Improving Visual Prompt Tuning for Self-supervised Vision Transformers

Seungryong Yoo [1] **Eunji Kim** [1] **Dahuin Jung** [1] **Jungbeom Lee** [1] **Sungroh Yoon** [1,2]

Abstract

Visual Prompt Tuning (VPT) is an effective tuning method for adapting pretrained Vision Transformers (ViTs) to downstream tasks. It leverages extra learnable tokens, known as prompts, which steer the frozen pretrained ViTs. Although VPT has demonstrated its applicability with supervised vision transformers, it often underperforms with self-supervised ones. Through empirical observations, we deduce that the effectiveness of VPT hinges largely on the ViT blocks with which the prompt tokens interact. Specifically, VPT shows improved performance on image classification tasks for MAE and MoCo v3 when the prompt tokens are inserted into later blocks rather than the first block. These observations suggest that there exists an optimal location of blocks for the insertion of prompt tokens. Unfortunately, identifying the optimal blocks for prompts within each self-supervised ViT for diverse future scenarios is a costly process. To mitigate this problem, we propose a simple yet effective method that learns a gate for each ViT block to adjust its intervention into the prompt tokens. With our method, prompt tokens are selectively influenced by blocks that require steering for task adaptation. Our method outperforms VPT variants in FGVC and VTAB image classification and ADE20K semantic segmentation. The code is available at https://github.com/ryongithub/GatedPromptTuning.

1. Introduction

Currently, self-supervised learning (SSL) with Vision Transformers (ViTs) (Bao et al., 2021; He et al., 2022; Chen et al., 2021; Caron et al., 2021) have exhibited remarkable results across diverse visual recognition tasks such as classification and semantic segmentation. SSL approaches strive to
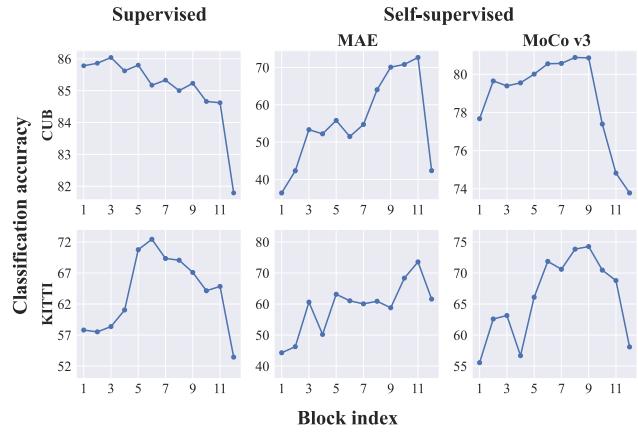

Figure 1. Classification accuracy on the CUB and KITTI datasets with a varying location where prompt tokens are inserted in the pretrained ViT-B/16. MAE and MoCo v3 significantly improve their performances when prompt tokens are affected by the blocks after the 11th and 8th blocks, respectively. The block index denotes the initial insertion point of the prompt tokens.

train the neural networks without biasing them toward labels containing very specific information about the corresponding tasks (Zhao et al., 2020). As a result, self-supervised models demonstrate superior scalability across various vision tasks compared to supervised ones (Chen et al., 2020; He et al., 2020; Grill et al., 2020; He et al., 2022; Caron et al., 2021; Bao et al., 2021). However, the efficacy of self-supervised models hinges on the chosen transfer learning strategy. For instance, a large performance gap exists between full fine-tuning and linear probing when employed as a transfer method for the Masked Autoencoder (MAE) (He et al., 2022). The performance of SSL ViTs can vary significantly based on the transfer method, underscoring the importance of research into transfer strategies for SSL ViTs.

Visual Prompt Tuning (VPT) (Jia et al., 2022) is an effective prompt-based transfer learning technique, drawing its concept from previous work in the field of natural language processing (NLP) (Li & Liang, 2021; Lester et al., 2021). In detail, VPT prepends learnable prompt tokens to the input sequences, which then act as task-specific instructions by steering the information from the fixed pretrained encoder. VPT, when used with supervised ViT backbones, has shown outstanding performance on numerous downstream tasks.

---

[1]Electrical and Computer Engineering, [2]Interdisciplinary Program in Artificial Intelligence, Seoul National University, Seoul, Korea. Correspondence to: Sungroh Yoon <sryoon@snu.ac.kr>.

However, despite the successes of prompt tuning (Lester et al., 2021; Hambardzumyan et al., 2021; Qin & Eisner, 2021; Huang et al., 2023) with SSL models in NLP (Devlin et al., 2018; Liu et al., 2019), VPT has demonstrated relatively poor performances with SSL ViTs (Jia et al., 2022). To address this, we propose a simple yet effective transfer method based on prompt tuning that enhances the performance of SSL ViTs.

It is well-known that Deep Neural Networks (DNNs) progressively abstract the information contained in data samples across their layers (LeCun et al., 2015; Schmidhuber, 2015; Kriegeskorte, 2015). Considering this in the context of VPT, information from all levels of abstraction influences the prompt tokens. However, these prompt tokens should be able to encode instructions by focusing solely on task-relevant information for target task adaptation. This can pose a challenge for prompt tokens from two perspectives. Firstly, the encoder possesses different information hierarchies across the blocks depending on how it was pretrained. Secondly, the task-relevant information required can vary depending on the downstream task.

We conjecture that what the prompt tokens learn heavily relies on which blocks influence them during training. To justify our conjecture, we conducted an experiment to observe how performance varies depending on which blocks intervene with the prompt tokens. Interestingly, in Figure 1, on the CUB (Wah et al., 2011) classification benchmark, MAE (He et al., 2022) and MoCo v3 (Chen et al., 2021) boost their accuracy when the prompt tokens begin interacting with the 11th and 8th blocks in ViT-B, respectively. Particularly for MAE, the performance gap is as large as 36.4%. Based on these findings, the key intuition of this study is that there exist desirable blocks that the prompt tokens should focus on to steer.

Depending on the pretraining strategy, the pretrained neural networks encode information differently in terms of the amount and content (Zhao et al., 2020; Bordes et al., 2021). Additionally, the relevant information required to solve the downstream task varies according to the task at hand. For these reasons, the task-relevant blocks will vary depending on their use in a downstream task and the pretraining method. Unfortunately, investigating all the possible cases to find the desirable sets of ViT blocks incurs substantial costs. To address this, we propose a simple yet effective method for learning to guide the prompt to selectively interact with desirable blocks that encode task-relevant information. We achieve this by introducing learnable gates for ViT blocks that adjust the intervention to the prompt tokens from ViT blocks. With our proposed method, the prompt tokens focus on the blocks that need to be steered for the target task adaptation.

Experimental results validate that our proposed method un-

locks the potential of prompt tuning as a universal tuning strategy for self-supervised ViTs. On the FGVC benchmark, which encompasses five fine-grained classification tasks (Wah et al., 2011; Van Horn et al., 2015; Khosla et al., 2011; Gebru et al., 2017; Nilsback & Zisserman, 2008), we achieve an average accuracy of 73.4% for MAE and 83.0% for MoCo v3. These results substantially outperform VPT-shallow and are either outperforming or comparable to VPT-deep. Moreover, on the VTAB-1K benchmark (Zhai et al., 2019), which consists of 19 diverse visual classification tasks, our proposed method attains an average accuracy of 49.2% and 65.8% for MAE and MoCo v3, respectively, significantly outperforming both VPT-deep and VPT-shallow. Our method demonstrates its strength not only in classification tasks but also in dense prediction tasks such as semantic segmentation. It exhibits superior performance compared to VPT counterparts on the ADE 20K semantic segmentation benchmark (Zhou et al., 2017), achieving 38.4 mIoU for MAE and 36.8 mIoU for MoCo v3.

