PromptStyler: Prompt-driven Style Generation for Source-free Domain Generalization

Junhyeong Cho[1] Gilhyun Nam[1] Sungyeon Kim[2] Hunmin Yang[1,3] Suha Kwak[2]
[1]ADD [2]POSTECH [3]KAIST
https://PromptStyler.github.io

Abstract

*In a joint vision-language space, a text feature (e.g., from "a photo of a dog") could effectively represent its relevant image features (e.g., from dog photos). Also, a recent study has demonstrated the cross-modal transferability phenomenon of this joint space. From these observations, we propose **PromptStyler** which simulates various distribution shifts in the joint space by synthesizing diverse styles via prompts without using any images to deal with source-free domain generalization. The proposed method learns to generate a variety of style features (from "a $\boldsymbol{S_*}$ style of a") via learnable style word vectors for pseudo-words $\boldsymbol{S_*}$. To ensure that learned styles do not distort content information, we force style-content features (from "a $\boldsymbol{S_*}$ style of a [class]") to be located nearby their corresponding content features (from "[class]") in the joint vision-language space. After learning style word vectors, we train a linear classifier using synthesized style-content features. PromptStyler achieves the state of the art on PACS, VLCS, OfficeHome and DomainNet, even though it does not require any images for training.*

1. Introduction

Deep neural networks are usually trained with the assumption that training and test data are independent and identically distributed, which makes them vulnerable to substantial distribution shifts between training and test data [23, 52]. This susceptibility is considered as one of the major obstacles to their deployment in real-world applications. To enhance their robustness to such distribution shifts, Domain Adaptation (DA) [2, 24, 32, 33, 54, 56, 57, 68] has been studied; it aims at adapting neural networks to a target domain using target domain data available in training. However, such a target domain is often latent in common training scenarios, which considerably limits the application of DA. Recently, a body of research has addressed this limitation by Domain Generalization (DG) [3, 5, 21, 29, 35, 37, 74] that aims to improve model's generalization capability to any unseen domains. It has been a common practice in DG to utilize multiple source domains for learning domain-invariant features [61, 69], but

DA和DG的问题

(a)
(b)

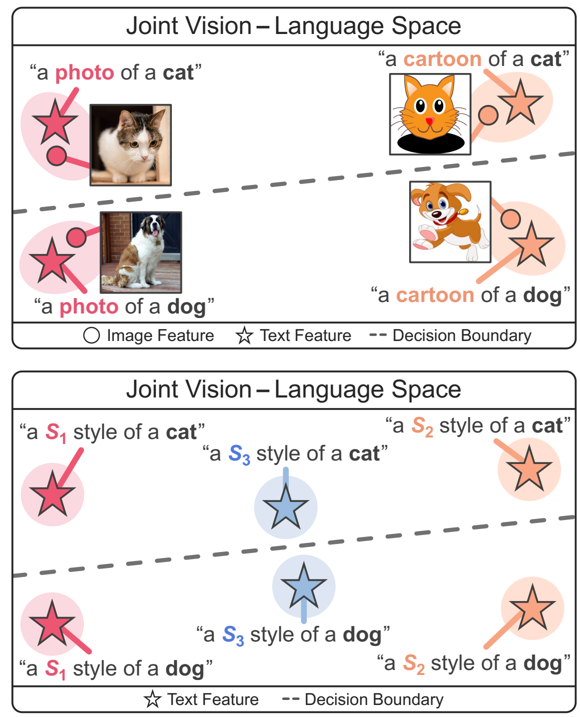


Figure 1: Motivation of our method. (a) Text features could effectively represent various image styles in a joint vision-language space. (b) PromptStyler synthesizes diverse styles in a joint vision-language space via learnable style word vectors for pseudo-words $\boldsymbol{S_*}$ without using any images.

it is unclear which source domains are ideal for DG, since arbitrary unseen domains should be addressed. Furthermore, it is costly and sometimes even infeasible to collect and annotate large-scale multi-source domain data for training.

We notice that a large-scale pre-trained model might have already observed a great variety of domains and thus can be used as an efficient proxy of actual multiple source domains. From this perspective, we raised a question *"Could we further improve model's generalization capability by simulating various distribution shifts in the latent space of such a large-scale model without using any source domain data?"* If this

提出利用大模型模拟数据分布变化

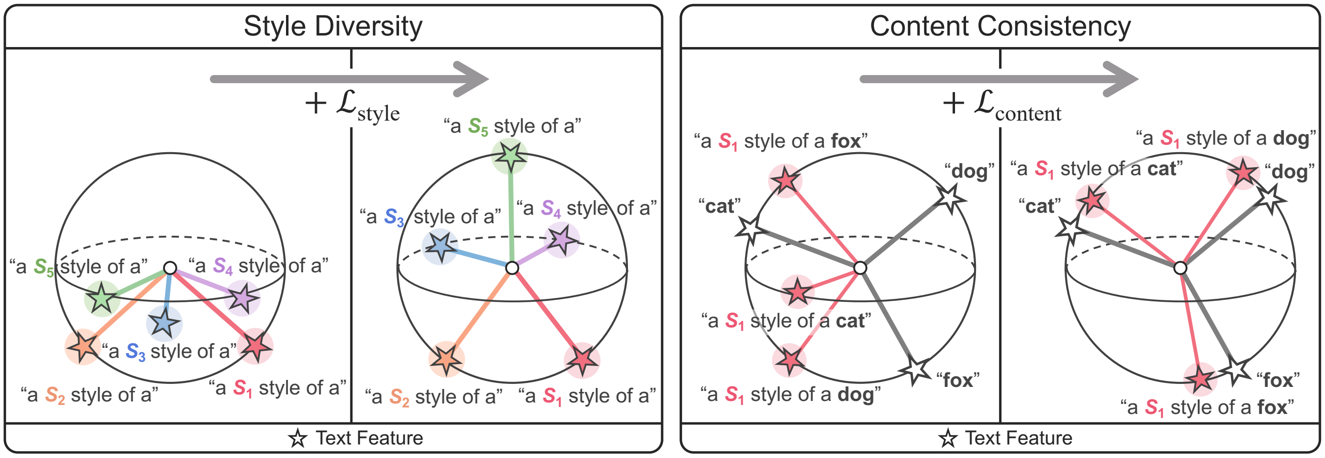


Figure 2: Important factors in the proposed method. PromptStyler learns style word vectors for pseudo-words $\boldsymbol{S}_*$ which lead to diverse style features (from "a $\boldsymbol{S}_*$ style of a") while preserving content information encoded in style-content features (from "a $\boldsymbol{S}_*$ style of a [class]"). $\mathcal{L}_{\text{style}}$ and $\mathcal{L}_{\text{content}}$ are the loss functions used for maximizing *style diversity* and *content consistency* in a hyperspherical joint vision-language space (*e.g.*, CLIP [50] latent space).

is possible, DG will become immensely practical by effectively and efficiently exploiting such a large-scale model. However, this approach is much more challenging since any actual data of source and target domains are not accessible but only the target task definition (*e.g.*, class names) is given.

提出本文方法，用文本特征训练分类器，然后图像特征来推理

In this paper, we argue that large-scale vision-language models [26, 50, 64] could shed light on this challenging *source-free domain generalization*. As conceptually illustrated in Figure 1(a), text features could effectively represent their relevant image features in a joint vision-language space. Despite the modality gap between two modalities in the joint space [39], a recent study has demonstrated the cross-modal transferability phenomenon [67]; we could train a classifier using text features while running an inference with the classifier using image features. This training procedure meets the necessary condition for the source-free domain generalization, *i.e.*, source domain images are not required. Using such a joint vision-language space, we could simulate various distribution shifts via prompts without any images.

提出提示词驱动风格生成，因为风格表征域

We propose a prompt-driven style generation method, dubbed **PromptStyler**, which synthesizes diverse styles via learnable word vectors to simulate distribution shifts in a hyperspherical joint vision-language space. PromptStyler is motivated by the observation that a shared style of images could characterize a domain [27, 74] and such a shared style could be captured by a learnable word vector for a pseudo-word $\boldsymbol{S}_*$ using CLIP [50] with a prompt ("a painting in the style of $\boldsymbol{S}_*$") [17]. As shown in Figure 1(b), our method learns a style word vector for $\boldsymbol{S}_*$ to represent each style.

最大化风格多样性和内容一致性

To effectively simulate various distribution shifts, we try to maximize *style diversity* as illustrated in Figure 2. Specifically, our method encourages learnable style word vectors to result in orthogonal style features in the hyperspherical space, where each style feature is obtained from a **style prompt** ("a $\boldsymbol{S}_*$ style of a") via a pre-trained text encoder. To prevent learned styles from distorting content information, we also consider *content consistency* as illustrated in Figure 2. Each style-content feature obtained from a **style-content prompt** ("a $\boldsymbol{S}_*$ style of a [class]") is forced to be located closer to its corresponding content feature obtained from a **content prompt** ("[class]") than the other content features.

本文方法操作过程

Learned style word vectors are used to synthesize style-content features for training a classifier; these synthesized features could simulate images of known contents with diverse unknown styles in the joint space. These style-content features are fed as input to a linear classifier which is trained by a classification loss using contents ("[class]") as their class labels. At inference time, an image encoder extracts image features from input images, which are fed as input to the trained classifier. Note that the text and image encoders are derived from the same pre-trained vision-language model (*e.g.*, CLIP [50]); the text encoder is only involved in training and the image encoder is only involved at inference time.

