Learning Common and Specific Visual Prompts for Domain Generalization

Aodi Li[1], Liansheng Zhuang[1,(✉)], Shuo Fan[1], and Shafei Wang[2]

[1] University of Science and Technology of China, Hefei 230026, China
[2] Peng Cheng Laboratory, Shenzhen 518000, China
aodi8055@mail.ustc.edu.cn, lszhuang@ustc.edu.cn

Abstract. Although fine-tuning a pre-trained large-scale model has become an effective method for domain generalization, domain shifts still issue a huge challenge for successfully transferring models to unseen test domains. In this paper, we study how to effectively adapt pre-trained vision Transformers for domain generalization problems in image classification. To this end, this paper proposes a novel Common-Specific Visual Prompt Tuning (CSVPT) method to transfer large-scale vision Transformer models to unknown test domains. Different from existing methods which learn fixed visual prompts for each task, CSVPT jointly learns domain-common prompts to capture the task context and sample-specific prompts to capture information about data distribution, which are generated for each sample through a trainable prompt-generating module (PGM). Combining the domain-common prompts and the sample-specific prompts, visual prompts learned by CSVPT are conditioned on each input sample rather than fixed once learned, which helps out-of-distribution generalization. Extensive experimental results show the effectiveness of CSVPT, and CSVPT with the backbone ViT-L/14 achieves state-of-the-art (SOTA) performance on five widely used benchmark datasets.

1 Introduction

Though deep learning has achieved remarkable success in many areas [1–4], it relies on the i.i.d. assumption that training and testing data are independent and identically distributed [5]. However, this assumption does not always hold in real applications. When collecting data under different conditions or from different sources, test data is often out of the distributions of training data. The out-of-distribution (OOD) problem significantly degrades the performance of deep models [6]. To tackle this problem, lots of *domain generalization* (DG) methods aim to learn a model from multiple training domains that will generalize well on unseen testing domains [6].

The past few years have witnessed the advance of DG algorithms [6, 7]. Among them, learning feature representations that were invariant across domains [8–12] and decomposing model parameters into shared and domain-specific components [13, 14] are the two most common methods. However, some researchers have revealed that none of the existing DG methods greatly outperform

simple baselines on the diverse DG benchmarks [15]. This is because the training and test distributions are too different to learn domain-invariant features or obtain excellent common-specific decomposition from the training domains alone.

Inspired by the great success of Transformers [16–19], a line of works such as [20–22] turned to large-scale pre-trained models for help. Benefiting from massive labeled and unlabeled data, pre-training models on diverse data can efficiently capture rich prior knowledge and improve OOD generalization [23]. By fine-tuning on specific tasks, the rich knowledge implicitly encoded in the pre-trained models can benefit a variety of downstream tasks. Full fine-tuning is one of the most common practices, which updates all parameters on the downstream tasks. However, storing a separate copy of the whole backbone parameters for each task is an expensive and infeasible proposal, especially for large-scale Transformer models. Moreover, full fine-tuning may distort pre-trained features and thus harm the robustness against distribution shifts [24]. Instead of fine-tuning the pre-trained Transformer itself, visual prompt tuning (VPT) [25] modifies the input to the Transformer. It introduces a small amount of task-specific trainable parameters (namely visual prompts) into the input space while keeping the whole pre-trained backbone frozen. Since only a few parameters (1% of model parameters) requires to be updated, VPT not only greatly reduces computational and storage costs but also prevents overfitting and feature distortion. Although a proper prompt that matches with data distribution could improve the performance, it is difficult to design an appropriate visual prompt for an unknown domain to address the domain shift problem.

In this paper, we investigate how to effectively adapt pre-trained vision Transformer models for domain generalization in image classification. Taking inspiration from VPT [25] and traditional DG methods [13, 14], this paper proposes a novel Common-Specific Visual Prompt Tuning (CSVPT) method to transfer the pre-trained vision Transformer models to unknown testing domains for better OOD generalization. To our best knowledge, it is the first work to design a DG algorithm based on visual prompts. Different from existing prompt-based methods [25, 26] which only learn task-specific prompts, our proposed CSVPT jointly learns domain-common prompts and sample-specific prompts to modify the input to the pre-trained vision Transformer models. The domain-common prompts capture the task context and are fixed once learned, thus easy to overfit the training domains. To generalize the prompts to wider unseen domains within the same task, CSVPT learns a lightweight neural network (namely prompt-generating module, PGM) to generate the sample-specific prompts for each sample so as to capture information about data distribution. Combining the domain-common prompts and the sample-specific prompts, our visual prompts are conditioned on each input sample rather than fixed once learned, which helps out-of-distribution generalization. To validate the effectiveness of our proposed CSVPT method, we perform extensive experiments on five popular datasets of DomainBed [15], including PACS [27], VLCS [28], OfficeHome [29], TerraIncognita [30] and DomainNet [31]. Experimental results demonstrate that CSVPT

consistently performs better on these datasets than the vanilla VPT methods and other DG methods based on pre-trained models. Moreover, CSVPT with ViT-L/14 pre-trained on CLIP [32] achieves new state-of-the-art performances on the five datasets.