2. Preliminaries

Vision Transformer. Typically, ViT consists of a patch embedding layer, a stack of L transformer blocks, and a classification head. To process an image of a height H, a width W, and a channel C into a ViT, we divide the image as a grid of patches $x_i \in \mathbb{R}^{P \times P \times C}$, where P is a patch size and $i = 1, \ldots, \frac{HW}{P^2}$. Each x_i is embedded as a D-dimensional feature and D-dimensional positional embedding is added to each patch token to provide position information to ViTs as follows:

$$z_i^0 = \text{PatchEmbed}(x_i) + e_i, \quad (1)$$

where z_i^0 denotes the embedded input token for the first ViT block and e_i denotes the positional embedding. Let the input patch tokens for l^{th} block as $\mathbf{Z}^{l-1} = \left[z_1^{l-1}, \ldots, z_N^{l-1} \right]$, where $N = \frac{HW}{P^2}$, $l = 1, \ldots, L$, and L is the number of blocks. Patch tokens and an additional learnable token for classification, z_{CLS}^l, is fed to the blocks as follows:

$$\left[z_{\text{CLS}}^l, \mathbf{Z}^l \right] = \text{Block}^l \left(\left[z_{\text{CLS}}^{l-1}, \mathbf{Z}^{l-1} \right] \right) \in \mathbb{R}^{(N+1) \times D}, \quad (2)$$

where each block consists of a multi-head self-attention followed by a feed-forward layer with layer normalization (Ba et al., 2016) and residual connection (He et al., 2016). Among multiple self-attention heads in l^{th} block, a single self-attention head (Vaswani et al., 2017) is formulated as follows:

$$\text{Attention}(Q^l, K^l, V^l) = a^l V^l$$
$$\text{s.t. } a^l = \text{Softmax}\left(Q^l {K^l}^T \right), \quad (3)$$

where Q^l, K^l, V^l present the input query, key, and value tokens built by the linear projection of $\left[z_{\text{CLS}}^{l-1}, \mathbf{Z}^{l-1} \right]$, respectively, and a^l presents the self-attention score calculated

in l^{th} block. Lastly, the classification head has a single feed-forward layer for class prediction.

Visual Prompt Tuning. VPT (Jia et al., 2022) trains continuous prompts in the embedded space. Learnable prompt tokens $\mathbf{P} = [p_1, \ldots, p_{N_p}] \in \mathbb{R}^{N_P \times D}$ are prepended to the input sequence, where N_p is the number of prompt tokens and D is the dimension of the prompt token. During transfer learning, the prompt tokens and a classification head are only trainable and the pretrained ViT encoder is fixed. The prompt tokens learn to encode task-specific instructions by interacting with patch representations across all ViT blocks. VPT introduced two variants: VPT-shallow and VPT-deep. VPT-shallow inserts learnable prompt tokens as input in the first block. The following equations formulate the VPT-shallow:

$$[z_{\text{CLS}}^1, \mathbf{Z}_P^1, \mathbf{Z}^1] = \text{Block}^1\Big([z_{\text{CLS}}^0, \mathbf{P}, \mathbf{Z}^0]\Big)$$
$$[z_{\text{CLS}}^l, \mathbf{Z}_P^l, \mathbf{Z}^l] = \text{Block}^l\Big([z_{\text{CLS}}^{l-1}, \mathbf{Z}_P^{l-1}, \mathbf{Z}^{l-1}]\Big), \quad (4)$$

where $\mathbf{Z}_P^l$ denotes the output prompt representation of lth block. Differently, VPT-deep injects block-wise learnable prompt tokens $\mathbf{P}^{l-1}$ to each block, not the previous block's output $\mathbf{Z}_P^{l-1}$:

$$[z_{\text{CLS}}^l, \_, \mathbf{Z}^l] = \text{Block}^l\Big([z_{\text{CLS}}^{l-1}, \mathbf{P}^{l-1}, \mathbf{Z}^{l-1}]\Big). \quad (5)$$

3. Motivation

The pretrained knowledge and the downstream task are two important factors in transfer learning scenarios. In this section, we discuss the motivational background of our research from these two perspectives.

Pretrained knowledge. We find that the block where the prompt tokens are inserted at first leads to varying performances in Figure 1. We verify these findings from the perspective of information contained in each block of ViTs. We utilize Deep Image Prior (DIP) (Ulyanov et al., 2018) to investigate the information change across the blocks of the pretrained ViTs. DIP reconstructs an image by updating the random noise so that the original image's representation and the updated noise's representation are close in the representation space. Using the reconstructed image, we can infer the information that the representation space of each neural network unit encodes. As shown in Figure 2, in the later blocks, supervised ViT tends to discard more information than self-supervised ViT. In contrast to supervised one, self-supervised ViTs, MoCo v3 (Chen et al., 2021) and MAE (He et al., 2022) retain rich information across the blocks. This tendency is also evident in the reconstructed image quality scores, such as PSNR and SSIM, as indicated in Figure 9 in Appendix C.1. Furthermore, it can be observed that even self-supervised Vision Transformers exhibit differences in
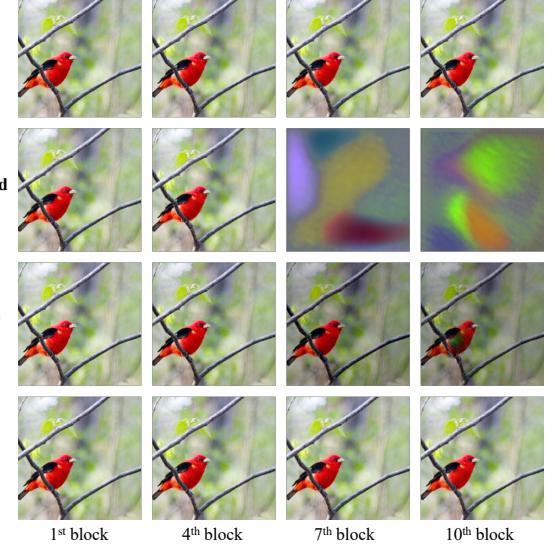

Figure 2. Reconstructed images using Deep Image Prior (DIP) with pretrained ViT block's representation as a training target. The reconstructed image maintains its similarity to the original image as the block preserves information till the last block. **Row 1**: original image. **Rows 2-4**: reconstruction results for each pretrained ViTs. Poor results in late blocks (7$^{\text{th}}$ and 10$^{\text{th}}$) of the supervised model indicate that it discards more information across blocks than the self-supervised ViTs.

the information content encoded by each block. When comparing MoCo v3 to MAE, it is noticeable that MoCo v3 exhibits a decrease in color information after the middle block. More DIP results are available in Appendix C.2.

This example shows that different pretraining methods lead ViT blocks to differ in terms of the amount and content of information they encode. It follows that the locations of blocks containing task-relevant information vary depending on the pretrained ViTs. If the prompt is inserted into the first block without considering these differences, the accumulated intervention of the task-irrelevant blocks could disturb the prompt tokens to focus on the task-relevant blocks.