The proposed method achieves state-of-the-art results on PACS [34], VLCS [15], OfficeHome [60] and DomainNet [48] without using any actual data of source and target domains. It takes just ∼30 minutes for the entire training using a single RTX 3090 GPU, and our model is ∼2.6× smaller and ∼243× faster at inference compared with CLIP [50].

Our contributions are summarized as follows:

- This work is the first attempt to synthesize a variety of styles in a joint vision-language space via prompts to effectively tackle source-free domain generalization.
- This paper proposes a novel method that effectively simulates images of known contents with diverse unknown styles in a joint vision-language space.
- PromptStyler achieves the state of the art on domain generalization benchmarks without using any images.

Setup	Source	Target	Task Definition
DA	✓	✓	✓
DG	✓	–	✓
Source-free DA	–	✓	✓
Source-free DG	–	–	✓

Table 1: Different requirements in each setup. Source-free DG only assumes the task definition (*i.e.*, what should be predicted) without requiring source and target domain data.

2. Related Work

Domain Generalization. Model's generalization capability to arbitrary unseen domains is the key factor to successful deployment of neural networks in real-world applications, since substantial distribution shifts between source and target domains could significantly degrade their performance [23, 52]. To this end, Domain Generalization (DG) [4, 5, 10, 16, 21, 29, 35, 37, 44, 45, 61, 69] has been studied. It assumes target domain data are not accessible while using data from source domains. Generally speaking, existing DG methods could be divided into two categories: multi-source DG [3, 12, 36, 42, 43, 51, 55, 63, 73, 74] and single-source DG [14, 38, 49, 62]. Mostly, multi-source DG methods aim to learn domain-invariant features by exploiting available multiple source domains, and single-source DG methods also aim to learn such features by generating diverse domains based on a single domain and then exploiting the synthesized domains.

DG：多源和单源

Source-free Domain Generalization. In this setup, we are not able to access any source and target domains as summarized in Table 1. Thus, source-free DG is much more challenging than multi-source and single-source DG. From the observation that synthesizing new domains from the given source domain could effectively improve model's generalization capability [27, 38, 62, 72, 73], we also try to generate diverse domains but without using any source domains to deal with source-free DG. By leveraging a large-scale pre-trained model which has already seen a great variety of domains, our method could simulate various distribution shifts in the latent space of the large-scale model. This approach has several advantages compared with existing DG methods; source domain images are not required and there is no concern for catastrophic forgetting which might impede model's generalization capability. Also, it would be immensely practical to exploit such a large-scale model for downstream visual recognition tasks, since we only need the task definition.

SFDG：利用大模型模拟潜在空间的分布偏移

Large-scale model in Domain Generalization. Recently, several DG methods [5, 53] exploit a large-scale pre-trained model (*e.g.*, CLIP [50]) to leverage its great generalization capability. While training neural networks on available data, CAD [53] and MIRO [5] try to learn robust features using such a large-scale model. Compared with them, the proposed method could learn domain-invariant features using a large-scale pre-trained model without requiring any actual data.

利用大模型学习域不变性特征

Joint vision-language space. Large-scale vision-language models [26, 50, 64] are trained with a great amount of image-text pairs, and achieve state-of-the-art results on downstream visual recognition tasks [20, 41, 66, 70, 71]. By leveraging their joint vision-language spaces, we could also effectively manipulate visual features via prompts [13, 18, 31, 47]. Interestingly, Textual Inversion [17] shows that a learnable style word vector for a pseudo-word $\boldsymbol{S}_*$ could capture a shared style of images using CLIP [50] with a prompt ("a painting in the style of $\boldsymbol{S}_*$"). From this observation, we argue that learnable style word vectors would be able to seek a variety of styles for simulating various distribution shifts in a joint vision-language space without using any images.

3. Method

The overall framework of the proposed method is shown in Figure 3, and pseudo-code of PromptStyler is described in Algorithm 1. Our method learns style word vectors to represent a variety of styles in a hyperspherical joint vision-language space (*e.g.*, CLIP [50] latent space). After learning those style word vectors, we train a linear classifier using synthesized style-content features produced by a pre-trained text encoder $T(\cdot)$. At inference time, a pre-trained image encoder $I(\cdot)$ extracts image features from input images, which are fed as input to the trained linear classifier. Thanks to the cross-modal transferability phenomenon of the joint vision-language space [67], this classifier could produce class scores using the image features. Note that we exploit CLIP as our large-scale vision-language model; its image encoder and text encoder are frozen in our entire framework.

整体流程介绍

3.1. Prompt-driven style generation

An input text prompt is converted to several tokens via a tokenization process, and then such tokens are replaced by their corresponding word vectors via a word lookup process. In PromptStyler, a pseudo-word $\boldsymbol{S}_i$ in a prompt is a placeholder which is replaced by a style word vector $\mathbf{s}_i \in \mathbb{R}^D$ during the word lookup process. Note that three kinds of prompts are used in the proposed method: a style prompt $\mathcal{P}_i^{\text{style}}$ ("a $\boldsymbol{S}_i$ style of a"), a content prompt $\mathcal{P}_m^{\text{content}}$ ("[class]$_m$"), and a style-content prompt $\mathcal{P}_i^{\text{style}} \circ \mathcal{P}_m^{\text{content}}$ ("a $\boldsymbol{S}_i$ style of a [class]$_m$"). $\boldsymbol{S}_i$ indicates the placeholder for i-th style word vector and [class]$_m$ denotes m-th class name.

风格词向量生成过程

Suppose we want to generate K different styles in a joint vision-language space. In this case, the proposed method needs to learn K style word vectors $\{\mathbf{s}_i\}_{i=1}^K$, where each $\mathbf{s}_i$ is randomly initialized at the beginning. To effectively simulate various distribution shifts in the joint vision-language space, those style word vectors need to be diverse while not distorting content information when they are exploited in style-content prompts. There are two possible design choices for learning such word vectors: (1) learning each style word vector $\mathbf{s}_i$ in a sequential manner, or (2) learning all style

风格词向量随机初始化和学习方式

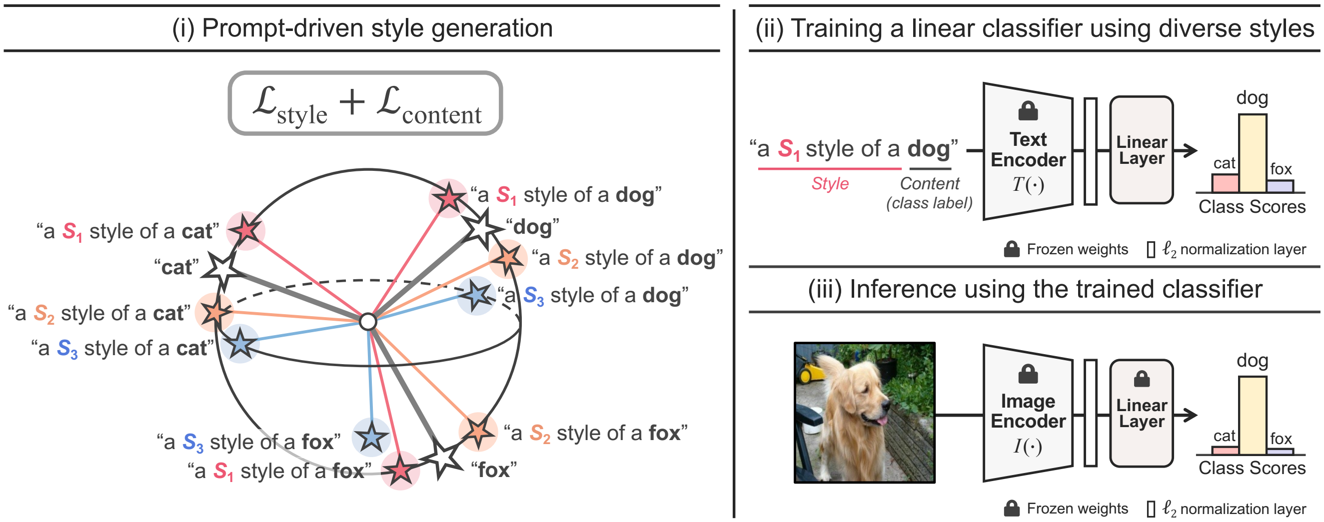


Figure 3: PromptStyler learns diverse style word vectors which do not distort content information of style-content prompts. After learning style word vectors, we synthesize style-content features (*e.g.*, from "a S_1 style of a **dog**") via a pre-trained text encoder for training a linear classifier. The classifier is trained by a classification loss using those synthesized features and their corresponding class labels (*e.g.*, "**dog**"). At inference time, a pre-trained image encoder extracts image features, which are fed as input to the trained classifier. Note that the encoders are derived from the same vision-language model (*e.g.*, CLIP [50]).

word vectors $\{\mathbf{s}_i\}_{i=1}^K$ in a parallel manner. We choose the former, since it takes much less memory during training. Please refer to the supplementary material (Section A.2) for the empirical justification of our design choice.