In summary, our key contributions are as follows:

- A novel CSVPT method is proposed to efficiently adapt large-scale vision Transformers for the DG problem. Instead of learning fixed visual prompts as VPT does, CSVPT generates the visual prompts conditioned on each input sample, and thus achieves better OOD generalization.
- Extensive experiments on five public datasets demonstrate that our proposed CVSVPT consistently outperforms existing fine-tuning methods for domain generalization in image classification. Especially, CSVPT with the backbone ViT-L/14 achieves new SOTA performance.

2 Related Work

Invariance learning and decomposition learning are two mainstream methods for DG. These conventional DG methods usually fail to learn domain-invariant features or obtain excellent common-specific decomposition from the training distributions alone. To solve this problem, we propose a novel CSVPT method, which is mainly inspired by decomposition learning and pre-trained model-based methods. In this section, we will introduce these related works concisely.

2.1 Decomposition Learning Method

Decomposition learning, which supposes that features, model parameters, or gradients are composed of domain-specific components and domain-common components, is one of the most popular methods for DG. Decomposition learning can be divided into three categories as follows. The first type is feature decomposition learning. Feature decomposition learning tries to disentangle the feature representation into two parts, i.e., domain-specific parts and domain-common parts. Afterward, we can either use the domain-common feature only or combine the domain-common features with augmented domain-specific features for prediction [33]. The second category is predictor decomposition learning, such as CSD [14]. It learns a domain-common predictor (which helps generalization) and a domain-specific predictor (which may harm generalization). The domain-specific predictors are discarded after training and only the common predictor is used for prediction. The last one is gradient decomposition learning. For example, AndMask [34] updates weights only when gradients from different domains point to the same direction, i.e., retaining the domain-common gradient component. Similarly, we assume prompts are composed of domain-common and sample-specific components in this paper. Domain-common components learned from the training data are shared by the test data, but sample-specific components are generated from the input data via a simple linear PGM. Combining

the domain-common and sample-specific prompts, we can obtain more appropriate prompts for samples from unseen testing domains, which helps out-of-distribution generalization.

2.2 Pre-trained Model-based Method

As we all know, full fine-tuning and linear probing are two popular methods when transferring a pre-trained model to a downstream task. All the model parameters are trainable for full fine-tuning, while only the parameters of the last fully connected layer are for linear probing. When training and testing data are independent and identically distributed, full fine-tuning usually outperforms linear probing. However, some researchers [24] pointed out full fine-tuning may distort pre-trained features and underperform out-of-distribution, because the features of in-distribution training data are updated greatly while those of out-of-distribution data change less. Ananya Kumar et al. [24] proposed a two-step approach (linear probing then full fine-tuning) to solve the problem. Besides, some researchers [21] utilized a mutual information regularization with the pre-trained model, called Mutual Information Regularization with Oracle (MIRO), to prevent overfitting and feature distortion. Although MIRO achieved SOTA performance on several DG benchmarks, updating all parameters and calculating the above loss consume a lot of computation and storage resources. Instead of fine-tuning the pre-trained Transformer itself, visual prompt tuning (VPT) [25] modifies the input to the Transformer models. It introduces a small amount of task-specific trainable parameters (namely visual prompts) into the input space while keeping the whole pre-trained backbone frozen. It not only greatly saves the computation and storage resources but also prevents the feature distortion brought by full fine-tuning. Inspired by VPT, we propose a novel visual prompt tuning method named CSVPT for domain generalization.

3 Methodology

We propose a novel CSVPT method to effectively adapt pre-trained vision Transformers for domain generalization problems in image classification. In this section, we will first define the problem of domain generalization formally and then elaborate on the proposed CSVPT method.

3.1 Problem

This paper mainly studies the problem of domain generalization. In this subsection, we give the formal definition of domains and domain generalization in the following.

Definition 1 (Domain) [6]. To be simple, a domain is a set of data sampled from a distribution, denoted as $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n} \sim P_{XY}$, where $\mathbf{x}_i \in \mathcal{X}$ denotes an input sample in the input space, $y_i \in \mathcal{Y}$ denotes corresponding label in the output space, n denotes number of samples in domain $\mathcal{S}$, and P_{XY} denotes the

joint distribution of random variable X (which means the input sample) and random variable Y (which means the output label).

Definition 2 (Domain Generalization) [6]. In the setting of DG problem, we have M training domains $\mathcal{S}_{train} = \{\{(\mathbf{x}_i^{(d)}, y_i^{(d)}\}_{i=1}^{n_d}\}_{d=1}^{M}$ and one testing domain $\mathcal{S}_{test} = \{(\mathbf{x}_i^{(M+1)}, y_i^{(M+1)}\}_{i=1}^{n_{M+1}}$. The joint distributions between every two above domains are different: $P_{XY}^{(i)} \neq P_{XY}^{(j)}$, $1 \leq i \neq j \leq M+1$. DG aims to learn a mapping function $f : \mathcal{X} \to \mathcal{Y}$ (using $\mathcal{S}_{train}$ only) such that f minimizes the generalization error on domain $\mathcal{S}_{test}$:

$$\min_f \mathbb{E}_{(\mathbf{x},y)\in\mathcal{S}_{test}}[l(f(\mathbf{x}), y)]. \tag{1}$$

Note that the testing domain $\mathcal{S}_{test}$ is not available during training, which differs from the problem of domain adaptation [35].