Task diversity. Task-relevant information varies depending on the downstream task. For example, in the case of classification on the CUB dataset, discriminating fine color and shape information is crucial to classify diverse bird species. Unlike the bird classification task with CUB, the KITTI (Geiger et al., 2013) distance task requires capturing position and scale information to accurately estimate the distance to the objects in the scene. Thus, even with the same pretrained Vision Transformer, the locations of blocks encoding task-relevant information can vary depending on the task. Figure 1 illustrates that the performance change according to the location of the prompt insertion is different between CUB and KITTI. This indicates that the blocks that are desirable for the prompt insertion to perform task-adaptation are task-dependent.
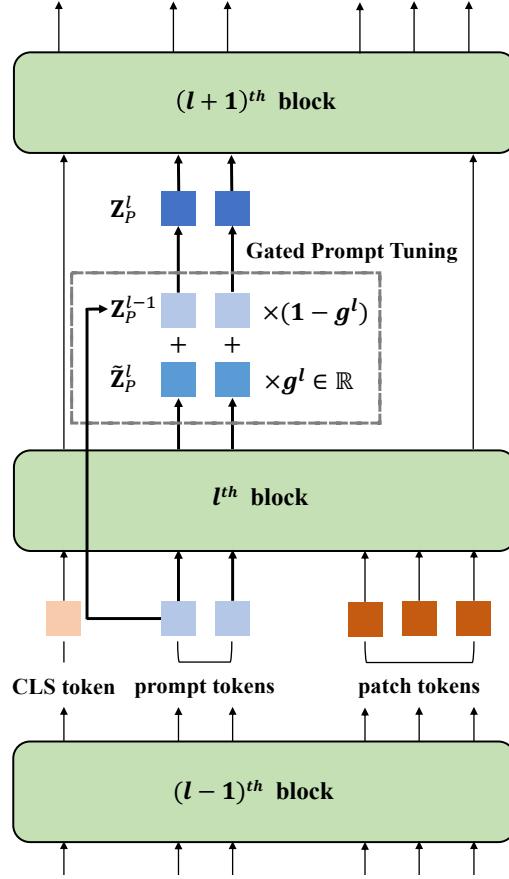
Figure 3. An illustration of our proposed method, **Gated Prompt Tuning**. $\mathbf{Z}_P^{l-1}$ and $\tilde{\mathbf{Z}}_P^l$ are input and output prompt representations of lth block. The learnable gate g^l convexly combinates $\mathbf{Z}_P^{l-1}$ and $\tilde{\mathbf{Z}}_P^l$ so that the $(l+1)^{\text{th}}$ block receives the prompt representation $\mathbf{Z}_P^l$ in which the intervention of lth block into the prompt representation is adjusted by g^l.

4. Proposed Method

In the previous section, we discuss that the locations of the blocks where the prompt can derive improved performances depend on the SSL method. Further, since the required task-specific instruction may vary depending on the task, the desirable blocks for the prompt could be different in the same pretrained model. Sufficient task performance may not be secured when prompts are inserted from the first block without careful consideration of this difference. In this respect, we consider a universal prompt tuning method that can learn to select the blocks where steering for task adaptation is strongly required. To achieve this goal, we introduce Gated Prompt Tuning, which leverages learnable gates to adjust each block's intervention into the prompt tokens. The learnable gates enable prompts to readily focus on the blocks that require steering for task adaptation. Similar to VPT-shallow, our method prepends the prompt tokens only once to the input patch sequence. Our framework is illustrated in Figure 3.

4.1. Gated Prompt Tuning

First, we define gate priors $\Gamma = [\gamma^1, \ldots, \gamma^{L-1}]$, a set of scalar values for all blocks except for the last block. After being scaled with a sigmoid function, the gate prior is utilized as a gate value of lth block:

$$g^l = \frac{1}{1 + e^{-\gamma^l}} \in \mathbb{R}. \tag{6}$$

The gating operation for the input prompt representations at $(l+1)^{\text{th}}$ block (*i.e.,* $\mathbf{Z}_P^l$) is formulated as follows:

$$
\begin{aligned}
\left[z_{CLS}^l, \tilde{\mathbf{Z}}_P^l, \mathbf{Z}^l\right] &= \text{Block}^l\left(\left[z_{CLS}^{l-1}, \mathbf{Z}_P^{l-1}, \mathbf{Z}^{l-1}\right]\right), \\
\mathbf{Z}_P^l &= g^l \cdot \tilde{\mathbf{Z}}_P^l + (1 - g^l) \cdot \mathbf{Z}_P^{l-1},
\end{aligned}
\tag{7}
$$

where $\tilde{\mathbf{Z}}_P^l$ denotes the output prompt representation of lth block. In Eq. 7, $(l+1)^{\text{th}}$ block receives the weighted sum of $\mathbf{Z}_P^{l-1}$ and $\tilde{\mathbf{Z}}_P^l$, which are the input and output prompt representation of lth block, respectively. Here, the gate value g^l controls the contribution on composing the input prompt representation of $(l+1)^{\text{th}}$ block between $\mathbf{Z}_P^{l-1}$ and $\tilde{\mathbf{Z}}_P^l$. Therefore, gate g^l determines how much of the previous block's influence on the prompt is carried over to the next block. During training, the gates learn to adjust the intervention of each block in the prompt tokens, which enables the prompts to focus on desirable blocks that require task-specific steering.

Right before the task head, the last block refines the representations to be discriminative for the task. The input prompt representations (*i.e.* $\mathbf{Z}_P^{L-1}$) for the last block have a significant role in this regard. Using Eq. 7, we can express the input prompt representations of the last block as follows:

$$
\begin{aligned}
\mathbf{Z}_P^{L-1} &= \left(\prod_{l=1}^{L-1}(1 - g^l)\right)\mathbf{P} \\
&+ \sum_{l=1}^{L-2}\left(\prod_{m=l+1}^{L-1}(1 - g^m)\right)g^l\tilde{\mathbf{Z}}_P^l + g^{L-1}\tilde{\mathbf{Z}}_P^{L-1}.
\end{aligned}
\tag{8}
$$

Since $\tilde{\mathbf{Z}}_P^l$ is the output prompt representations of lth block, Eq. 8 can be interpreted as a selective aggregation of all output prompt representations from the ViT blocks by the learned gates. The selective aggregation results in adaptive instructions for target tasks.

4.2. Adaptive Attention Shaping

Prompt tuning can be understood as learning to steer the behavior of ViTs by manipulating the pretrained attention score of the patch tokens by extending input sequences with extra prompt tokens. Based on this intuition, as an additional technique to boost the task-adaptability of prompt

tuning, we introduce Adaptive Attention Shaping. We define learnable temperature $\mathcal{T} = [\tau^1, \ldots, \tau^L]$, a set of scalar values that adjust the attention value in the self-attention operation of the corresponding block. With the learnable temperature, we rewrite the self-attention score in Eq. 3 as follows:

$$a^l = \text{Softmax}\Big(\frac{Q^l K^{l^T}}{\tau^l}\Big). \tag{9}$$

$\mathcal{T}$ directly reshape the blocks' self-attentions by making them sharper or smoother so that $\mathcal{T}$ aids prompts to encode beneficial instruction to solve the current task.

4.3. Comparison with VPT

Our method incorporates learnable gates to regularize prompt tokens to interact with task-relevant blocks and enables effective prompt learning by utilizing the additional capacity obtained through learnable temperatures. However, in VPT, there is no such consideration for effective learning of the prompt tokens.

Another difference between our method and VPT lies in whether the prompt can provide sample-specific but task-relevant instruction. In VPT-shallow, the prompt representation passed to each block is conditioned on the patch representation from the previous block, allowing for sample-specific instructions at each block. VPT-shallow can be considered as a special case of our method, as all learned gates are set to 1, and it lacks a learnable temperature. However, when applied to self-supervised ViTs, the prompt tokens interact with all blocks in VPT-shallow, and thus, VPT-shallow has a limitation in effectively targeting task-relevant information.

VPT-deep addresses the differences in task relevancy across blocks by providing independent learnable prompt token sets for each block. The prompts in each block of VPT-deep are trained to provide task-relevant instructions on average over the entire training data. However, since all samples receive the same instruction from shared prompt tokens at each block, VPT-deep could not provide sample-specific instruction.

In our method, the gating operation allows the prompt token to selectively interact with the blocks, and the input prompt representation for each block is dependent on the patch representation from the preceding block. Therefore, our method enables providing sample-specific but task-relevant instructions at each block.

5. Experiments

5.1. Experimental Setup

Downstream tasks and datasets. We evaluate our method with two types of downstream tasks: image classification

and semantic segmentation. For image classification, we conduct experiments on FGVC and VTAB-1K (Zhai et al., 2019) benchmark. FGVC includes five fine-grained classification tasks: CUB (Wah et al., 2011), Oxford Flowers (Nilsback & Zisserman, 2008), Stanford Cars (Gebru et al., 2017), Stanford Dogs (Khosla et al., 2011) and NABirds (Van Horn et al., 2015). VTAB-1K is divided into three subgroups: *Natural* with natural images, *Specialized* with images obtained from specialized equipment, and *Structured* which requires structural understanding such as 3D depth prediction.