Style diversity loss. To maximize the diversity of K styles in a hyperspherical joint vision-language space, we sequentially learn style word vectors $\{\mathbf{s}_i\}_{i=1}^K$ in such a way that i-th style feature $T(\mathcal{P}_i^{\text{style}}) \in \mathbb{R}^C$ 正交 produced by i-th style word vector $\mathbf{s}_i$ is orthogonal to $\{T(\mathcal{P}_j^{\text{style}})\}_{j=1}^{i-1}$ produced by previously learned style word vectors $\{\mathbf{s}_j\}_{j=1}^{i-1}$. Regarding this, the style diversity loss $\mathcal{L}_{\text{style}}$ for learning i-th style word vector $\mathbf{s}_i$ is computed by

$$\mathcal{L}_{\text{style}} = \frac{1}{i-1}\sum_{j=1}^{i-1}\left|\frac{T(\mathcal{P}_i^{\text{style}})}{\|T(\mathcal{P}_i^{\text{style}})\|_2} \cdot \frac{T(\mathcal{P}_j^{\text{style}})}{\|T(\mathcal{P}_j^{\text{style}})\|_2}\right| . \quad (1)$$

This style loss $\mathcal{L}_{\text{style}}$ aims to minimize the absolute value of the cosine similarity between i-th style feature and each of the existing style features. When the value of this loss 正交性 becomes zero, it satisfies the orthogonality between i-th style feature and all the existing style features.

Content consistency loss. Learning the style word vectors $\{\mathbf{s}_i\}_{i=1}^K$ only using the style diversity loss sometimes leads to undesirable outcome, since a learned style $\mathbf{s}_i$ could substantially distort content information when used to generate a style-content feature $T(\mathcal{P}_i^{\text{style}} \circ \mathcal{P}_m^{\text{content}}) \in \mathbb{R}^C$. To 缓解 alleviate this problem, we encourage the content information in the style-content feature to be consistent with its corresponding content feature $T(\mathcal{P}_m^{\text{content}}) \in \mathbb{R}^C$ while learning each i-th style word vector $\mathbf{s}_i$. Specifically, each style-content feature synthesized via i-th style word vector $\mathbf{s}_i$ should have the highest cosine similarity score with its corresponding content feature. For i-th style word vector $\mathbf{s}_i$, a cosine similarity score z_{imn} between a style-content feature with m-th class name and a content feature with n-th class name is computed by

$$z_{imn} = \frac{T(\mathcal{P}_i^{\text{style}} \circ \mathcal{P}_m^{\text{content}})}{\|T(\mathcal{P}_i^{\text{style}} \circ \mathcal{P}_m^{\text{content}})\|_2} \cdot \frac{T(\mathcal{P}_n^{\text{content}})}{\|T(\mathcal{P}_n^{\text{content}})\|_2} . \quad (2)$$

Using cosine similarity scores between style-content features and content features, the content consistency loss $\mathcal{L}_{\text{content}}$ for learning i-th style word vector $\mathbf{s}_i$ is computed by

$$\mathcal{L}_{\text{content}} = -\frac{1}{N}\sum_{m=1}^{N} \log\left(\frac{\exp(z_{imm})}{\sum_{n=1}^{N}\exp(z_{imn})}\right), \quad (3)$$

where N denotes the number of classes pre-defined in the target task. This content loss $\mathcal{L}_{\text{content}}$ is a contrastive loss which encourages each style-content feature to be located closer to its corresponding content feature so that it forces each i-th style word vector $\mathbf{s}_i$ to preserve content information when used to synthesize style-content features.

Total prompt loss. PromptStyler learns K style word vectors $\{\mathbf{s}_i\}_{i=1}^K$ in a sequential manner, where each i-th style word vector $\mathbf{s}_i$ is learned using both $\mathcal{L}_{\text{style}}$ (Eq. (1)) and $\mathcal{L}_{\text{content}}$ (Eq. (3)). In the proposed method, the total loss $\mathcal{L}_{\text{prompt}}$ for learning i-th style word vector is computed by

$$\mathcal{L}_{\text{prompt}} = \mathcal{L}_{\text{style}} + \mathcal{L}_{\text{content}} . \quad (4)$$

Using this prompt loss $\mathcal{L}_{\text{prompt}}$, we train i-th style word vector $\mathbf{s}_i$ for L training iterations.

Algorithm 1 PromptStyler

Requirement: pre-trained text encoder $T(\cdot)$, pre-defined N class names in the target task

Input: number of style word vectors K, number of training iterations L

Output: KN style-content features

randomly initialize style word vectors
$\{\mathbf{s}_i\}_{i=1}^K \leftarrow \texttt{random\_initialize}(\{\mathbf{s}_i\}_{i=1}^K)$
# sequentially learn K style word vectors
for $i = 1, 2, \ldots, K$ **do**
  # L training iterations for learning each word vector
  for iteration $= 1, 2, \ldots, L$ **do**
    # compute $\mathcal{L}_{\text{style}}$ using $T(\cdot)$ and word vectors
    $\mathcal{L}_{\text{style}} \leftarrow \texttt{style\_diversity\_loss}(\mathbf{s}_i, \{\mathbf{s}_j\}_{j=1}^{i-1})$
    # compute $\mathcal{L}_{\text{content}}$ using $T(\cdot)$ and a word vector
    $\mathcal{L}_{\text{content}} \leftarrow \texttt{content\_consistency\_loss}(\mathbf{s}_i)$
    $\mathcal{L}_{\text{prompt}} \leftarrow \mathcal{L}_{\text{style}} + \mathcal{L}_{\text{content}}$
    Update $\mathbf{s}_i$ using $\mathcal{L}_{\text{prompt}}$ by gradient descent
  end for
end for
Synthesize KN style-content features using the learned K style word vectors and N class names via $T(\cdot)$

3.2. Training a linear classifier using diverse styles

After learning K style word vectors $\{\mathbf{s}_i\}_{i=1}^K$, we generate KN style-content features for training a linear classifier. To be specific, we synthesize those features using the learned K styles and pre-defined N classes via the text encoder $T(\cdot)$. The linear classifier is trained by a classification loss using ℓ_2-normalized style-content features and their class labels; each class label is the class name used to generate each style-content feature. To effectively leverage the hyperspherical joint vision-language space, we adopt ArcFace [8] loss as our classification loss $\mathcal{L}_{\text{class}}$. Note that ArcFace loss is an angular Softmax loss which computes the cosine similarities between classifier input features and classifier weights with an additive angular margin penalty between classes. This angular margin penalty allows for more discriminative predictions by pushing features from different classes further apart. Thanks to the property, this angular Softmax loss has been widely used in visual recognition tasks [7, 9, 30, 40, 65].

3.3. Inference using the trained classifier

The trained classifier is used with a pre-trained image encoder $I(\cdot)$ at inference time. Given an input image $\mathbf{x}$, the image encoder extracts its image feature $I(\mathbf{x}) \in \mathbb{R}^C$, which is mapped to the hyperspherical joint vision-language space by ℓ_2 normalization. Then, the trained classifier produces class scores using the ℓ_2-normalized image feature. Note that the text encoder $T(\cdot)$ is not used at inference time, while the image encoder $I(\cdot)$ is only exploited at inference time.

4. Experiments

For more comprehensive understanding, please refer to the supplementary material (Section B and D).

4.1. Evaluation datasets

The proposed method does not require any actual data for training. To analyze its generalization capability, four domain generalization benchmarks are used for evaluation: **PACS** [34] (4 domains and 7 classes), **VLCS** [15] (4 domains and 5 classes), **OfficeHome** [60] (4 domains and 65 classes) and **DomainNet** [48] (6 domains and 345 classes). On these benchmarks, we repeat each experiment three times using different random seeds and report average top-1 classification accuracies with standard errors. Unlike the *leave-one-domain-out cross-validation* evaluation protocol [21], we do not exploit any source domain data for training.

4.2. Implementation details

PromptStyler is implemented and trained with the same configuration regardless of the evaluation datasets. Training takes about 30 minutes using a single RTX 3090 GPU.

Architecture. We choose CLIP [50] as our large-scale pre-trained vision-language model, and use the publicly available pre-trained model.[1] The text encoder $T(\cdot)$ used in training is Transformer [59] and the image encoder $I(\cdot)$ used at inference is ResNet-50 [22] as default setting in experiments; our method is also implemented with ViT-B/16 [11] or ViT-L/14 [11] for further evaluations as shown in Table 2. Note that text and image encoders are derived from the same CLIP model and frozen in the entire pipeline. The dimension of each text feature or image feature is $C = 1024$ when our method is implemented with ResNet-50, while $C = 512$ in the case of ViT-B/16 and $C = 768$ in the case of ViT-L/14.

Learning style word vectors. We follow prompt learning methods [70, 71] when learning the word vectors. Using a zero-mean Gaussian distribution with 0.02 standard deviation, we randomly initialize K style word vectors $\{\mathbf{s}_i\}_{i=1}^K$, where $K = 80$. The dimension of each style word vector is $D = 512$ when the proposed method is implemented with ResNet-50 [22] or ViT-B/16 [11], while $D = 768$ in the case of ViT-L/14 [11]. Each i-th style word vector $\mathbf{s}_i$ is trained by the prompt loss $\mathcal{L}_{\text{prompt}}$ for $L = 100$ training iterations using the SGD optimizer with 0.002 learning rate and 0.9 momentum. The number of classes N is pre-defined by each target task definition, *e.g.*, $N = 345$ for DomainNet [48].