3.2 Our CSVPT Algorithm

We propose a novel Common-Specific Visual Prompt Tuning (CSVPT) method to adapt large-scale vision Transformer models for domain generalization (Fig.1). It tunes the pre-trained vision Transformer by appending N trainable prompt tokens to the input of the Transformer layers. Different from existing prompt-based methods [25, 26] which only learn task-specific prompts, our proposed CSVPT jointly learns domain-common prompts and sample-specific prompts to modify the input to the pre-trained vision Transformer models. The domain-common prompts capture the task context and are fixed once learned, while the sample-specific prompts capture information about data distribution and are generated through a lightweight neural network (PGM). Combining the two types of prompts above, we can make more appropriate prompts for test data from unseen domains, which helps out-of-distribution generalization. Before elaborating on our CSVPT algorithm, let us review the vanilla vision Transformer (ViT) first.

For a vision Transformer [18], an input image $\mathbf{x}$ is divided into M patches first: $P = \{I_i | I_i \in \mathbb{R}^{3\times h\times w}, 1 \leq i \leq M\}$, where h and w denote the height and width of an image patch, respectively. Then, each image patch is mapped to a d-dimension vector, a.k.a, the context token, by a feature embedding module (usually a 2D convolution layer):

$$t_i^{(0)} = \text{Embed}(I_i), \tag{2}$$

where $I_i \in P$ and $t_i^{(0)} \in \mathbb{R}^d$. After that, the class token $c^{(0)} \in \mathbb{R}^d$ and all d-dimensional context tokens are concatenated as the input of the first Transformer layer:

$$(c^{(1)}, t_1^{(1)}, \ldots, t_M^{(1)}) = \text{L}_1(c^{(0)}, t_1^{(0)}, \ldots, t_M^{(0)}), \tag{3}$$

where the superscripts "(1)" and "(0)" denote the output and input of the first Transformer layer, respectively. For a vision Transformer with L layers, we have

$$(c^{(l)}, t_1^{(l)}, \ldots, t_M^{(l)}) = \text{L}_l(c^{(l-1)}, t_1^{(l-1)}, \ldots, t_M^{(l-1)}), \tag{4}$$

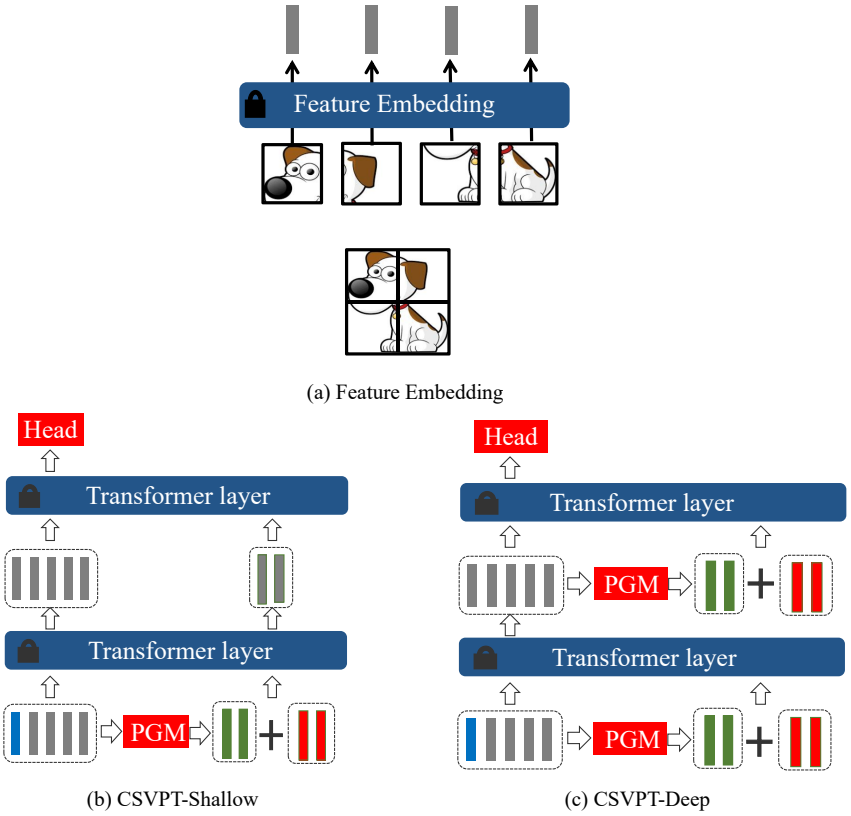

Fig. 1. Overview of the proposed Common-Specific Visual Prompt Tuning (CSVPT). (a) We first utilize a feature embedding module to embed input image patches into several context tokens (marked gray). (b) Mapping the class token (marked blue, pre-trained) and input context tokens to several prompt tokens (sample-specific prompts, marked green) through a prompt-generating module (PGM). Then, initialize domain-common prompts (marked red) to zeros and add domain-common prompts and sample-specific prompts together as total prompts. Finally, concatenate the class token, input context tokens and total prompt tokens as the input of the frozen Transformer Layer (CSVPT-Shallow). (c) Use PGM to generate sample-specific prompt tokens and initialize domain-common prompt tokens at each Transformer encoder layer (CSVPT-Deep). During training on downstream DG tasks, only the parameters (marked red) of domain-common prompts, PGMs and the linear head are updated, while keeping the whole Transformer encoder frozen.