For semantic segmentation, we evaluate the performances on ADE20K (Zhou et al., 2017) benchmark which contains 20K images with 150 object categories. For the segmentation model, we use SETR-PUP (Zheng et al., 2021) which utilizes ViT (Dosovitskiy et al., 2020) as a backbone encoder. Note that the original implementation of SETR-PUP adopts four auxiliary heads at the 10th, 15th, 20th, and 24th blocks of ViT-L. Since we use ViT-B/16 in all experiments, two auxiliary heads are used at the 5th and 9th blocks. Further experimental details are described in Sec. A.2 in Appendix.

Self-supervised Vision Transformers. Our study utilizes two well-performing self-supervised Vision Transformers, MAE (He et al., 2022) and MoCo v3 (Chen et al., 2021), which are pretrained on ImageNet-1K (Deng et al., 2009). Pretrained model parameters are taken from the official repository of Visual Prompt Tuning (Jia et al., 2022). We use ViT-B/16 as the backbone architecture in all experiments of this study.

5.2. Main Results

5.2.1. CLASSIFICATION ON FGVC

We evaluate the fine-grained classification performance in the FGVC benchmark. For VPT-shallow and our method, we set 100 tokens, and for VPT-deep, we set 10 tokens for each block, which means that 120 tokens are totally used for VPT-deep. Table 1 shows that our method consistently outperforms the VPT-shallow counterpart by a large margin both for MAE and MoCo v3 in all datasets. This shows that our approach leads the prompt to encode enhanced instruction in SSL ViTs for task adaptation. Note that compared to VPT-deep, MAE with our method has higher average accuracy while surpassing in almost all datasets. When applied to MoCo v3, even though VPT-deep uses 20% more extra parameters, our method shows comparable performance to VPT-deep on average. According to this, when the number of added prompt tokens is similar, our method is more effective at utilizing prompt tokens for fine-grained classification than both of these two variants of VPT. Moreover, our method shows more efficient results with fewer prompt tokens compared to VPT-deep. The additional experimental results for FGVC with fewer tokens can be found in Table 7 in Appendix B.1.

Table 1. Classification results on FGVC. TOTAL PARAMS denotes the total number of parameters for all tasks including the backbone encoder ViT-B, prompt tokens and the task heads. Bold fonts denote the best performance in each benchmark.

SSL	METHOD	TOTAL PARAMS	CUB	FLOWERS	CARS	DOGS	NABIRDS	AVG
MAE	VPT-SHALLOW	1.02×	42.15	69.15	43.38	77.07	57.43	57.84
	VPT-DEEP	1.02×	68.33	**80.05**	67.67	78.83	65.22	72.02
	OURS	1.02×	**70.56**	78.55	**71.70**	**78.9**	**67.26**	**73.39**
MOCO v3	VPT-SHALLOW	1.02×	79.05	90.47	71.91	81.97	72.92	79.26
	VPT-DEEP	1.02×	82.67	**94.41**	**79.18**	83.33	75.99	**83.12**
	OURS	1.02×	**82.86**	93.71	79.02	**83.37**	**76.02**	83.00

Table 2. Classification results on VTAB-1K. TOTAL PARAMS denotes the total parameters for all tasks, including the backbone encoder ViT-B, prompt tokens, and the task heads. (†) denotes the reported performances in the original paper of VPT (Jia et al., 2022). Bold fonts denote the best performance in each benchmark.

SSL	METHOD	TOTAL PARAMS	NATURAL (7)	SPECIALIZED (4)	STRUCTURED (8)	AVG
MAE	VPT-SHALLOW	1.01×	39.96[†]	69.65[†]	27.50[†]	40.96[†]
	VPT-DEEP	1.04×	36.02[†]	60.61[†]	26.57[†]	37.22[†]
	OURS	1.01×	**47.61**	**76.86**	**36.80**	**49.22**
MOCO v3	VPT-SHALLOW	1.01×	67.34[†]	82.26[†]	37.55[†]	57.94[†]
	VPT-DEEP	1.01×	70.27[†]	83.04[†]	42.38[†]	61.22[†]
	OURS	1.01×	**74.84**	**83.38**	**49.10**	**65.80**

5.2.2. CLASSIFICATION ON VTAB-1K

With the VTAB-1K benchmark, we test our method's capability of driving the backbone encoder to capture generic visual concepts on three distinct groups of datasets. Table 2 shows the classification results on VTAB-1K. MAE with our approach largely outperforms the VPT-deep and VPT-shallow in all three groups. Also, our method provides substantial gains for MoCo v3 in *Natural* and *Structured* groups compared to VPT counterparts. Especially for both MAE and MoCo v3, we observe the largest performance boosts in *Structured* group, 10.23% and 6.72% from VPT-deep, respectively. This indicates that our method learns prompt tokens that enable SSL ViTs to transfer more effectively. It is effective not only with natural images but also in situations where image domains are different or when structural comprehension is required.

5.2.3. SEMANTIC SEGMENTATION ON ADE20K

To validate that the applicability of our method is not limited to image classification, we evaluate the semantic segmentation task on ADE20K. Following the settings in VPT (Jia et al., 2022), we used SETR-PUP (Zheng et al., 2021) for the segmentation model. As shown in Table 3, our method brings large performance gains from VPT-shallow for both MAE and MoCo v3. When compared to VPT-deep, MAE and MoCo v3 with our method also shows advanced re-

Table 3. Semantic segmentation results on ADE20K with SETR-PUP (Zheng et al., 2021) as the segmentation model. For VPT-deep, (×12) denotes that the same number of prompt tokens are used for each block of ViT-B/16. PT denotes prompt token. Bold fonts denote the best performance in each metric.

SSL	METHOD	# OF PTS	MIOU (SS)	MIOU (MS)
MAE	VPT-SHALLOW	100	34.20	35.23
	VPT-DEEP	10(×12)	37.76	38.80
	OURS	100	**38.44**	**39.81**
MOCO v3	VPT-SHALLOW	100	34.55	36.18
	VPT-DEEP	10(×12)	35.50	37.15
	OURS	100	**36.81**	**38.55**

sults. Note that in this experiment, our method beats all VPT counterparts while VPT-deep uses 20% more prompt tokens. This implies that our method utilizes prompt tokens more effectively in semantic segmentation tasks. In Appendix B.1, we present the results for ADE20K when using fewer prompt tokens in Table 6. In summary, as shown in the FGVC, VTAB-1K, and ADE20K benchmark results, our proposed method is a prompt-based transfer strategy that better exploits SSL ViTs for diverse vision tasks.

5.3. Additional Analysis

Analysis on the learned gates. As we explained in Eq. 8, our method is interpreted as a selective aggregation of the prompt representation from each block. Using the learned gate values, g^l, we are able to determine the contribution of each block to the prompt delivered to the last block. In Eq. 8, the weight applied to the output prompt representation of each block is expressed as follows:

$$\tilde{g}^l = \Big(\prod_{m=l+1}^{L-1} (1 - g^m) \Big) g^l, \ l = 1, \ldots, L-2 \tag{10}$$
$$\tilde{g}^{L-1} = g^{L-1}$$

Based on the weight $\tilde{g}^l$, we define the selection ratio, which represents the influence of each block on the prompt representation of the last block:

$$\mathbf{r}^l = \frac{\tilde{g}^l}{\sum_{m=1}^{L-1} \tilde{g}^m}. \tag{11}$$

Figure 4 shows the selection ratio calculated from models trained for different transfer scenarios. As shown in the figure, the selection ratio varies according to the downstream task and SSL method. For instance, the prompts learned for MAE focus primarily on the 11th block for the NABirds dataset while focusing almost equally on the 4th and 11th blocks for Stanford Cars dataset. In addition, we observe that the learned prompts focus on different blocks depending on the SSL ViT. On NABirds and Stanford Cars, learned prompts with MAE tend to focus on the 11th block, while with MoCo v3, they tend to focus on the 10th block. In other words, the gates guide the prompts to take into account the information change across the blocks that differs depending on the self-supervised method. In ADE20K segmentation, the learned prompts appear to have uniform ratios across the blocks. This indicates that the gates learn to make prompts to steer all the blocks since multi-level information is advantageous for segmentation tasks (Lin et al., 2017). These differences in selection ratios support our motivation that prompts should focus on different blocks so that they can be selectively affected by ViT blocks for effective task adaptation.