Training a linear classifier. The classifier is trained for 50 epochs using the SGD optimizer with 0.005 learning rate, 0.9 momentum, and a batch size of 128. In ArcFace [8] loss, its scaling factor is set to 5 with 0.5 angular margin.

Inference. Input images are pre-processed in the same way with the CLIP model; resized to 224×224 and normalized.

[1] https://github.com/openai/CLIP

Method	Configuration: Source Domain	Configuration: Domain Description	Accuracy (%): PACS	VLCS	OfficeHome	DomainNet	Avg.
ResNet-50 [22] with pre-trained weights on ImageNet [6]							
DANN [19]	✓	–	83.6±0.4	78.6±0.4	65.9±0.6	38.3±0.1	66.6
RSC [25]	✓	–	85.2±0.9	77.1±0.5	65.5±0.9	38.9±0.5	66.7
MLDG [35]	✓	–	84.9±1.0	77.2±0.4	66.8±0.6	41.2±0.1	67.5
SagNet [46]	✓	–	**86.3**±0.2	77.8±0.5	68.1±0.1	40.3±0.1	68.1
SelfReg [28]	✓	–	85.6±0.4	77.8±0.9	67.9±0.7	42.8±0.0	68.5
GVRT [44]	✓	–	85.1±0.3	**79.0**±0.2	70.1±0.1	44.1±0.1	69.6
MIRO [5]	✓	–	85.4±0.4	**79.0**±0.0	**70.5**±0.4	**44.3**±0.2	**69.8**
ResNet-50 [22] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	90.6±0.0	76.0±0.0	68.6±0.0	45.6±0.0	70.2
CAD [53]	✓	–	90.0±0.6	81.2±0.6	70.5±0.3	45.5±2.1	71.8
ZS-CLIP (PC) [50]	–	✓	90.7±0.0	80.1±0.0	72.0±0.0	46.2±0.0	72.3
PromptStyler	–	–	**93.2**±0.0	**82.3**±0.1	**73.6**±0.1	**49.5**±0.0	**74.7**
ViT-B / 16 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	95.7±0.0	76.4±0.0	79.9±0.0	57.8±0.0	77.5
MIRO [5]	✓	–	95.6	82.2	82.5	54.0	78.6
ZS-CLIP (PC) [50]	–	✓	96.1±0.0	82.4±0.0	82.3±0.0	57.7±0.0	79.6
PromptStyler	–	–	**97.2**±0.1	**82.9**±0.0	**83.6**±0.0	**59.4**±0.0	**80.8**
ViT-L / 14 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	97.6±0.0	77.5±0.0	85.9±0.0	63.3±0.0	81.1
ZS-CLIP (PC) [50]	–	✓	98.5±0.0	**82.4**±0.0	86.9±0.0	64.0±0.0	83.0
PromptStyler	–	–	**98.6**±0.0	**82.4**±0.2	**89.1**±0.0	**65.5**±0.0	**83.9**

Table 2: Comparison with the state-of-the-art domain generalization methods. ZS-CLIP (C) denotes zero-shot CLIP using "[class]" as its text prompt, and ZS-CLIP (PC) indicates zero-shot CLIP using "a photo of a [class]" as its text prompt. Note that PromptStyler does not exploit any source domain data and domain descriptions.

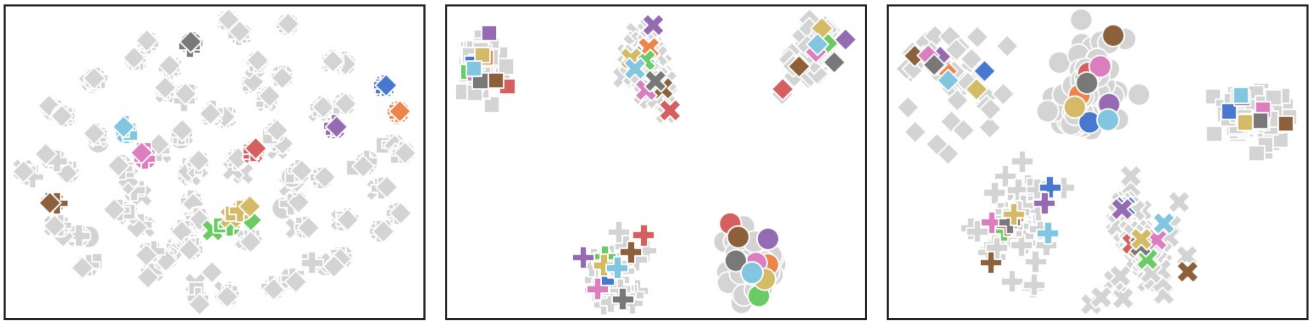
4.3. Evaluations

Main results. PromptStyler achieves the state of the art in every evaluation on PACS [34], VLCS [15], OfficeHome [60] and DomainNet [48] as shown in Table 2. Note that all existing methods utilize source domain data except for zero-shot CLIP [50] in Table 2. Compared with zero-shot CLIP which 域无关的 generates each text feature using a domain-agnostic prompt ("[class]"), PromptStyler largely outperforms its records in all evaluations. Our method also shows higher accuracy compared with zero-shot CLIP which produces each text feature using a domain-specific prompt ("a photo of a [class]"), even though we do not exploit any domain descriptions. These results confirm that the proposed method effectively improves the generalization capability of the chosen pre-trained model, *i.e.*, CLIP, without using any images by simulating various distribution shifts via prompts in its latent space.

Computational evaluations. In Table 3, we compare our PromptStyler and zero-shot CLIP [50] in terms of the number of parameters and inference speed; the inference speed was measured using a single RTX 3090 GPU with a batch size of 1. Note that we do not exploit a text encoder at inference time, which makes our model ~2.6× smaller and ~243× faster compared with CLIP. Regarding the inference speed, the proposed model is about 45× faster for the target task OfficeHome [60] (65 classes) and it is about 243× faster for the target task DomainNet [48] (345 classes).

Method	Inference Module: Image Encoder	Inference Module: Text Encoder	# Params	FPS
OfficeHome (65 classes)				
ZS-CLIP [50]	✓	✓	102.0M	1.6
PromptStyler	✓	–	**38.4M**	**72.9**
DomainNet (345 classes)				
ZS-CLIP [50]	✓	✓	102.0M	0.3
PromptStyler	✓	–	**38.7M**	**72.9**

Table 3: The number of parameters and inference speed on OfficeHome [60] and DomainNet [48] using ResNet-50 [22] as an image encoder. Note that CLIP [50] text encoder needs to generate text features as many as the number of classes.



(a) $\mathcal{L}_{\text{style}}$ (b) $\mathcal{L}_{\text{content}}$ (c) $\mathcal{L}_{\text{style}} + \mathcal{L}_{\text{content}}$

Figure 4: t-SNE [58] visualization results for the target task VLCS [15] (5 classes) using synthesized style-content features. We visualize such features obtained from the learned 80 style word vectors $\{\mathbf{s}_i\}_{i=1}^{80}$ and all the 5 classes (bird, car, chair, dog, person). Different colors denote features obtained from different style word vectors, and different shapes indicate features obtained from different class names. We only colorize features from the first 10 styles $\{\mathbf{s}_i\}_{i=1}^{10}$. Combining the style diversity loss $\mathcal{L}_{\text{style}}$ and content consistency loss $\mathcal{L}_{\text{content}}$ leads to diverse styles while preserving content information.

Figure 5: Text-to-Image synthesis results using style-content features (from "a $\boldsymbol{S_*}$ style of a **cat**") with 6 different style word vectors. By leveraging the proposed method, we could learn a variety of styles while not distorting content information.

		Accuracy (%)				
$\mathcal{L}_{\text{style}}$	$\mathcal{L}_{\text{content}}$	PACS	VLCS	OfficeHome	DomainNet	Avg.
–	–	92.6	78.3	72.2	48.0	72.8
✓	–	92.3	80.9	71.5	48.2	73.2
–	✓	92.8	80.5	72.4	48.6	73.6
✓	✓	**93.2**	**82.3**	**73.6**	**49.5**	**74.7**

Table 4: Ablation study on the style diversity loss $\mathcal{L}_{\text{style}}$ and content consistency loss $\mathcal{L}_{\text{content}}$ used in the prompt loss.

	Accuracy (%)				
$\mathcal{L}_{\text{class}}$	PACS	VLCS	OfficeHome	DomainNet	Avg.
Softmax	92.5	81.2	72.3	48.6	73.7
ArcFace	**93.2**	**82.3**	**73.6**	**49.5**	**74.7**

Table 5: Ablation study on the classification loss $\mathcal{L}_{\text{class}}$ used for training a linear classifier in the proposed framework.

t-SNE visualization results. In Figure 4, we qualitatively evaluate style-content features synthesized for the target task VLCS [15] (5 classes) using t-SNE [58] visualization. As shown in Figure 4(c), PromptStyler generates a variety of styles while not distorting content information; style-content features obtained from the same class name share similar semantics with diverse variations. This result confirms that we could effectively simulate various distribution shifts in the latent space of a large-scale vision-language model by synthesizing diverse styles via learnable style word vectors.
Text-to-Image synthesis results. In Figure 5, we visualize style-content features (from "a $\boldsymbol{S_*}$ style of a **cat**") via diffusers library.[2] These results are obtained with 6 different style word vectors, where the word vectors are learned for the target task DomainNet [48] using ViT-L/14 [11] model.