where $l \in \{1, 2, \ldots, L\}$. Finally, $c^{(L)}$ is used for classification through the head module:

$$\hat{y} = \text{Head}(c^{(L)}). \tag{5}$$

As mentioned above, our proposed CSVPT algorithm (Fig.1) attempts to learn common-specific visual prompts to tune the pre-trained ViTs for better OOD generalization. Specifically, we first utilize a feature embedding module to embed input image patches into several context tokens (Fig. 1(a)) as in (2). Then, CSVPT utilizes a Prompt-generating Module (PGM) to generate sample-specific prompt tokens from the input context tokens to adapt to changes of data distributions:

$$(p_{s,1}^{(k)}, \ldots, p_{s,N}^{(k)}) = \text{PGM}(c^{(k)}, t_1^{(k)}, \ldots, t_M^{(k)}), \tag{6}$$

where k represents the input prompts of the $(k+1)$th layer, and the subscript "s" in $p_{s,i}^{(k)}$ means specific prompts. For the sake of simplicity, we apply a linear module as PGM:

$$[p_{s,1}^{(k)}, \ldots, p_{s,N}^{(k)}] = [c^{(k)}, t_1^{(k)}, \ldots, t_M^{(k)}]\mathbf{W}^{\mathbf{T}}, \tag{7}$$

where $\mathbf{W} \in \mathbb{R}^{N \times (M+1)}$ denotes the weights of the PGM. Next, we add specific and common prompt tokens (which are initialized to zeros and updated during training like other prompt tuning methods) together:

$$p_i^{(k)} = p_{s,i}^{(k)} + p_{c,i}^{(k)}, \tag{8}$$

where $i \in \{1, 2, \ldots, N\}$ and the subscript "c" in $p_{c,i}^{(k)}$ means common prompts. Similar to VPT, we propose two specific implementations: CSVPT-Shallow and CSVPT-Deep. In the setting of CSVPT-Shallow (Fig.1(b)), sample-specific prompt tokens generated by PGM and domain-common prompt tokens are used only in the input space of the first Transformer layer (i.e., $k \in \{0\}$), while at each Transformer layer (i.e., $k \in \{0, 1, \ldots, L-1\}$) in the setting of CSVPT-Deep (Fig.1(c). Finally, the total prompt tokens are used to modify the input to the pre-trained vision Transformer models. Specifically, for CSVPT-Shallow, we have

$$\begin{aligned} &(c^{(l)}, t_1^{(l)}, \ldots, t_M^{(l)}, p_1^{(l)}, \ldots, p_N^{(l)}) \\ =&\text{L}_l(c^{(l-1)}, t_1^{(l-1)}, \ldots, t_M^{(l-1)}, p_1^{(l-1)}, \ldots, p_N^{(l-1)}), \end{aligned} \tag{9}$$

where $\{p_i^{(0)}\}_{i=1}^N$ is obtained from (8). But for VPT-Deep, we have

$$\begin{aligned} &(c^{(l)}, t_1^{(l)}, \ldots, t_M^{(l)}, \widetilde{p}_1^{(l)}, \ldots, \widetilde{p}_N^{(l)}) \\ =&\text{L}_l(c^{(l-1)}, t_1^{(l-1)}, \ldots, t_M^{(l-1)}, p_1^{(l-1)}, \ldots, p_N^{(l-1)}), \end{aligned} \tag{10}$$

where $\{p_i^{(l-1)}\}_{i=1}^N$ is obtained from (8) and $\widetilde{p}_i^{(j)}$ is discarded in the next layer's input in order to prevent the number of tokens from gradually increasing with layers. During training on downstream DG tasks, only parameters of common prompt tokens, PGMs and the linear head are updated while the whole Transformer encoder is frozen, simply using the cross entropy loss.

4 Experiments

In this section, we conduct extensive experiments to compare the performance of the proposed CSVPT algorithm and other pre-trained model-based DG methods on different challenging and widely used datasets via DomainBed [15], a testbed for domain generalization.

4.1 Datasets and Implementation Details

Datesets. We evaluate our method on five challenging benchmark datasets, including PACS [27] (4 domains, 7 classes, and 9, 991 examples), VLCS [28] (4 domains, 5 classes, and 10, 729 examples), OfficeHome [29] (4 domains, 65 classes,and 15, 588 images), TerraIncognita [30] (4 domains, 10 classes, and 24, 788 examples), and DomainNet [31] (6 domains, 345 classes, and 586, 575 examples).
Implement details. Two backbones, ViT-B/16 [18] and ViT-L/14, are used on these five datasets to explore the influence of backbone scale on the performance of downstream DG tasks. Without further explanation, ViT-B/16 is used in the next experiments. In general, the number of training iterations is 5,000, and the batch size is 16 times the number of domains. But more training iterations are implemented for large datasets like DomainNet, and we reduce the batch size by half when using a larger backbone ViT-L/14. In our experiments, we uniformly use the Adam optimizer and determine the optimal learning rate through a model selection process. We follow the same training, validation and testing split scheme as in MIRO [21]. For each domain, twenty percent of the data is set aside as a validation set, that is, the holdout fraction is 0.2.

As Ishaan Gulrajani and David Lopez-Paz [15] put it, any DG algorithm without model selection is incomplete. We use two model selection criteria in our experiments:

- **Training-domain validation set**. We split every training domain into two subsets used for training and validation respectively. Then, we aggregate the validation subsets of all training domains to obtain the final validation set. Finally, the model maximizing the top-1 accuracy on the validation set is chosen as the best model.
- **Testing-domain validation set**. We split the testing domain into testing and validation subsets. The model maximizing the top-1 accuracy on this validation set is chosen as the best model.