Adjusted Self-attention. As we discussed in Section 4.2, prompt tuning manipulates self-attention so that it steers the behavior of the pretrained ViTs. We visualize the self-attention map at the 3rd (early), 7th (middle), and 12th (late) blocks of SSL ViT with and without our method in Figure 5. MAE with prompt tuning attends to a different region in the image. Especially at the late block, MAE with Gated Prompt Tuning attends to the object's head while MAE attends to the object's boundary. In addition, when we added learnable temperatures, the resulting self-attention was different from when Gated Prompt Tuning was used alone. This suggests
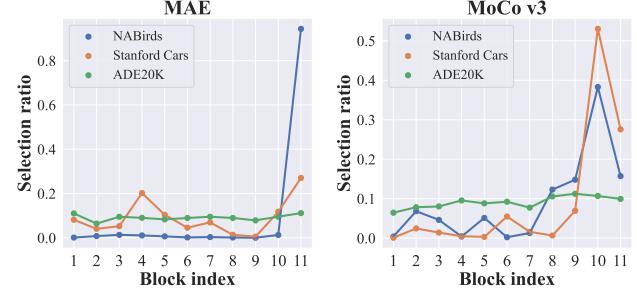

Figure 4. Selection ratio **r** on the NABirds, Stanford Cars fine-grained classification and ADE20K semantic segmentation. The selection ratio represents the influence of each block on the prompt representation of the last block.
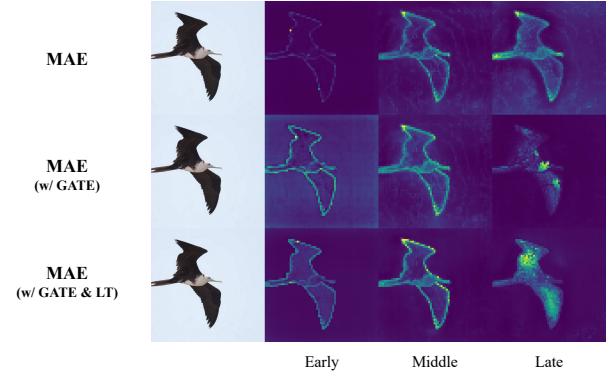

Figure 5. Visualization on self-attention map of ViT-B/16 blocks. Both prompt tuning and temperature scaling adjust the self-attention map from MAE. GATE denotes Gated Prompt Tuning and LT denotes Adaptive Attention Shaping with learnable temperatures.

that learnable temperature also plays a role in adjusting self-attention.

5.4. Ablation Studies

In this section, we conduct ablation studies on the efficacy of our Gated Prompt Tuning and Adaptive Attention Shaping. In Figure 6, we report the performance changes as our proposed components are added to VPT-shallow. Across all the benchmarks and SSL ViTs, our Gated Prompt Tuning consistently improves performance from VPT-shallow. This verifies that interaction with selective ViT blocks rather than all blocks boosts the strength of prompt tokens for task adaptation. Moreover, Adaptive Attention Shaping with learnable temperatures improves performances in almost all cases. These results support that adjusting self-attention score with adaptive temperature scaling aids prompts to encode improved instructions. The results on each individual dataset are shown in Appendix B.3.

To evaluate the effectiveness of gating operation, we apply the learnable hard gates implemented with Gumbel-
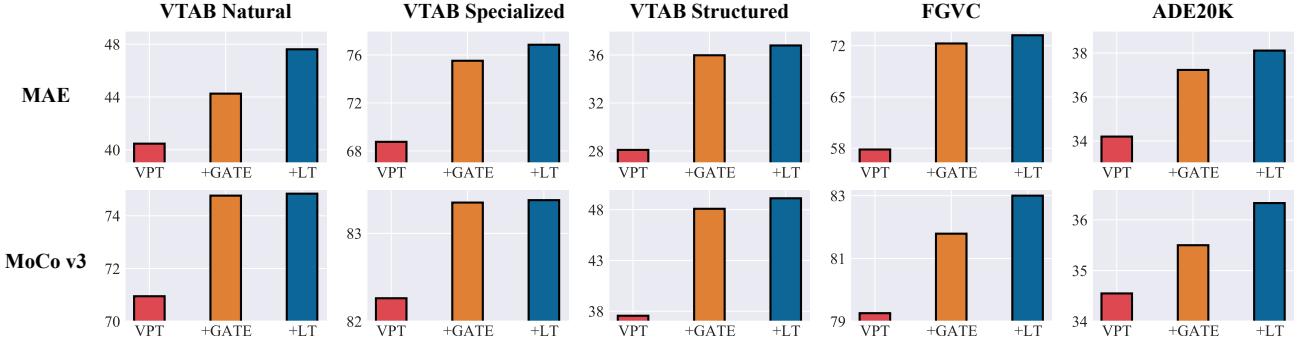
Figure 6. Ablation study across the benchmarks. For VTAB-1K and FGVC, we report average classification performance, and for ADE20K, we report semantic segmentation performance. GATE denotes Gated Prompt Tuning and LT denotes Adaptive Attention Shaping with learnable temperatures.
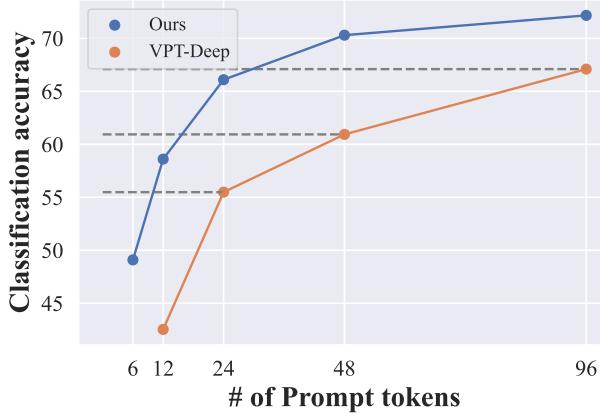

Figure 7. Performance comparison between VPT-deep and our method under the same number of prompt tokens. We used MAE as the SSL ViT backbone and evaluated it on the Stanford Cars dataset. For VPT-deep, using 12, 24, 48, and 96 prompt tokens denotes that 1, 2, 4, and 8 prompt tokens are inserted into each block of ViT-B/16.

Sigmoid (Geng et al., 2020; Jang et al., 2016) to our proposed method for CUB and OxfordFlowers classification. As shown in Table 4, using hard gates implemented with Gumbel-Sigmoid outperforms VPT-shallow. This indicates that selective interaction with ViT blocks is effective in task-adaptation using prompt tokens. Our method with soft gates shows improved performances compared to using hard gates. This is because soft gates enable prompt tokens to interact partially with blocks if there is any desirable factor for the task. On the other hand, hard gates would lead to suboptimal results since they exclude the entire block even though it is partially task-relevant.

In addition, we investigate the parameter efficiency of our method using MAE on the Stanford Cars dataset compared to VPT-deep by varying the number of input prompt tokens.

Table 4. Ablation study on the gate. We used MAE as Self-supervised ViT. PT denotes prompt token and GH denotes the hard gate implemented with Gumbel-Sigmoid function.