[2] https://github.com/huggingface/diffusers

4.4. More analyses

Ablation study on the prompt loss. In Table 4, we evaluate the effects of $\mathcal{L}_{\text{style}}$ and $\mathcal{L}_{\text{content}}$ in $\mathcal{L}_{\text{prompt}}$ used for learning style words. Interestingly, our method also achieves state-of-the-art results even without using these losses, *i.e.*, the proposed framework (Fig. 3) is substantially effective by itself. Note that randomly initialized style word vectors are already diverse, and CLIP [50] is already good at extracting correct content information from a style-content prompt even without training the word vectors using $\mathcal{L}_{\text{content}}$. When we learn style word vectors using $\mathcal{L}_{\text{style}}$ without $\mathcal{L}_{\text{content}}$, style-content features obtained from different class names share more similar features than those from the same class name (Fig. 4(a)). On the other hand, using $\mathcal{L}_{\text{content}}$ without $\mathcal{L}_{\text{style}}$ leads to less diverse style-content features (Fig. 4(b)). When incorporating both losses, we could generate diverse styles while not distorting content information (Fig. 4(c)).

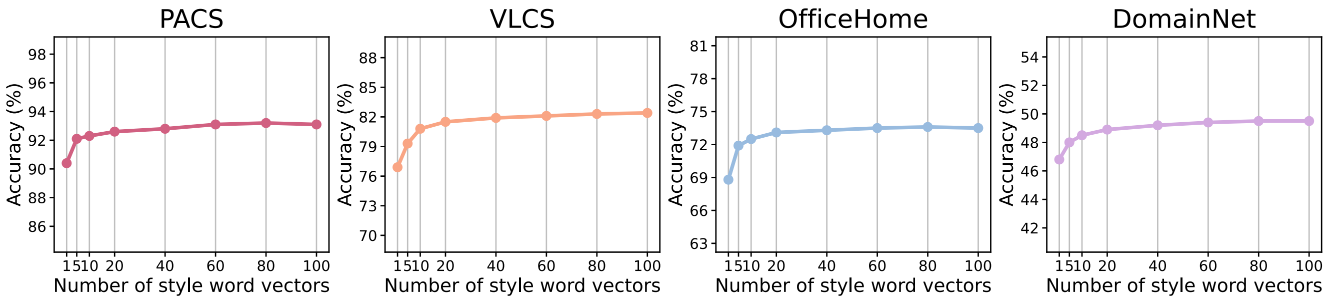


Figure 6: Top-1 classification accuracy on the PACS [34], VLCS [15], OfficeHome [60] and DomainNet [48] datasets with regard to the number of learnable style word vectors K.

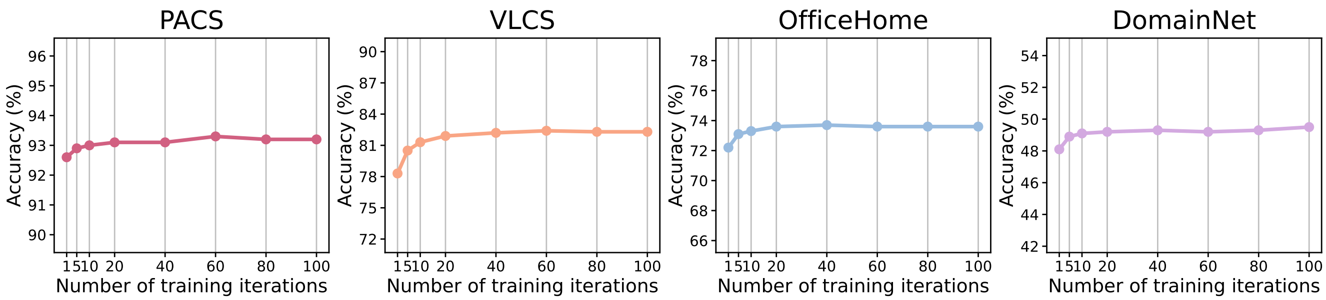


Figure 7: Top-1 classification accuracy on the PACS [34], VLCS [15], OfficeHome [60] and DomainNet [48] datasets with regard to the number of training iterations L for learning each style word vector $\mathbf{s}_i$.

Method	Configuration: Source Domain	Configuration: Domain Description	Accuracy (%): Terra Incognita
ResNet-50 [22] with pre-trained weights on ImageNet [6]			
SelfReg [28]	✓	–	47.0±0.3
GVRT [44]	✓	–	**48.0**±0.2
ResNet-50 [22] with pre-trained weights from CLIP [50]			
ZS-CLIP (C) [50]	–	–	19.5±0.0
ZS-CLIP (PC) [50]	–	✓	23.8±0.0
PromptStyler	–	–	**30.5**±0.8

Table 6: Unsatisfactory results obtained from CLIP [50] without using source domain data from Terra Incognita [1].

Ablation study on the classification loss. In Table 5, we evaluate the effects of the original Softmax loss and the angular Softmax loss (*i.e.*, ArcFace [8]). PromptStyler also achieves the state of the art using the original one, which validates that the performance improvement of our method mainly comes from the proposed framework (Fig. 3). Note that the angular Softmax loss further improves its accuracy by leveraging the hyperspherical joint vision-language space.

Effect of the number of styles. We evaluate our method with regard to the number of style word vectors K as shown in Figure 6. Interestingly, our PromptStyler outperforms CLIP [50] using just 5 styles. This evaluation shows that 20 style word vectors are enough to achieve decent results.

Effect of the number of iterations. We evaluate our method with regard to the number of training iterations L for learning each style word vector as shown in Figure 7. This evaluation shows that 20 iterations are enough to achieve decent results.

5. Limitation

The performance of our method depends on the quality of the joint vision-language space constructed by the chosen vision-language model. For example, although PromptStyler largely outperforms its base model (*i.e.*, CLIP [50]) in all evaluations, our method shows lower accuracy on the Terra Incognita dataset [1] compared with other methods which utilize several images from the dataset as shown in Table 6. The main reason for this might be due to the low accuracy of CLIP on the dataset. Nevertheless, given that our method consistently outperforms its base model in every evaluation, this limitation could be alleviated with the development of large-scale vision-language models.

6. Conclusion

We have presented a novel method that synthesizes a variety of styles in a joint vision-language space via learnable style words without exploiting any images to deal with source-free domain generalization. PromptStyler simulates various distribution shifts in the latent space of a large-scale pre-trained model, which could effectively improve its generalization capability. The proposed method achieves state-of-the-art results without using any source domain data on multiple domain generalization benchmarks. We hope that future work could apply our method to other tasks using different large-scale vision-language models.

Acknowledgment. This work was supported by the Agency for Defense Development grant funded by the Korean government.

References

[1] Sara Beery, Grant van Horn, and Pietro Perona. Recognition in Terra Incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 8, 13

[2] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of Representations for Domain Adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, 2006. 1

[3] Fabio Maria Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain Generalization by Solving Jigsaw Puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 3

[4] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. SWAD: Domain Generalization by Seeking Flat Minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3

[5] Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. Domain Generalization by Mutual-Information Regularization with Pre-trained Models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 1, 3, 6

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6, 8, 13, 14, 15

[7] Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, and Stefanos Zafeiriou. Sub-center ArcFace: Boosting Face Recognition by Large-Scale Noisy Web Faces. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 5

[8] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5, 8, 16

[9] Jiankang Deng and Stefanos Zafeririou. ArcFace for Disguised Face Recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 5

[10] Yu Ding, Lei Wang, Bin Liang, Shuming Liang, Yang Wang, and Fang Chen. Domain Generalization by Learning and Removing Domain-specific Features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*, 2021. 5, 6, 7, 12, 14, 15

[12] Qi Dou, Daniel C. Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain Generalization via Model-Agnostic Learning of Semantic Features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[13] Lisa Dunlap, Clara Mohri, Devin Guillory, Han Zhang, Trevor Darrell, Joseph E. Gonzalez, Aditi Raghunathan, and Anja Rohrbach. Using Language to Extend to Unseen Domains. In *International Conference on Learning Representations (ICLR)*, 2023. 3

[14] Xinjie Fan, Qifei Wang, Junjie Ke, Feng Yang, Boqing Gong, and Mingyuan Zhou. Adversarially Adaptive Normalization for Single Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[15] Chen Fang, Ye Xu, and Daniel N. Rockmore. 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 (ICCV)*, 2013. 2, 5, 6, 7, 8, 13, 14, 16

[16] Ahmed Frikha, Haokun Chen, Denis Krompaß, Thomas Runkler, and Volker Tresp. Towards Data-Free Domain Generalization. In *Asian Conference on Machine Learning (ACML)*, 2022. 3

[17] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion. In *International Conference on Learning Representations (ICLR)*, 2023. 2, 3

