Cugu, Massimiliano Mancini, Yanbei Chen, and Zeynep Akata. 2022. Attention consistency on visual corruptions for single-source domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4165–4174.

[14] Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. 2018. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European conference on computer vision (ECCV)*. 447–463.

[15] Shai Ben David, Tyler Lu, Teresa Luu, and Dávid Pál. 2010. Impossibility theorems for domain adaptation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 129–136.

[16] David L Davies and Donald W Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence* 2 (1979), 224–227.

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

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[19] Dapeng Du, Jiawei Chen, Yuexiang Li, Kai Ma, Gangshan Wu, Yefeng Zheng, and Limin Wang. 2022. Cross-Domain Gated Learning for Domain Generalization. *International Journal of Computer Vision* 130, 11 (2022), 2842–2857.

[20] Pavlos S Efraimidis and Paul G Spirakis. 2006. Weighted random sampling with a reservoir. *Information processing letters* 97, 5 (2006), 181–185.

[21] Chen Fang, Ye Xu, and Daniel N Rockmore. 2013. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*. 1657–1664.

[22] Ishaan Gulrajani and David Lopez-Paz. 2020. In Search of Lost Domain Generalization. In *International Conference on Learning Representations*.

[23] Jintao Guo, Lei Qi, and Yinghuan Shi. 2023. Domaindrop: Suppressing domain-sensitive channels for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19114–19124.

[24] Jintao Guo, Lei Qi, Yinghuan Shi, and Yang Gao. 2023. PLACE Dropout: A Progressive Layer-wise and Channel-wise Dropout for Domain Generalization. *ACM Transactions on Multimedia Computing, Communications and Applications* 20, 3 (2023), 1–23.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[26] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. 2019. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781* (2019).

[27] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. 2020. Self-challenging Improves Cross-Domain Generalization. In *European Conference on Computer Vision*. 124–140.

[28] Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification* 2 (1985), 193–218.

[29] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*. pmlr, 448–456.

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

[31] Sangrok Lee, Jongseong Bae, and Ha Young Kim. 2023. Decompose, Adjust, Compose: Effective Normalization by Playing with Frequency for Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11776–11785.

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

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

[34] Haoliang Li, YuFei Wang, Renjie Wan, Shiqi Wang, Tie-Qiang Li, and Alex Kot. 2020. Domain generalization for medical imaging classification with linear-dependency regularization. *Advances in neural information processing systems* 33 (2020), 3118–3129.

[35] Xiaotong Li, Yongxing Dai, Yixiao Ge, Jun Liu, Ying Shan, and Ling-Yu Duan. 2022. Uncertainty modeling for out-of-distribution generalization. *arXiv preprint arXiv:2202.03958* (2022).

[36] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. 2016. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779* (2016).

[37] Jian Liang, Kaixiong Gong, Shuang Li, Chi Harold Liu, Han Li, Di Liu, Guoren Wang, et al. 2021. Pareto domain adaptation. *Advances in Neural Information Processing Systems* 34 (2021), 12917–12929.

[38] Mateusz Michalkiewicz, Masoud Faraki, Xiang Yu, Manmohan Chandraker, and Mahsa Baktashmotlagh. 2023. Domain Generalization Guided by Gradient Signal to Noise Ratio of Parameters. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6177–6188.

[39] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. 2013. Domain generalization via invariant feature representation. In *International conference on machine learning*. PMLR, 10–18.

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

[41] Oren Nuriel, Sagie Benaim, and Lior Wolf. 2021. Permuted adain: Reducing the bias towards global statistics in image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9482–9491.

[42] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. 2018. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 464–479.

[43] Sanqing Qu, Yingwei Pan, Guang Chen, Ting Yao, Changjun Jiang, and Tao Mei. 2023. Modality-agnostic debiasing for single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24142–24151.

[44] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision.

In *International conference on machine learning*. PMLR, 8748–8763.

[45] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.

[46] Mattia Segu, Alessio Tonioni, and Federico Tombari. 2023. Batch normalization embeddings for deep domain generalization. *Pattern Recognition* 135 (2023), 109115.

[47] Seonguk Seo, Yumin Suh, Dongwan Kim, Geeho Kim, Jongwoo Han, and Bohyung Han. 2020. Learning to Optimize Domain Specific Normalization for Domain Generalization. In *European Conference on Computer Vision*. 68–83.

[48] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. 2018. Wasserstein distance guided representation learning for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[49] Baifeng Shi, Dinghuai Zhang, Qi Dai, Zhanxing Zhu, Yadong Mu, and Jingdong Wang. 2020. Informative dropout for robust representation learning: A shape-bias perspective. In *International Conference on Machine Learning*. PMLR, 8828–8839.

[50] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

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

[52] Vladimir Vapnik. 1991. Principles of risk minimization for learning theory. *Advances in neural information processing systems* 4 (1991).

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

[54] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. 2022. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[55] Pengfei Wang, Zhaoxiang Zhang, Zhen Lei, and Lei Zhang. 2023. Sharpness-aware gradient matching for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3769–3778.

[56] Zhe Wang, Jake Grigsby, and Yanjun Qi. 2022. PGrad: Learning Principal Gradients For Domain Generalization. In *The Eleventh International Conference on Learning Representations*.

[57] Ziqi Wang, Marco Loog, and Jan Van Gemert. 2021. Respecting domain relations: Hypothesis invariance for domain generalization. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 9756–9763.

[58] Qinwei Xu, Ruipeng Zhang, Yi-Yan Wu, Ya Zhang, Ning Liu, and Yanfeng Wang. 2023. Simde: A simple domain expansion approach for single-source domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4797–4807.

[59] Fuming You, Jingjing Li, and Zhou Zhao. 2021. Test-time batch statistics calibration for covariate shift. *arXiv preprint arXiv:2110.04065* (2021).

[60] Tao Yu, Zongyu Guo, Xin Jin, Shilin Wu, Zhibo Chen, Weiping Li, Zhizheng Zhang, and Sen Liu. 2020. Region normalization for image inpainting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 12733–12740.

[61] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* 33 (2020), 5824–5836.

[62] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. 2021. Domain generalization with mixstyle. *arXiv preprint arXiv:2104.02008* (2021).