Note that the leave-one-domain-out validation experiments are conducted just like previous work [27]. We select one domain for testing and the remaining domains for training every time. Finally, the average top-1 accuracy of classification is calculated as the evaluation metric for each dataset.

4.2 Main Results and Discussion

Comparison with other methods. In our experiments, we choose the vanilla VPT [25] method and popular DG methods (including ERM [15], DANN [8],

Table 1. Experimental Results of ViT-B/16.

Model selection method: training-domain validation set						
Algorithm	PACS	VLCS	OH	TI	DN	Avg.
ERM [15]	92.9	81.4	78.9	53.6	56.1	72.6
DANN [8]	92.2	80.1	78.0	47.9	57.5	71.1
CORAL [11]	92.6	79.6	78.5	51.7	56.4	71.8
MIRO [21]	95.2	81.1	82.5	52.9	56.6	73.7
VPT-Shallow [25]	**97.0**	81.7	83.6	53.1	58.7	74.8
CSVPT-Shallow (Ours)	96.6	**82.7**	**84.2**	**56.3**	**59.2**	**75.8**
VPT-Deep [25]	**96.6**	**82.9**	85.0	57.0	59.6	76.2
CSVPT-Deep (Ours)	**96.6**	82.7	**85.5**	**57.7**	**59.8**	**76.5**
Model selection method: testing-domain validation set						
ERM [15]	90.8	79.1	74.9	54.2	56.2	71.0
DANN [8]	90.6	81.4	76.2	50.7	57.7	71.3
CORAL [11]	90.6	80.2	76.8	52.0	56.0	71.1
MIRO [21]	95.8	83.6	82.3	58.8	57.2	75.5
VPT-Shallow [25]	**96.8**	83.2	83.6	56.7	59.0	75.9
CSVPT-Shallow (Ours)	96.6	**83.5**	**84.5**	**60.1**	**59.5**	**76.8**
VPT-Deep [25]	97.2	**84.9**	**85.2**	59.9	59.8	77.4
CSVPT-Deep (Ours)	**97.3**	**84.9**	85.0	**60.1**	**60.0**	**77.5**

CORAL [11], MIRO [21]) as our baselines. We provide extensive performance on five DG benchmarks in Table 1. For the sake of fairness, all the experiments in Table 1 are conducted with the same backbone, ViT-B/16 for CLIP [32], keeping other experimental conditions as consistent as possible. As said in the last subsection, we use two model selection criteria in our experiments: training-domain validation set and testing-domain validation set. We show the corresponding experimental results at the top and bottom of Table 1. Experimental results in Table 1 show that popular full fine-tuning DG methods (e.g., ERM [15], DANN [8], CORAL [11]) are prone to overfitting training data and often have inferior DG performances. MIRO that utilizes a mutual information regularization significantly improves performance on every dataset, resulting in +0.9pp average improvement (72.6% $\rightarrow$ 73.7%) with training domain validation set and +4.2pp average improvement (71.3% $\rightarrow$ 75.5%) with testing domain validation set, which achieved SOTA performance with the backbone of ViT-B/16 for CLIP before VPT. Compared to MIRO, the vanilla VPT algorithm further improves performance on almost every benchmark dataset. We obtain 74.8% average top-1 accuracy with VPT-Shallow and 76.2% with VPT-Deep using the first model selection criterion, while VPT-Shallow achieves 75.9% average top-1 accuracy and VPT-Deep 77.4% using the second model selection criterion. VPT-Deep slightly outperforms VPT-Shallow, because VPT-Shallow only tunes the input of the first Transformer layer, and VPT-Deep tunes that of each layer. Most importantly, our proposed CSVPT further improves the performance on the five benchmark datasets, especially in the setting of "Shallow": +1.0pp average

Table 2. Experimental Results of ViT-L/14.

Model selection method: training-domain validation set						
Algorithm	PACS	VLCS	OH	TI	DN	Avg.
VPT-Shallow [25]	98.5	81.6	88.5	60.0	63.6	78.4
CSVPT-Shallow (Ours)	**98.6**	**81.8**	**89.0**	**61.5**	**63.7**	**78.9**
VPT-Deep [25]	98.4	82.3	90.8	**65.4**	65.0	80.4
CSVPT-Deep (Ours)	**98.5**	**83.0**	**90.9**	65.3	**65.3**	**80.6**
Model selection method: testing-domain validation set						
VPT-Shallow [25]	**98.5**	81.7	88.3	62.9	**64.3**	79.1
CSVPT-Shallow (Ours)	**98.5**	**82.1**	**88.9**	**66.1**	64.1	**79.9**
VPT-Deep [25]	98.5	84.3	90.3	**67.3**	65.5	81.2
CSVPT-Deep (Ours)	**98.7**	**85.3**	**90.7**	67.1	**65.7**	**81.5**

improvement (74.8% $\rightarrow$ 75.8%) with the training domain validation set and +0.9pp average improvement (75.9% $\rightarrow$ 76.8%) with the testing domain validation set. However, in the setting of "Deep", the improvement of CSVPT is not so remarkable: +0.3pp average improvement (76.2% $\rightarrow$ 76.5%) with the training domain validation set and +0.1pp average improvement (77.4% $\rightarrow$ 77.5%) with the testing domain validation set. Anyway, the experimental results have shown the superior performance of CSVPT over popular DG methods and the vanilla VPT, which validates the effectiveness of the CSVPT algorithm.