[18] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. StyleGAN-NADA: CLIP-Guided Domain Adaptation of Image Generators. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 2022. 3

[19] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. In *Journal of Machine Learning Research (JMLR)*, 2016. 6

[20] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. CLIP-Adapter: Better Vision-Language Models with Feature Adapters. *arXiv preprint arXiv:2110.04544*, 2021. 3

[21] Ishaan Gulrajani and David Lopez-Paz. In Search of Lost Domain Generalization. In *International Conference on Learning Representations (ICLR)*, 2021. 1, 3, 5

[22] He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5, 6, 8, 12, 13, 14, 15

[23] Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 3

[24] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In *International Conference on Machine Learning (ICML)*, 2018. 1

[25] Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. Self-Challenging Improves Cross-Domain Generalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 6

[26] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In *International Conference on Machine Learning (ICML)*, 2021. 2, 3, 12

[27] Juwon Kang, Sohyun Lee, Namyup Kim, and Suha Kwak. Style Neophile: Constantly Seeking Novel Styles for Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[28] Daehee Kim, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. SelfReg: Self-supervised Contrastive Regularization for Domain Generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 6, 8, 13, 14, 15

[29] Donghyun Kim, Kaihong Wang, Stan Sclaroff, and Kate Saenko. A Broad Study of Pre-training for Domain Generalization and Adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 1, 3

[30] Dimitrios Kollias and Stefanos Zafeiriou. Expression, Affect, Action Unit Recognition: Aff-Wild2, Multi-Task Learning and ArcFace. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2019. 5

[31] Gihyun Kwon and Jong Chul Ye. CLIPstyler: Image Style Transfer with a Single Text Condition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[32] Sohyun Lee, Taeyoung Son, and Suha Kwak. FIFO: Learning Fog-invariant Features for Foggy Scene Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[33] Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical Fine-Tuning Improves Adaptation to Distribution Shifts. In *International Conference on Learning Representations (ICLR)*, 2023. 1

[34] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, Broader and Artier Domain Generalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 2, 5, 6, 8, 13, 14, 16

[35] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to Generalize: Meta-Learning for Domain Generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 1, 3, 6

[36] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. Episodic Training for Domain Generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3

[37] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Domain Generalization With Adversarial Feature Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 3

[38] Lei Li, Ke Gao, Juan Cao, Ziyao Huang, Yepeng Weng, Xiaoyue Mi, Zhengze Yu, Xiaoya Li, and Boyang xia. Progressive Domain Expansion Network for Single Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[39] Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Zou. Mind the Gap: Understanding the Modality Gap in Multi-modal Contrastive Representation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 16

[40] Boxiao Liu, Guanglu Song, Manyuan Zhang, Haihang You, and Yu Liu. Switchable K-Class Hyperplanes for Noise-Robust Representation Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 5

[41] Yuning Lu, Jianzhuang Liu, Yonggang Zhang, Yajing Liu, and Xinmei Tian. Prompt Distribution Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[42] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain Generalization using Causal Matching. In *International Conference on Machine Learning (ICML)*, 2021. 3

[43] Toshihiko Matsuura and Tatsuya Harada. Domain Generalization Using a Mixture of Multiple Latent Domains. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 3

[44] Seonwoo Min, Nokyung Park, Siwon Kim, Seunghyun Park, and Jinkyu Kim. Grounding Visual Representations with Texts for Domain Generalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 3, 6, 8, 13, 14, 15

[45] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain Generalization via Invariant Feature Representation. In *International Conference on Machine Learning (ICML)*, 2013. 3

[46] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing Domain Gap by Reducing Style Bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6

[47] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3

[48] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment Matching for Multi-Source Domain Adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2, 5, 6, 7, 8, 13, 15, 16

[49] Fengchun Qiao, Long Zhao, and Xi Peng. Learning to Learn Single Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[50] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 2021. 2, 3, 4, 5, 6, 7, 8, 12, 13, 14, 15, 16

[51] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant Gradient Variances for Out-of-Distribution

Generalization. In *International Conference on Machine Learning (ICML)*, 2022. 3

[52] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet Classifiers Generalize to ImageNet? In *International Conference on Machine Learning (ICML)*, 2019. 1, 3

[53] Yangjun Ruan, Yann Dubois, and Chris J. Maddison. Optimal Representations for Covariate Shift. In *International Conference on Learning Representations (ICLR)*, 2022. 3, 6

[54] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-Supervised Domain Adaptation via Minimax Entropy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 1

[55] Seonguk Seo, Yumin Suh, Dongwan Kim, Geeho Kim, Jongwoo Han, and Bohyung Han. Learning to Optimize Domain Specific Normalization for Domain Generalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[56] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of Frustratingly Easy Domain Adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2016. 1

[57] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial Discriminative Domain Adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[58] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. In *Journal of Machine Learning Research (JMLR)*, 2008. 7

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 5, 12

[60] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep Hashing Network for Unsupervised Domain Adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 5, 6, 8, 12, 13, 15, 16

[61] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S. Yu. Generalizing to Unseen Domains: A Survey on Domain Generalization. In *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2021. 1, 3

[62] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to Diversify for Single Domain Generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3

[63] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A Fourier-based Framework for Domain Generalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[64] Jinyu Yang, Jiali Duan, Son Tran, Yi Xu, Sampath Chanda, Liqun Chen, Belinda Zeng, Trishul Chilimbi, and Junzhou Huang. Vision-Language Pre-Training with Triple Contrastive Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3, 12

[65] Dingyi Zhang, Yingming Li, and Zhongfei Zhang. Deep metric learning with spherical embedding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 5

[66] Renrui Zhang, Zhang Wei, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-Adapter: Training-free Adaption of CLIP for Few-shot Classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 3

[67] Yuhui Zhang, Jeff Z. HaoChen, Shih-Cheng Huang, Kuan-Chieh Wang, James Zou, and Serena Yeung. Diagnosing and Rectifying Vision Models using Language. In *International Conference on Learning Representations (ICLR)*, 2023. 2, 3, 16

[68] Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoffrey J. Gordon. On Learning Invariant Representation for Domain Adaptation. In *International Conference on Machine Learning (ICML)*, 2019. 1

[69] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain Generalization: A Survey. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022. 1, 3

[70] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional Prompt Learning for Vision-Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 5, 16

[71] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to Prompt for Vision-Language Models. In *International Journal of Computer Vision (IJCV)*, 2022. 3, 5, 16

[72] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep Domain-Adversarial Image Generation for Domain Generalisation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 3

[73] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to Generate Novel Domains for Domain Generalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[74] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain Generalization with MixStyle. In *International Conference on Learning Representations (ICLR)*, 2021. 1, 2, 3

PromptStyler: Prompt-driven Style Generation for Source-free Domain Generalization
— Supplementary Material —

Junhyeong Cho[1] Gilhyun Nam[1] Sungyeon Kim[2] Hunmin Yang[1,3] Suha Kwak[2]
[1]ADD [2]POSTECH [3]KAIST
https://PromptStyler.github.io

In this supplementary material, we provide more method details (Section A), analyses on Terra Incognita (Section B), evaluation results (Section C) and discussion (Section D).

A. Method Details

This section provides more details of the chosen vision-language model (Section A.1) and design choices for learning style word vectors (Section A.2).

A.1. Large-scale vision-language model

We choose CLIP [50] as our pre-trained vision-language model which is a large-scale model trained with 400 million image-text pairs. Note that the proposed method is broadly applicable to the CLIP-like vision-language models [26, 64] which also construct hyperspherical joint vision-language spaces using contrastive learning methods. Given a batch of image-text pairs, such models jointly train an image encoder and a text encoder considering similarity scores obtained from image-text pairings.

Joint vision-language training. Suppose there is a batch of M image-text pairs. Among all possible $M \times M$ pairings, the matched M pairs are the positive pairs and the other $M^2 - M$ pairs are the negative pairs. CLIP [50] is trained to maximize cosine similarities of image and text features from the positive M pairs while minimizing the similarities of such features from the negative $M^2 - M$ pairs.

Image encoder. CLIP [50] utilizes ResNet [22] or ViT [11] as its image encoder. Given an input image, the image encoder extracts its image feature. After that, the image feature is mapped to a hyperspherical joint vision-language space by ℓ_2 normalization.

Text encoder. CLIP [50] utilizes Transformer [59] as its text encoder. Given an input text prompt, it is converted to word vectors via a tokenization process and a word lookup procedure. Using these word vectors, the text encoder generates a text feature which is then mapped to a hyperspherical joint vision-language space by ℓ_2 normalization.

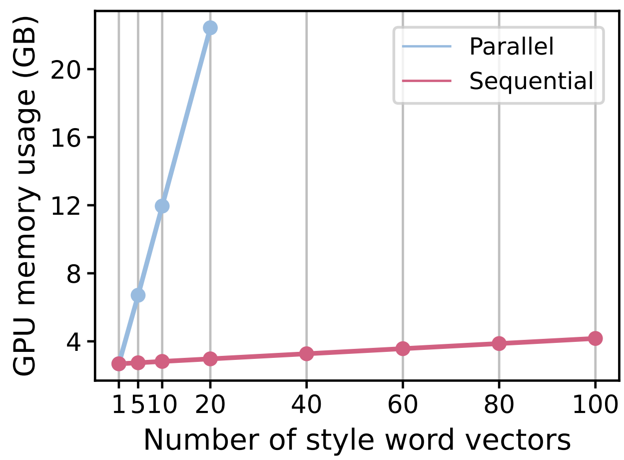


Figure A1: GPU memory usage when learning K style word vectors for the target task OfficeHome [60] (65 classes) with respect to the design choices, *Sequential* or *Parallel*.