METHOD	# OF PTS	CUB	FLOWERS
VPT-SHALLOW	100	42.15	69.15
OURS (W/ GH)	100	65.46 (+23.31)	76.55 (+7.4)
OURS	100	**70.56** (+28.41)	**78.55** (+9.4)

In Figure 7, our method outperforms VPT-deep with only half the number of prompt tokens in all cases. For example, by using only 24 prompt tokens, our method outperforms VPT-deep with 48 prompt tokens (66.1% vs. 60.9%). For VPT-deep, it is impossible to use fewer than 12 prompt tokens because it requires at least one prompt token for each ViT block, but our method can handle the number of prompt tokens fewer than 12 and still outperform with fewer prompt tokens. In Appendix B.1, we provide additional experimental results on the FGVC and ADE20K benchmarks, which demonstrate that our Gated Prompt Tuning employs prompt tokens efficiently for task adaptation.

6. Related Work

6.1. Self-supervised Vision Transformers

Self-supervised Vision Transformers have proven to be an excellent pretrained backbone for computer vision tasks (Bao et al., 2021; He et al., 2022; Chen et al., 2021; Xie et al., 2022; Zhou et al., 2021; Caron et al., 2021). MoCo v3 (Chen et al., 2021) and DINO (Caron et al., 2021) are representative of the instance-based approach, in which they learn representations that are invariant over random transformations. Masked image modeling (He et al., 2022; Bao et al., 2021; Zhou et al., 2021; Xie et al., 2022), which learns representations by recovering randomly masked patches, is

another promising branch of self-supervised learning. Utilizing these models for diverse computer vision tasks is a promising strategy, as they have demonstrated excellent transferability and high performance (He et al., 2022; Chen et al., 2021; Caron et al., 2021; Bao et al., 2021; Zhou et al., 2021). However, the tuning methods for self-supervised Vision Transformers during their transfer to downstream vision tasks, particularly prompt-based tuning, have been less explored.

6.2. Transfer Learning

Transfer learning aims to efficiently utilize pretrained neural networks for a wide range of downstream tasks. The most basic approach, full fine-tuning, involves training both the pretrained backbone and the task-specific network. Recently, considerable research has been conducted on tuning large pretrained models in a parameter-efficient manner (Jia et al., 2022; Houlsby et al., 2019; Cai et al., 2020; Chen et al., 2022; Bahng et al., 2022; Li & Liang, 2021; Lester et al., 2021; Huang et al., 2023). Cai et al. (2020) proposed to freeze the weights and only update the bias of the pretrained models. Chen et al. (2022) introduced an additional lightweight module, known as AdaptMLP, in an MLP module of ViT blocks and fine-tune it. Among these, prompt tuning employs learnable perturbations in the embedding space or pixel space (Jia et al., 2022; Bahng et al., 2022). However, since these mainly deal with supervised pretrained models, there are few studies on parameter-efficient tuning for self-supervised ViTs. In this work, we develop a prompt-based transfer learning method for self-supervised ViTs.

6.3. Discussion

Our proposed method can be interpreted as performing a scaling operation on prompt tokens using a gate. There are previous works that utilize scaling operations, such as AdaptFormer (Chen et al., 2022), to achieve efficient fine-tuning of Vision Transformers. However, our work and AdaptFormer diverge in two aspects. First, the location of the scaling operation in AdaptFormer is different from that of our gating operation. In AdaptFormer, the scaling operation is applied to the patch representation, emanating from an additional branch in the MLP module. In contrast, in our proposed method, the gating operation is exclusively conducted for the prompt token and is situated between the Transformer blocks. Second, the purpose of scaling differs. While the scaling operation in AdaptFormer seeks to balance task-agnostic and task-specific features from two distinct branches within each block, our gating operation is designed to regulate the interaction between prompt tokens and each block. To the best of our knowledge, the approach we propose represents one of the first implementations of a gating operation in prompting for computer vision tasks.

In NLP, a concurrent study, known as Prompt Gating (Huang et al., 2023), was recently reported that makes use of trainable gates. This approach aims to combine independently trained prefixes, each learned separately for single-aspect text generation, to enable controllable multiple-aspect text generation during inference. A problem arises in this context where the independently trained prefixes for each aspect interact in an attention sublayer, causing mutual interference and thereby reducing controllability. Prompt Gating addresses this issue by introducing gates that rescale the prefixes for each aspect, adjusting the magnitudes of the prefixes for each aspect within the attention sublayer to alleviate mutual interference. Our approach differs from Prompt Gating in terms of the motivation, objectives, and the implementation of the gating operation. First, our work is driven by the observation that self-supervised ViTs encode more information in the blocks compared to supervised ViTs, thereby making it challenging for prompts to focus on task-relevant blocks. Second, our work aims to facilitate focused interactions between the prompt and the task-relevant blocks, as opposed to adjusting the interaction between multiple task prompts. To accomplish this, our gating operation functions by creating a convex combination of the output prompt representations from the previous block and the current block. In contrast, Prompt Gating merely scales the attention hidden state of the prompt in the current block.

7. Conclusion

In this work, we propose an enhanced prompt-based transfer method for self-supervised ViTs. The task-relevant blocks in the pretrained ViTs depend on pretraining methods and downstream tasks. To address this, we introduce Gated Prompt Tuning, which adopts learnable gates and directs the prompt to selectively focus on task-relevant blocks for effective task adaptation. Furthermore, we introduce Adaptive Attention Shaping, which adjusts the attention score and further enhances task-specific instruction with prompts. Extensive experimental results across diverse benchmarks confirm that our proposed method more effectively utilizes prompt tokens for task adaptation.

Acknowledgements

References

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bahng, H., Jahanian, A., Sankaranarayanan, S., and Isola, P. Visual prompting: Modifying pixel space to adapt pre-trained models. *arXiv preprint arXiv:2203.17274*, 2022.

Bao, H., Dong, L., and Wei, F. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Bordes, F., Balestriero, R., and Vincent, P. High fidelity visualization of what your self-supervised representation knows about. *arXiv preprint arXiv:2112.09164*, 2021.

Cai, H., Gan, C., Zhu, L., and Han, S. Tinytl: Reduce memory, not parameters for efficient on-device learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., and Luo, P. Adaptformer: Adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*. PMLR, 2020.

Chen, X., Xie, S., and He, K. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Gebru, T., Krause, J., Wang, Y., Chen, D., Deng, J., and Fei-Fei, L. Fine-grained car detection for visual census estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11), 2013.

Geng, X., Wang, L., Wang, X., Qin, B., Liu, T., and Tu, Z. How does selective mechanism improve self-attention networks? *arXiv preprint arXiv:2005.00979*, 2020.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Hambardzumyan, K., Khachatrian, H., and May, J. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*. PMLR, 2019.

Huang, X., Liu, Z., Li, P., Li, T., Sun, M., and Liu, Y. An extensible plug-and-play method for multi-aspect controllable text generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jia, M., Tang, L., Chen, B.-C., Cardie, C., Belongie, S., Hariharan, B., and Lim, S.-N. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022.

Khosla, A., Jayadevaprakash, N., Yao, B., and Li, F.-F. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, volume 2. Citeseer, 2011.

Kriegeskorte, N. Deep neural networks: a new framework for modelling biological vision and brain information processing. *biorxiv*, 2015.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553), 2015.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Nilsback, M.-E. and Zisserman, A. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. IEEE, 2008.

Qin, G. and Eisner, J. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

Schmidhuber, J. Deep learning in neural networks: An overview. *Neural networks*, 61, 2015.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Van Horn, G., Branson, S., Farrell, R., Haber, S., Barry, J., Ipeirotis, P., Perona, P., and Belongie, S. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The caltech-ucsd birds-200-2011 dataset. 2011.

Xie, Z., Zhang, Z., Cao, Y., Lin, Y., Bao, J., Yao, Z., Dai, Q., and Hu, H. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A. S., Neumann, M., Dosovitskiy, A., et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

Zhao, N., Wu, Z., Lau, R. W., and Lin, S. What makes instance discrimination good for transfer learning? *arXiv preprint arXiv:2006.06606*, 2020.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A., and Kong, T. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021.