CSVPT on various backbone scales. In this part, we explore how different backbone scales influence the performance of our proposed CSVPT. We choose two different backbones for comparison: ViT-B/16 and ViT-L/14 for CLIP. The experimental results are listed in Table 1 and Table 2, respectively. Similarly, in Table 2, CSVPT also achieves superior performances over VPT, especially in the setting of "Shallow": +0.5pp average improvement (78.4% $\rightarrow$ 78.9%) with the training domain validation set and +0.8pp average improvement (79.1% $\rightarrow$ 79.9%) with the testing domain validation set. Besides, we can find that there is a remarkable performance improvement on the five benchmarks when using larger Transformer backbones ViT-L/14: +4.0pp average improvement (77.5% $\rightarrow$ 81.5%), which achieves new SOTA performances.

CSVPT with different dataset scales. As we know, the VPT method [25] consistently outperforms other tuning methods across different training dataset scales. In order to explore whether our proposed CSVPT algorithm could handle domain generalization problems with different dataset scales, we gradually reduce the training data in the next experiments. Detailed experimental results are presented in Fig.2. From Fig.2, we can find that the average accuracy decreases gradually as the size of the training set decreases, but our proposed CSVPT always outperforms VPT and MIRO in this process. Surprisingly, CSVPT with less than 40% of training data (i.e., the holdout fraction is 0.6) could achieve the performance of VPT with 80% of training data (i.e., the holdout fraction is 0.2), while CSVPT with only 20% of training data achieves the performance of MIRO

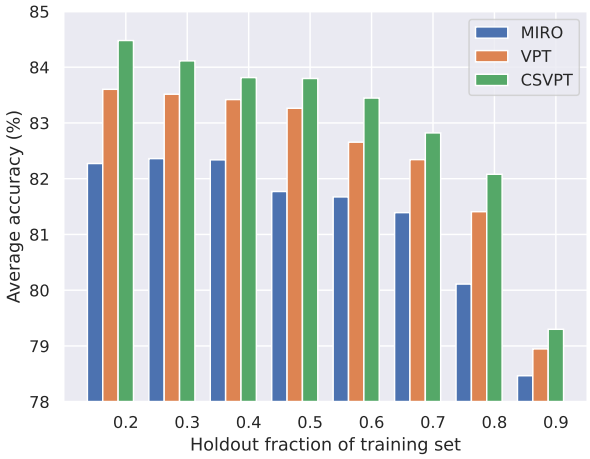

Fig. 2. DG performances with different dataset scales on the OfficeHome benchmark. We change the holdout fraction of the training set from 0.2 to 0.9, that is, we gradually reduce the training data from 80% to 10% of all training domain datasets. The variation of the average accuracy on the four domains that are alternately used as the testing domain with the size of the training set is shown in the histogram.

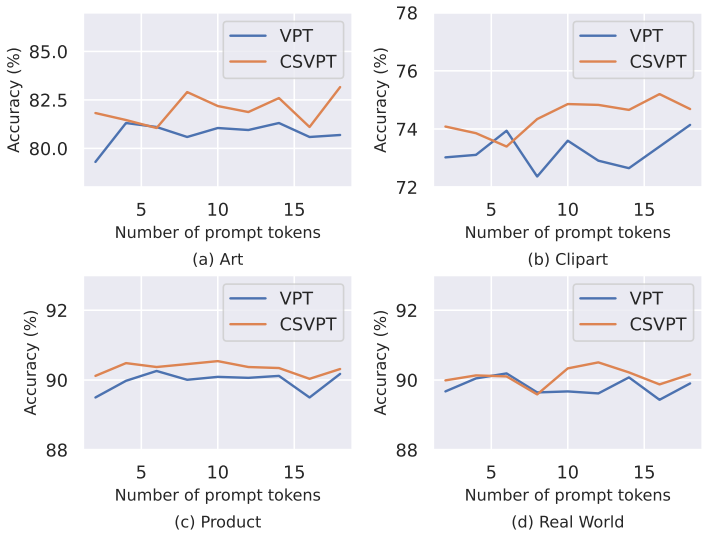

Fig. 3. Performaces of VPT (without PGMs) and VPT (with PGMs) with different numbers of prompt tokens N. In this experiment, we also adopt the leave-one-domain-out scheme, which chooses one domain as the testing domain and aggregates the remaining domains as training domains. We show experimental results on the four domains of OfficeHome benchmark in this figure: (a) Art; (b) Clipart; (c) Product; (d) Real World. The caption represents the testing domain at each time.

with 80% of training data. Besides, when training data is extremely scarce, the performance of CSVPT deteriorates rapidly. We believe that the reason for this phenomenon is that training data is too scarce to train PGMs well enough to generate meaningful specific prompts.