Zero-shot inference. At inference time, zero-shot CLIP [50] synthesizes classifier weights via the text encoder using N class names pre-defined in the target task. Given an input image, the image encoder extracts its image feature and the text encoder produces N text features using the N class names. Then, it computes cosine similarity scores between the image feature and text features, and selects the class name which results in the highest similarity score as its classification output.

A.2. Empirical justification of our design choice

As described in Section 3.1 of the main paper, there are two possible design choices for learning K style word vectors: (1) learning each style word vector $\mathbf{s}_i$ in a sequential manner, or (2) learning all style word vectors $\{\mathbf{s}_i\}_{i=1}^K$ in a parallel manner. We choose the former mainly due to its much less memory overhead. As shown in Figure A1, we could sequentially learn ~100 style word vectors with ~4.2 GB memory usage. However, it is not possible to learn more than 21 style word vectors in a parallel manner using a single

(a) Dog (b) Cat (c) Squirrel

Figure B1: Several examples from the Terra Incognita [1] dataset. We visualize class entities using red bounding boxes, since they are not easily recognizable due to their small sizes and complex background scenes.

Method	Configuration: Source Domain	Configuration: Domain Description	Accuracy (%): Location100	Location38	Location43	Location46	Avg.
ResNet-50 [22] with pre-trained weights on ImageNet [6]							
SelfReg [28]	✓	–	48.8±0.9	41.3±1.8	57.3±0.7	**40.6**±0.9	47.0
GVRT [44]	✓	–	**53.9**±1.3	**41.8**±1.2	**58.2**±0.9	38.0±0.6	**48.0**
ResNet-50 [22] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	8.4±0.0	13.7±0.0	32.5±0.0	23.3±0.0	19.5
ZS-CLIP (PC) [50]	–	✓	9.9±0.0	28.3±0.0	32.9±0.0	24.0±0.0	23.8
PromptStyler	–	–	**13.8**±1.7	**39.8**±1.3	**38.0**±0.4	**30.3**±0.3	**30.5**

Table B1: Top-1 classification accuracy on the Terra Incognita [1] dataset. Compared with existing domain generalization methods which utilize source domain data, zero-shot methods using CLIP [50] show unsatisfactory results on this dataset.

RTX 3090 GPU (24 GB Memory) due to its large memory overhead. In detail, learning 20 and 21 style word vectors takes 22.4 GB and 23.5 GB, respectively. The large memory overhead caused by the parallel learning design substantially limits the number of learnable style word vectors.

To be specific, PromptStyler with the parallel learning design needs to generate K style features, KN style-content features, and N content features for learning K style word vectors at the same time; these features are used to compute the style diversity loss $\mathcal{L}_{\text{style}}$ and the content consistency loss $\mathcal{L}_{\text{content}}$ for learning all the style word vectors in a parallel manner. Note that the large memory overhead is mainly caused by the KN style-content features. Suppose we want to learn 80 style word vectors for the target task OfficeHome [60] (65 classes). Then, we need to synthesize $5200 (= 80 \times 65)$ style-content features. Even worse, we need to generate $27600 (= 80 \times 345)$ style-content features for the target task DomainNet [48] (345 classes). On the other hand, PromptStyler with the sequential learning design only requires i style features, N style-content features, and N content features for learning i-th style word vector, where $1 \leq i \leq K$. For scalability, we chose the sequential learning design since it could handle a lot of learnable style word vectors and numerous classes in the target task.

B. Analyses on Terra Incognita

As described in Section 5 of the main paper, the quality of the latent space constructed by a large-scale pre-trained model significantly affects the effectiveness of PromptStyler. To be specific, the proposed method depends on the quality of the joint vision-language space constructed by CLIP [50]. Although our method achieves state-of-the-art results on PACS [34], VLCS [15], OfficeHome [60], and DomainNet [48], its performance on Terra Incognita [1] is not satisfactory. This section provides more analyses on the dataset.

Table B1 shows that PromptStyler outperforms zero-shot CLIP [50] for all domains in the Terra Incognita dataset [1]. However, its accuracy on this dataset is lower compared with existing domain generalization methods [28,44] which utilize several images from the dataset as their source domain data. This unsatisfactory result might be due to the low accuracy of CLIP on the dataset. We suspect that images in the Terra Incognita dataset (Fig. B1) might be significantly different from the domains that CLIP has observed. The distribution shifts between CLIP training dataset and the Terra Incognita dataset might be extreme, and thus such distribution shifts could not be entirely covered by our method which exploits CLIP latent space. We hope this issue could be alleviated with the development of large-scale models.

Method	Configuration: Source Domain	Configuration: Domain Description	Accuracy (%): Art Painting	Cartoon	Photo	Sketch	Avg.
ResNet-50 [22] with pre-trained weights on ImageNet [6]							
GVRT [44]	✓	–	**87.9**±0.3	78.4±1.0	**98.2**±0.1	75.7±0.4	85.1
SelfReg [28]	✓	–	**87.9**±1.0	**79.4**±1.4	96.8±0.7	**78.3**±1.2	**85.6**
ResNet-50 [22] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	88.9±0.0	94.4±0.0	99.3±0.0	79.8±0.0	90.6
ZS-CLIP (PC) [50]	–	✓	90.8±0.0	93.3±0.0	**99.4**±0.0	79.3±0.0	90.7
PromptStyler	–	–	**93.7**±0.1	**94.7**±0.2	**99.4**±0.0	**84.9**±0.1	**93.2**
ViT-B/16 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	96.4±0.0	98.9±0.0	**99.9**±0.0	87.7±0.0	95.7
ZS-CLIP (PC) [50]	–	✓	97.2±0.0	**99.1**±0.0	**99.9**±0.0	88.2±0.0	96.1
PromptStyler	–	–	**97.6**±0.1	**99.1**±0.1	**99.9**±0.0	**92.3**±0.3	**97.2**
ViT-L/14 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	97.2±0.0	99.5±0.0	99.9±0.0	93.8±0.0	97.6
ZS-CLIP (PC) [50]	–	✓	99.0±0.0	**99.7**±0.0	99.9±0.0	**95.5**±0.0	98.5
PromptStyler	–	–	**99.1**±0.0	**99.7**±0.0	**100.0**±0.0	**95.5**±0.1	**98.6**

Table C1: Comparison with state-of-the-art domain generalization methods in terms of per-domain top-1 classification accuracy on PACS [34]. We repeat each experiment using three different seeds, and report average accuracies with standard errors. ZS-CLIP (C) denotes zero-shot CLIP using "[class]" as its text prompt, and ZS-CLIP (PC) indicates zero-shot CLIP using "a photo of a [class]" as its text prompt. Note that PromptStyler does not use any source domain data and domain descriptions.

Method	Configuration: Source Domain	Configuration: Domain Description	Accuracy (%): Caltech	LabelMe	SUN09	VOC2007	Avg.
ResNet-50 [22] with pre-trained weights on ImageNet [6]							
SelfReg [28]	✓	–	96.7±0.4	**65.2**±1.2	73.1±1.3	76.2±0.7	77.8
GVRT [44]	✓	–	**98.8**±0.1	64.0±0.3	**75.2**±0.5	**77.9**±1.0	**79.0**
ResNet-50 [22] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	99.2±0.0	62.4±0.0	69.0±0.0	73.5±0.0	76.0
ZS-CLIP (PC) [50]	–	✓	99.4±0.0	65.0±0.0	71.7±0.0	84.2±0.0	80.1
PromptStyler	–	–	**99.5**±0.0	**71.2**±0.2	**72.0**±0.0	**86.5**±0.3	**82.3**
ViT-B/16 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	99.7±0.0	61.8±0.0	70.1±0.0	73.9±0.0	76.4
ZS-CLIP (PC) [50]	–	✓	**99.9**±0.0	68.9±0.0	**74.8**±0.0	85.9±0.0	82.4
PromptStyler	–	–	**99.9**±0.0	**71.5**±0.3	73.9±0.2	**86.3**±0.1	**82.9**
ViT-L/14 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	**99.9**±0.0	59.3±0.0	71.0±0.0	79.9±0.0	77.5
ZS-CLIP (PC) [50]	–	✓	**99.9**±0.0	70.9±0.0	**72.9**±0.0	86.0±0.0	**82.4**
PromptStyler	–	–	**99.9**±0.0	**71.1**±0.7	71.8±1.0	**86.8**±0.0	**82.4**

Table C2: Comparison with state-of-the-art domain generalization methods in terms of per-domain top-1 classification accuracy on VLCS [15]. We repeat each experiment using three different seeds, and report average accuracies with standard errors. ZS-CLIP (C) denotes zero-shot CLIP using "[class]" as its text prompt, and ZS-CLIP (PC) indicates zero-shot CLIP using "a photo of a [class]" as its text prompt. Note that PromptStyler does not use any source domain data and domain descriptions.