A. Implementation Details

A.1. Pseudo code

We provide a Pytorch-like pseudo code for our proposed Gated Prompt Tuning in Algorithm 1. The gating operation is implemented by adding a few additional lines of code in the block operation of the Vision Transformer.

Algorithm 1 PyTorch-like Pseudocode for Gated Prompt Tuning

```
# cls: CLS token
# x: patch tokens
# p: prompt tokens
# gamma: gate priors
# blocks: ViT blocks
# N_p: number of prompt tokens

# prepend prompt tokens
x = cat([cls, p, x], dim=1)
for i, blk in enumerate(blocks):
   if i == len(blocks) - 1:
      x = blk(x)
   else:
      # compute gate values
      gate = gamma[i].sigmoid()
      # input prompt representation of i-th block
      prompt_before_block = x[:, 1: 1+N_p, :]

      x = blk(x)
      # output prompt representation of i-th block
      prompt_after_block = x[:, 1: 1+N_p, :]
      # gated prompt representation
      gated_prompt = gate * prompt_after_block + (1 - gate) * prompt_before_block

      # pass the gated prompt representation to the next block
      x = cat([
          x[:, 0:1, :],
          gated_prompt,
          x[:, 1+N_p:, :]
      ], dim=1)
```

A.2. Selected Hyperparameters

The hyperparameters used to train the models for FGVC (Table 1), VTAB-1K (Zhai et al., 2019) (Table 2), and ADE20K (Table 3) are listed in Table 5. We used the SGD optimizer, and the learning rate was searched among {0.05, 0.1, 0.25, 0.5, 1.0, 2.5, 5.0}. For ADE20K semantic segmentation, we used the default hyperparameters following SETR-PUP (Zheng et al., 2021).

In Figure 1, for CUB, we used a learning rate 0.1 for MAE and Supervised ViT, and a learning rate 1.0 for MoCo v3. For KITTI, we used a learning rate 0.1 for all models. In Figure 7, learning rates used for each model training are the same as those used in Table 1.

Table 5. Selected hyper-parameters of our method for each downstream task and SSL method. BS denotes batch size, PT denotes prompt token, LR denotes learning rate and GATE INIT. denotes the initialized gate prior.

	MAE				MoCo v3			
	BS	# OF PTs	LR	GATE INIT.	BS	# OF PTs	LR	GATE INIT.
CALTECH101	128	1	0.5	5	128	10	2.5	4
CIFAR-100	128	1	0.25	5	128	10	1	15
CLEVR_DIST	128	1	0.1	15	128	5	1	10
CLEVR_COUNT	128	1	0.05	15	128	5	0.5	5
RETINOPATHY	128	5	0.05	10	128	5	0.5	5
DMLAB	128	1	0.1	5	128	1	1	5
DSPR_ORI	128	1	0.05	5	128	5	0.5	5
DSPR_LOC	128	1	0.1	5	128	5	0.5	5
DTD	128	1	0.25	1	128	10	1	5
EUROSAT	128	5	0.1	1	128	10	1	5
KITTI-DIST	128	5	0.1	7	128	5	1	5
FLOWERS102	128	1	0.25	5	128	5	1	7
PETS	128	1	0.5	10	64	100	1	1
CAMELYON	128	1	0.05	5	128	1	1	10
RESISC45	128	10	0.25	5	64	100	1	7
SNORB_AZIM	128	1	0.1	20	128	10	0.5	3
SNORB_ELEV	128	1	0.05	10	128	5	0.5	4
SUN397	128	1	0.5	10	128	1	1	4
SVHN	128	1	0.1	10	128	5	1	10
CUB	64	100	0.1	5	64	100	1	5
FLOWERS	64	100	0.1	15	64	100	1	10
CARS	64	100	0.25	5	64	100	0.5	5
DOGS	64	100	0.5	15	64	100	0.5	10
NABIRDS	64	100	0.5	5	64	100	1	5
ADE20K	16	100	0.001	10	16	100	0.001	10

B. Additional Experiments

B.1. Experiments with Fewer Prompt Tokens

To demonstrate that our Gated Prompt Tuning employs prompt tokens efficiently for task adaptation, we conducted additional experiments on FGVC image classification and ADE20K semantic segmentation. For ADE20K semantic segmentation we used 24 prompt tokens. In Table 6, our proposed method still outperforms VPT-deep when using fewer tokens.

Table 6. Semantic segmentation results on ADE20K with fewer tokens. PT denotes prompt token.

SSL	METHOD	# OF PTs	MIOU (SS)	MIOU (MS)
MAE	VPT-SHALLOW	24	34.77	35.93
	VPT-DEEP	24	37.03	38.25
	OURS	24	**38.13**	**39.23**
MOCO V3	VPT-SHALLOW	24	34.75	36.34
	VPT-DEEP	24	35.96	37.47
	OURS	24	**37.02**	**38.57**

For the FGVC classification, we evaluated the performance of our proposed method and VPT-deep using 24, 48, and 96 prompt tokens. We utilized MAE as the self-supervised ViT in this experiment. We observed that the average performance gap between our method and VPT-deep widens when using fewer tokens in Table 7. The hyperparameters used for each model training are the same as those used in Table 1, except for the number of prompt tokens.

Table 7. Classification results on FGVC with fewer tokens. PT denotes prompt token.

# OF PTs	METHOD	CUB	FLOWERS	CARS	DOGS	NABIRDS	AVG
24	VPT-DEEP	59.68	67.51	57.68	**77.91**	54.65	63.49
	OURS	**65.26**	**73.80**	**66.09**	75.72	**62.10**	**68.59** (+5.10)
48	VPT-DEEP	66.43	71.31	64.06	**78.89**	60.62	68.26
	OURS	**69.00**	**74.71**	**70.29**	76.77	**63.73**	**70.90** (+2.64)
96	VPT-DEEP	70.85	**76.29**	67.63	**78.38**	62.99	71.23
	OURS	**71.40**	75.96	**72.16**	77.81	**66.25**	**72.72** (+1.49)

B.2. Results on Alternative Vision Transformer Backbones

In this section, we provide experimental results on other Vision Transformer variants other than ViT-B. Due to the lack of public pretrained models, we used MoCo v3 for ViT-S and MAE for ViT-L. We performed experiments on CUB and OxrfordFlowers classification. The pretrained model checkpoints are obtained from the official repositories of MoCo v3 and MAE (Chen et al., 2021; He et al., 2022).

In Table 8a and Table 8b, our method outperforms VPT in both CUB and OxfordFlowers datasets in all the ViT variants. In particular, for ViT-L (MAE), there was a significant performance gap between our method and VPT-deep.

Table 8. CUB and OxfordFlowers classification results on ViT variants. PT denotes prompt token.

(a) ViT-L (MAE)

	#OF PTs	CUB	FLOWERS
VPT-SHALLOW	48	39.26	62.77
VPT-DEEP	48	70.54	65.44
OURS	48	**72.99**	**74.71**

(b) ViT-S (MoCo v3)

	#OF PTs	CUB	FLOWERS
VPT-SHALLOW	48	68.62	84.65
VPT-DEEP	48	73.18	89.53
OURS	48	**73.3**	**91.14**

B.3. Detailed Ablation Studies

In this section, we provide ablation results of our proposed method on each individual dataset in FGVC, VTAB-1K, and ADE20K in Figure 8.

(a) MAE

(b) MoCo v3

Figure 8. Ablation study across the benchmarks. GATE denotes Gated Prompt Tuning and LT denotes Adaptive Attention Shaping with learnable temperatures.

C. Empricial Observations

In this section, we present more qualitative and quantitative experimental results of Deep Image Prior (DIP) (Ulyanov et al., 2018) applied on the representations of each block in the pretrained ViTs. For supervised ViT, we used pretrained model checkpoint from the official repository of Visual Prompt Tuning (Jia et al., 2022). In Figure 10 and Figure 11, it can be observed that the DIP results of each block differ for supervised ViT, MoCo v3, and MAE for both the CUB and OxfordFlowers datasets.