CSVPT with different numbers of prompt tokens N. The difference between the vanilla VPT and our proposed CSVPT lies in whether PGMs are used to generate specific prompts. We conduct more experiments to illustrate the difference with and without the PGMs. In this part, different numbers of prompt tokens N are used in our experiments to show the consistency of CSVPT's superiority over VPT methods. Specially, we only change the number of prompt tokens N from 2 to 18, keeping other experimental conditions the same. We show detailed experimental results on the OfficeHome benchmark in Fig.3. Observing the curves in Fig.3, we can find that the CSVPT method achieves the highest performance when using different numbers of prompt tokens for different datasets. For Art, CSVPT achieved the highest performance of 83.2% when N equals 18. But for Clipart, Product and Real World, N equals 18, 10 and 12, respectively. Yet despite all that, the orange curve almost always lies on top of the blue curve in each subfigure, which means CSVPT (with PGMs) almost always outperforms VPT (without PGMs) on each testing domain of OfficeHome benchmark. This observation is violated only at some special points, e.g., when N equals 6 or 8. However, CSVPT always achieves higher optimal performance than VPT.

More Ablations on common and specific prompts. The above experiments mainly focus on the comparison between VPT (which only uses common prompts) and CSVPT (which uses both common prompts and specific prompts). To further show the necessity of both kinds of prompts, we conduct an ablation study to compare results with common prompts only (VPT), specific prompts only (SVPT), and a combination of both (CSVPT). Experimental results in Table 3 show that CSVPT achieves the best performance, which also justifies both the necessity of common prompts and that of specific prompts in CSVPT.

Table 3. Ablation Experiments on Common and Specific Prompts.

Algorithm	PACS	VLCS	OH	TI	DN	Avg.
VPT	97.0	81.7	83.6	53.1	58.7	74.8
SVPT	95.8	81.5	82.8	50.7	58.0	73.8
CSVPT	**96.6**	**82.7**	**84.2**	**56.3**	**59.2**	**75.8**

4.3 Case Study

In this part, we study three cases in detail to further illustrate how the PGMs influence the output of Transformers. Similar to other tuning methods [36], VPT-

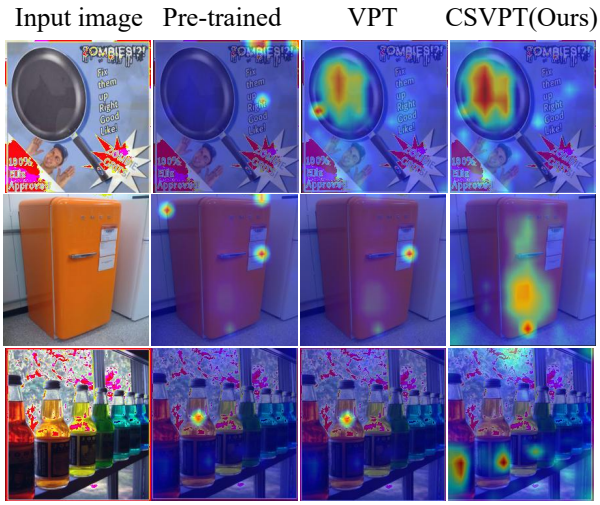

Fig. 4. We show how the attention flows from the start to the end throughout the Transformer when using the pre-trained backbone (without fine-tuning), VPT methods and our proposed CSVPT methods.

based methods mainly influence the process of attention in this way:

$$\begin{aligned}&\text{Attn}(\mathbf{q^T W_q}, \text{concat}(\mathbf{C W_k}, \mathbf{P W_k}), \text{concat}(\mathbf{C W_v}, \mathbf{P W_v}))\\ =&\lambda \text{Attn}(\mathbf{q^T W_q}, \mathbf{C W_k}, \mathbf{C W_v}) + (1-\lambda)\text{Attn}(\mathbf{q^T W_q}, \mathbf{P W_k}, \mathbf{P W_v}),\end{aligned} \tag{11}$$

where $\mathbf{q} \in \mathbb{R}^d$ denotes a query token, $\mathbf{C} = [t_1, \ldots, t_M]^T \in \mathbb{R}^{M \times d}$ denotes context tokens, $\mathbf{P} = [p_1, \ldots, p_N]^T \in \mathbb{R}^{N \times d}$ denotes prompt tokens, $\mathbf{W_q}, \mathbf{W_k}, \mathbf{W_v} \in \mathbb{R}^{d \times d'}$ denote the weights of the linear layers before attention, and

$$\lambda = \frac{\sum_i \exp(\mathbf{q^T W_q W_k^T C^T})_i}{\sum_i \exp(\mathbf{q^T W_q W_k^T C^T})_i + \sum_i \exp(\mathbf{q^T W_q W_k^T P^T})_i}. \tag{12}$$

In (11), the second term shows how prompt tokens influence the previous attention. In order to intuitively demonstrate the effect of (sample-specific) prompt tokens on the attention, we show how the attention flows from the start to the end in Fig.4 using a technique called "Attention Rollout" [37]. In the first case of Fig.4, the pre-trained model without fine-tuning focuses on pixels not on the "pan", while the models with VPT-based method focus on the right pixels. However, compared to the VPT method (without specific prompt tokens), the proposed CSVPT method (with specific prompt tokens generated by PGMs) pays more attention to the object in the image. In the second case, pre-trained models without fine-tuning and with VPT method only focus on local point-like regions of the object in the image, but our CSVPT method focuses on bigger regions of the object. Surprisingly, in the third case, the CSVPT method captures three objects in the image, while other methods only focus on the bottleneck

of one bottle in that image. In a word, the success of the three cases shown in Fig.4 further illustrates the benefit of PGMs and sample-specific prompt tokens on the DG classification tasks.