Method	Configuration: Source Domain	Configuration: Domain Description	Art	Clipart	Product	Real World	Avg.
ResNet-50 [22] with pre-trained weights on ImageNet [6]							
SelfReg [28]	✓	–	63.6$\pm$0.1.4	53.1$\pm$1.0	76.9$\pm$0.4	78.1$\pm$0.4	67.9
GVRT [44]	✓	–	**66.3**$\pm$0.1	**55.8**$\pm$0.4	**78.2**$\pm$0.4	**80.4**$\pm$0.2	**70.1**
ResNet-50 [22] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	69.9$\pm$0.0	46.8$\pm$0.0	77.7$\pm$0.0	79.8$\pm$0.0	68.6
ZS-CLIP (PC) [50]	–	✓	71.7$\pm$0.0	52.0$\pm$0.0	81.6$\pm$0.0	82.6$\pm$0.0	72.0
PromptStyler	–	–	**73.4**$\pm$0.1	**52.4**$\pm$0.2	**84.3**$\pm$0.1	**84.1**$\pm$0.1	**73.6**
ViT-B / 16 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	80.7$\pm$0.0	64.6$\pm$0.0	86.3$\pm$0.0	88.0$\pm$0.0	79.9
ZS-CLIP (PC) [50]	–	✓	82.7$\pm$0.0	67.6$\pm$0.0	89.2$\pm$0.0	89.7$\pm$0.0	82.3
PromptStyler	–	–	**83.8**$\pm$0.1	**68.2**$\pm$0.0	**91.6**$\pm$0.1	**90.7**$\pm$0.1	**83.6**
ViT-L / 14 [11] with pre-trained weights from CLIP [50]							
ZS-CLIP (C) [50]	–	–	86.2$\pm$0.0	73.3$\pm$0.0	92.0$\pm$0.0	92.2$\pm$0.0	85.9
ZS-CLIP (PC) [50]	–	✓	87.2$\pm$0.0	73.8$\pm$0.0	93.0$\pm$0.0	93.4$\pm$0.0	86.9
PromptStyler	–	–	**89.1**$\pm$0.1	**77.6**$\pm$0.1	**94.8**$\pm$0.1	**94.8**$\pm$0.0	**89.1**

Table C3: Comparison with state-of-the-art domain generalization methods in terms of per-domain top-1 classification accuracy on OfficeHome [60]. We repeat each experiment using three different seeds, and report average accuracies with standard errors. ZS-CLIP (C) denotes zero-shot CLIP using "[class]" as its text prompt, and ZS-CLIP (PC) indicates zero-shot CLIP using "a photo of a [class]" as its text prompt. Note that PromptStyler does not use any source domain data and domain descriptions.

Method	Configuration: Source Domain	Configuration: Domain Description	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	Avg.
ResNet-50 [22] with pre-trained weights on ImageNet [6]									
SelfReg [28]	✓	–	60.7$\pm$0.1	**21.6**$\pm$0.1	49.4$\pm$0.2	12.7$\pm$0.1	60.7$\pm$0.1	51.7$\pm$0.1	42.8
GVRT [44]	✓	–	**62.4**$\pm$0.4	21.0$\pm$0.0	**50.5**$\pm$0.4	**13.8**$\pm$0.3	**64.6**$\pm$0.4	**52.4**$\pm$0.2	**44.1**
ResNet-50 [22] with pre-trained weights from CLIP [50]									
ZS-CLIP (C) [50]	–	–	53.1$\pm$0.0	39.2$\pm$0.0	52.7$\pm$0.0	**6.3**$\pm$0.0	75.2$\pm$0.0	47.1$\pm$0.0	45.6
ZS-CLIP (PC) [50]	–	✓	53.6$\pm$0.0	39.6$\pm$0.0	53.4$\pm$0.0	5.9$\pm$0.0	76.6$\pm$0.0	48.0$\pm$0.0	46.2
PromptStyler	–	–	**57.9**$\pm$0.0	**44.3**$\pm$0.0	**57.3**$\pm$0.0	6.1$\pm$0.1	**79.5**$\pm$0.0	**51.7**$\pm$0.0	**49.5**
ViT-B / 16 [11] with pre-trained weights from CLIP [50]									
ZS-CLIP (C) [50]	–	–	70.7$\pm$0.0	49.1$\pm$0.0	66.4$\pm$0.0	**14.8**$\pm$0.0	82.7$\pm$0.0	63.1$\pm$0.0	57.8
ZS-CLIP (PC) [50]	–	✓	71.0$\pm$0.0	47.7$\pm$0.0	66.2$\pm$0.0	14.0$\pm$0.0	83.7$\pm$0.0	63.5$\pm$0.0	57.7
PromptStyler	–	–	**73.1**$\pm$0.0	**50.9**$\pm$0.0	**68.2**$\pm$0.1	13.3$\pm$0.1	**85.4**$\pm$0.0	**65.3**$\pm$0.0	**59.4**
ViT-L / 14 [11] with pre-trained weights from CLIP [50]									
ZS-CLIP (C) [50]	–	–	78.2$\pm$0.0	53.0$\pm$0.0	70.7$\pm$0.0	21.6$\pm$0.0	86.0$\pm$0.0	70.3$\pm$0.0	63.3
ZS-CLIP (PC) [50]	–	✓	79.2$\pm$0.0	52.4$\pm$0.0	71.3$\pm$0.0	**22.5**$\pm$0.0	86.9$\pm$0.0	71.8$\pm$0.0	64.0
PromptStyler	–	–	**80.7**$\pm$0.0	**55.6**$\pm$0.1	**73.8**$\pm$0.1	21.7$\pm$0.0	**88.2**$\pm$0.0	**73.2**$\pm$0.0	**65.5**

Table C4: Comparison with state-of-the-art domain generalization methods in terms of per-domain top-1 classification accuracy on DomainNet [48]. We repeat each experiment using three different seeds, and report average accuracies with standard errors. ZS-CLIP (C) denotes zero-shot CLIP using "[class]" as its text prompt, and ZS-CLIP (PC) indicates zero-shot CLIP using "a photo of a [class]" as its text prompt. Note that PromptStyler does not use any source domain data and domain descriptions.

Distribution	Accuracy (%)				
	PACS	VLCS	OfficeHome	DomainNet	Avg.
$\mathcal{U}(0.00,\ 0.20)$	93.1	**82.6**	**73.8**	49.2	**74.7**
$\mathcal{N}(0.00,\ 0.20^2)$	93.0	81.0	73.6	**49.5**	74.3
$\mathcal{N}(0.20,\ 0.02^2)$	93.1	82.5	73.5	49.3	74.6
$\mathcal{N}(0.00,\ 0.02^2)$	**93.2**	82.3	73.6	**49.5**	**74.7**

Table C5: Effects of the distributions used for initializing style word vectors. Uniform or Normal distribution is used.

C. Evaluation Results

Per-domain accuracy. As shown in Table C1–C4, we provide per-domain top-1 classification accuracy on domain generalization benchmarks including PACS [34] (4 domains and 7 classes), VLCS [15] (4 domains and 5 classes), OfficeHome [60] (4 domains and 65 classes) and DomainNet [48] (6 domains and 345 classes); each accuracy is obtained by averaging results from experiments repeated using three different random seeds. Interestingly, compared with zero-shot CLIP [50] which leverages a photo domain description ("a photo of a [class]"), our PromptStyler achieves similar or better results on photo domains, *e.g.*, on the VLCS dataset which consists of 4 photo domains. Note that the description has more domain-specific information and more detailed contexts compared with the naïve prompt ("[class]").

Different distributions for initializing style word vectors. Following prompt learning methods [70, 71], we initialized learnable style word vectors using zero-mean Gaussian distribution with 0.02 standard deviation. To measure the effect of the used distribution for the initialization, we also quantitatively evaluate PromptStyler using different distributions for initializing style word vectors. As shown in Table C5, the proposed method also achieves similar results when initializing style word vectors using different distributions.

D. Discussion

PromptStyler aims to improve model's generalization capability by simulating various distribution shifts in the latent space of a large-scale pre-trained model. To achieve this goal, our method leverages a joint vision-language space where text features could effectively represent their relevant image features. It does not mean that image and text features should be perfectly interchangeable in the joint vision-language space; a recent study has demonstrated the modality gap phenomenon of this joint space [39]. However, thanks to the cross-modal transferability in the joint vision-language space [67], the proposed method could still be effective, *i.e.*, we could consider text features as proxies for image features while training a linear classifier (Fig. 3 of the main paper).

When our method is implemented with CLIP [50] and we adopt ArcFace [8] as our classification loss $\mathcal{L}_{\text{class}}$, there is another interesting interpretation of the proposed method. As described in Section A.1, CLIP text encoder synthesizes classifier weights using class names for zero-shot inference and then it computes cosine similarity scores between the classifier weights and input image features. Similarly, our method computes cosine similarity scores between classifier weights of the trained classifier (Fig. 3 of the main paper) and input image features. From this perspective, the proposed method improves the decision boundary of the synthesized classifier used in zero-shot CLIP by generating diverse style-content features and then training a linear classifier using the style-content features. In other words, the trained classifier could be considered as an improved version of the synthesized classifier used in zero-shot CLIP.