C.1. PSNR and SSIM scores

As a quantitative result, we measured the PSNR and SSIM scores of 100 images generated by DIP for each block of the pretrained ViTs. As shown in Figure 9, it can be inferred that self-supervised ViTs retain relatively more information, even in the later blocks, resulting in higher-quality reconstructed images.
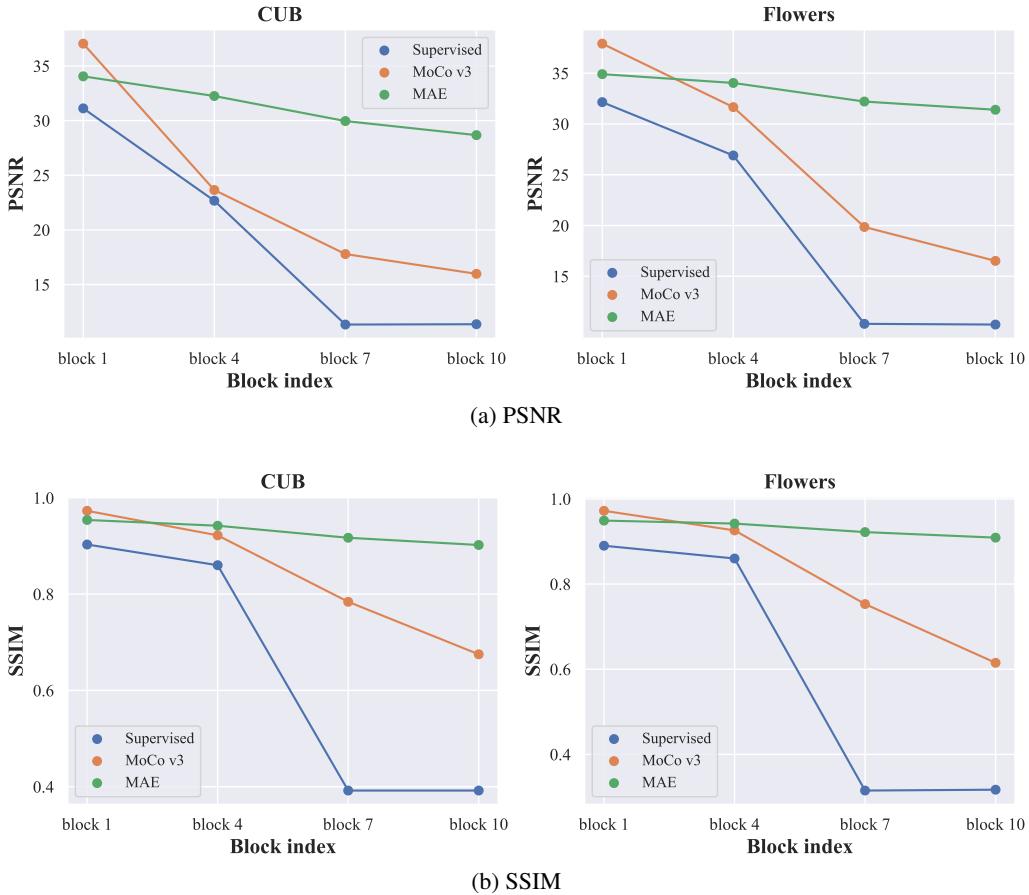

(a) PSNR

(b) SSIM

Figure 9. PSNR and SSIM scores for the reconstructed images using Deep Image Prior.

C.2. Deep Image Prior (DIP) results

Figure 10. Reconstructed images using Deep Image Prior (DIP) with pretrained ViT block's representation as a training target. **Row 1**: original image. **Rows 2-4**: reconstruction results for each pretrained ViTs.
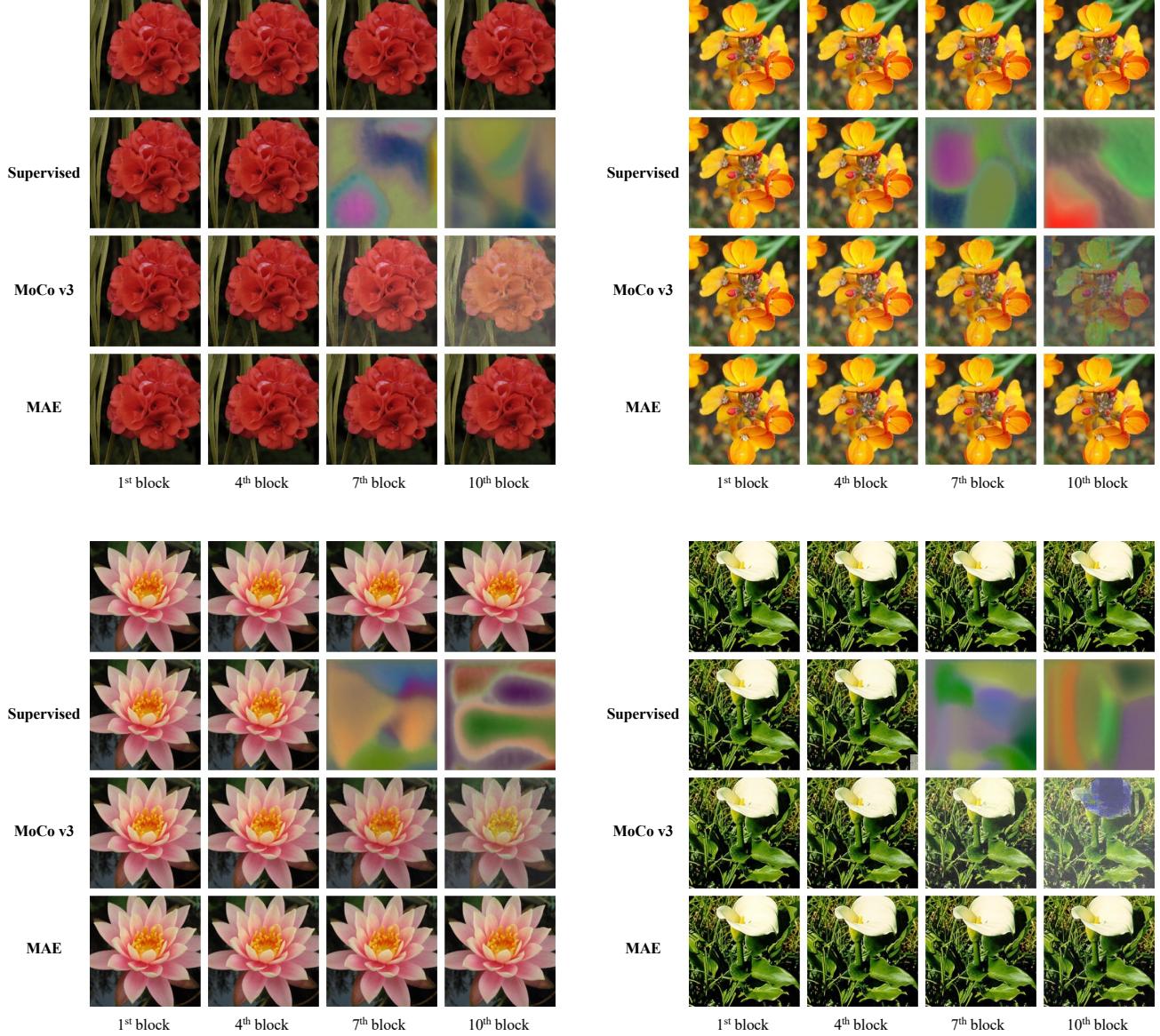
Figure 11. Reconstructed images using Deep Image Prior (DIP) with pretrained ViT block's representation as a training target. **Row 1**: original image. **Rows 2-4**: reconstruction results for each pretrained ViTs.