5 Conclusion

In this paper, we propose a novel Common-Specific Visual Prompt Tuning (CSVPT) method to adapt pre-trained vision Transformers for domain generalization problems in image classification. It tunes the pre-trained vision Transformer by appending several trainable prompt tokens to the input of the Transformer layers. Different from existing methods, CSVPT jointly learns domain-common prompts to capture the task context and sample-specific prompts to capture information about data distribution, which are generated for each sample through a trainable prompt-generating module (PGM). Combining the domain-common prompts and the sample-specific prompts, CSVPT makes dynamic visual prompts changing with data distributions adaptively, which helps OOD generalization. Our experimental results demonstrate the effectiveness of the proposed method. CSVPT achieves consistent superior performance over other related DG methods under different experimental settings. Furthermore, CSVPT with ViT-L/14 achieves new state-of-the-art performances on the five widely used datasets of DomainBed benchmarks. Our experimental results also illustrate that the success of CSVPT probably results from its positive influence on the attention flows in the vision Transformer models. We hope that this study will encourage more research on advanced fine-tuning approaches for domain generalization, and more theoretical analysis and technical improvement will be our future work.

Acknowledgements. This work was supported in part to Dr. Liansheng Zhuang by NSFC under contract No.U20B2070 and No.61976199.

References

1. A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.
2. M. M. Lopez and J. Kalita, "Deep learning applied to nlp," *arXiv preprint arXiv:1703.03091*, 2017.
3. Z. Zhang, J. Geiger, J. Pohjalainen, A. E.-D. Mousa, W. Jin, and B. Schuller, "Deep learning for environmentally robust speech recognition: An overview of recent developments," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 9, no. 5, pp. 1–28, 2018.
4. U. Kamath, J. Liu, and J. Whitaker, *Deep learning for NLP and speech recognition*. Springer, 2019, vol. 84.
5. Z. Shen, J. Liu, Y. He, X. Zhang, R. Xu, H. Yu, and P. Cui, "Towards out-of-distribution generalization: A survey," *arXiv preprint arXiv:2108.13624*, 2021.

6. J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Transactions on Knowledge and Data Engineering*, 2022.
7. K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization in vision: A survey," *arXiv preprint arXiv:2103.02503*, 2021.
8. Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.
9. H. Li, S. J. Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5400–5409.
10. Y. Li, X. Tian, M. Gong, Y. Liu, T. Liu, K. Zhang, and D. Tao, "Deep domain generalization via conditional invariant adversarial networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 624–639.
11. B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *European conference on computer vision*. Springer, 2016, pp. 443–450.
12. M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, "Invariant risk minimization," *arXiv preprint arXiv:1907.02893*, 2019.
13. A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba, "Undoing the damage of dataset bias," in *European Conference on Computer Vision*. Springer, 2012, pp. 158–171.
14. V. Piratla, P. Netrapalli, and S. Sarawagi, "Efficient domain generalization via common-specific low-rank decomposition," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7728–7738.
15. I. Gulrajani and D. Lopez-Paz, "In search of lost domain generalization," *arXiv preprint arXiv:2007.01434*, 2020.
16. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
17. L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, no. 4, pp. 681–694, 2020.
18. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.
19. C. Zhang, M. Zhang, S. Zhang, D. Jin, Q. Zhou, Z. Cai, H. Zhao, X. Liu, and Z. Liu, "Delving deep into the generalization of vision transformers under distribution shifts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7277–7286.
20. Z. Li, K. Ren, X. Jiang, B. Li, H. Zhang, and D. Li, "Domain generalization using pretrained models without fine-tuning," *arXiv preprint arXiv:2203.04600*, 2022.
21. J. Cha, K. Lee, S. Park, and S. Chun, "Domain generalization by mutual-information regularization with pre-trained models," *arXiv preprint arXiv:2203.10789*, 2022.
22. X. Zhang, Y. Iwasawa, Y. Matsuo, and S. S. Gu, "Amortized prompt: Guide clip to domain transfer learning," *arXiv preprint arXiv:2111.12853*, 2021.
23. D. Hendrycks, X. Liu, E. Wallace, A. Dziedzic, R. Krishnan, and D. Song, "Pretrained transformers improve out-of-distribution robustness," *arXiv preprint arXiv:2004.06100*, 2020.

24. A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, "Fine-tuning can distort pretrained features and underperform out-of-distribution," *arXiv preprint arXiv:2202.10054*, 2022.
25. M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.-N. Lim, "Visual prompt tuning," *arXiv preprint arXiv:2203.12119*, 2022.
26. K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *arXiv preprint arXiv:2109.01134*, 2021.
27. D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5542–5550.
28. C. Fang, Y. Xu, and D. 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*, 2013, pp. 1657–1664.
29. H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5018–5027.
30. S. Beery, G. Van Horn, and P. Perona, "Recognition in terra incognita," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 456–473.
31. X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1406–1415.
32. A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763.
33. H. Bai, R. Sun, L. Hong, F. Zhou, N. Ye, H.-J. Ye, S.-H. G. Chan, and Z. Li, "Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 6705–6713.
34. G. Parascandolo, A. Neitz, A. Orvieto, L. Gresele, and B. Schölkopf, "Learning explanations that are hard to vary," *arXiv preprint arXiv:2009.00329*, 2020.
35. M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018.
36. J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, "Towards a unified view of parameter-efficient transfer learning," *arXiv preprint arXiv:2110.04366*, 2021.
37. S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," *arXiv preprint arXiv:2005.00928*, 2020.