E²VPT: An Effective and Efficient Approach for Visual Prompt Tuning

Cheng Han[1], Qifan Wang[2], Yiming Cui[3], Zhiwen Cao[4], Wenguan Wang[5], Siyuan Qi[6], Dongfang Liu[1]*

Rochester Institute of Technology[1], Meta AI[2], University of Florida[3], Purdue University[4]

Zhejiang University[5], BIGAI[6]†

{ch7858, dongfang.liu}@rit.edu, wqfcr@fb.com, cuiyiming@ufl.edu, cao270@purdue.edu

wenguanwang.ai@gmail.com, syqi@bigai.ai

Abstract

As the size of transformer-based models continues to grow, fine-tuning these large-scale pretrained vision models for new tasks has become increasingly parameter-intensive. Parameter-efficient learning has been developed to reduce the number of tunable parameters during fine-tuning. Although these methods show promising results, there is still a significant performance gap compared to full fine-tuning. To address this challenge, we propose an Effective and Efficient Visual Prompt Tuning (E²VPT) approach for large-scale transformer-based model adaptation. Specifically, we introduce a set of learnable key-value prompts and visual prompts into self-attention and input layers, respectively, to improve the effectiveness of model fine-tuning. Moreover, we design a prompt pruning procedure to systematically prune low importance prompts while preserving model performance, which largely enhances the model's efficiency. Empirical results demonstrate that our approach outperforms several state-of-the-art baselines on two benchmarks, with considerably low parameter usage (e.g., 0.32% of model parameters on VTAB-1k). Our code is available at https://github.com/ChengHan111/E2VPT.

1. Introduction

The development of artificial intelligence (AI) should not only prioritize performance advances, but also emphasize sustainable deployment [64, 78, 80, 87]. Despite the captivating pursuit of performance improvements in visual-related tasks, the size of present models has been rapidly increasing, resulting in energy-intensive and computationally expensive training [31, 73, 92]. Transformer-based architectures currently dominate visual-related models, such as ViT-Huge [12] (632M) and Swin-Large [54] (197M), with significantly more parameters than the Convolutional
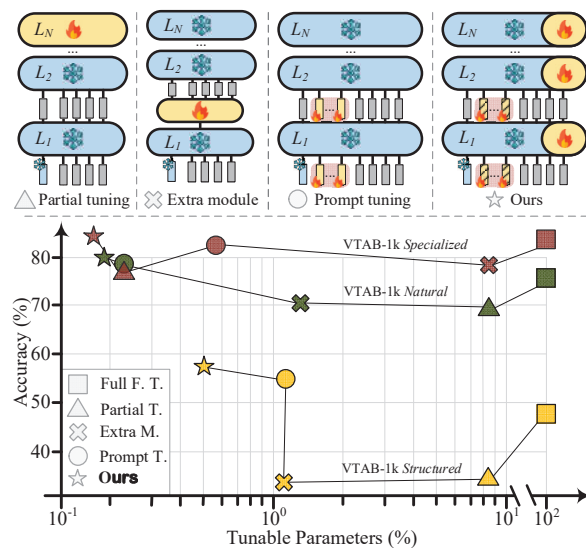
---

*Corresponding author

†National Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence

Figure 1. **E²VPT (ours)** *vs* **concurrent arts** (*i.e.*, △ partial tuning [91], ⊠ extra module [6], and ◯ prompt tuning [34] methods) under *pretrain-then-finetune* paradigm. Our method yields solid performance gains over state-of-the-art fine-tuning methods and competitive to full fine-tuning on a wide range of classification tasks adapting the pretrained ViT-Base/16 [12] as backbone with considerable lower parameter usage (see Table 1). ■■■ colors represent results on VTAB-1k [96] *Specialized*, *Natural* and *Structure*, respectively.

Neural Networks (CNN) like ResNet [26] (25M). Training such large models from scratch presents challenges such as limited data [5, 20, 75] and slow convergence at low accuracy [37, 47]. A common paradigm to overcome these challenges is *pretrain-then-finetune*, which reduces the need for vast amounts of training data and speeds up processing of various visual tasks. However, the traditional *full fine-tuning* involves storing and deploying a complete copy of the backbone parameters for every single task [34], which remains computationally expensive and not suitable for fast model deployment.

To address this issue, various approaches have been de-

veloped, which can be divided into three main categories (see Fig. 1): partial tuning, extra module, and prompt tuning methods. *Partial tuning* methods [10, 35, 58] only fine-tune part of the backbone, such as the classifier head or last few layers, while freezing the others. *Extra module* methods insert learnable bias term [6] or additional adapters [70, 98] to the network for adaptation. *Prompt tuning* methods add prompt tokens [34, 36, 94] to the input layer of the transformer without changing or fine-tuning the backbone itself. All of these methods operate within the *pretrain-then-finetune* paradigm, which reduces the number of learnable parameters compared to full fine-tuning [10, 35, 58, 70, 98]. However, despite achieving promising results, there are two main limitations in existing parameter-efficient methods. **Firstly**, they do not scrutinize the core architecture of the transformer's self-attention mechanism, resulting in a large performance gap with full fine-tuning. **Secondly**, they usually need to fine-tune a relatively large number of parameters to achieve reasonable performance and fail to explore the extremes of parameter efficiency.

The perspective outlined above leads to two fundamental questions: ❶ *How can we establish the **effectiveness** of prompt tuning for large-scale transformer-based vision models?* ❷ *How can we explore the extremes of parameter **efficiency** to reduce the number of tunable parameters?* These two questions are the foundation of our work. The intuition is that instead of solely focusing on modifying inputs, as in previous prompt tuning methods, we should explicitly investigate the potential of improving the self-attention mechanism during fine-tuning, and explore the extremes of parameter efficiency.

In response to question ❶, we discuss and analyze the self-attention mechanism of the transformer, which is crucial in capturing long-range token dependencies within a global context [21, 38, 49]. In additional to the input visual prompts, we introduce learnable key-value prompts and integrate them into the Key and Value matrices in the self-attention layers. The key-value prompts are jointly learned with the input visual prompts during fine-tuning. This approach effectively leverages the well-designed prompt architecture of the transformer, resulting in significant performance improvements. Moreover, it provides a generic plug-and-play prompt module for current transformer architectures, and its fine-tuning solution is conceptually different from all aforementioned arts in the vision domain.

Motivated by ❷, we propose a pruning strategy to further reduce the number of parameters while maintaining the model performance. Our approach draws inspiration from the lottery ticket hypothesis (LTH) [16, 102], which posits that for a given task, there exists a sub-network that can match the test accuracy of the original over-parameterized network without the unnecessary weights [22, 23, 41, 43, 44]. Building on this paradigm, we revisit the core design

of prompt tuning methods and further reduce the number of learnable parameters. Specifically, we aim to retain the prompt tokens that contribute significantly to the performance, while pruning the prompt tokens that are redundant or unnecessary during fine-tuning. By pruning these unnecessary prompts, we can significantly improve the prompt tuning efficiency while maintaining the performance.

To answer question ❶-❷, we propose E^2VPT, namely **E**ffective and **E**fficient **V**isual **P**rompt **T**uning. E^2VPT is a novel prompt tuning framework that is both architecture-aware and pruning-anchored (see Fig. 1). In §2, we conduct a literature review and discuss relevant works. Our proposed approach is presented in §3, where we describe in detail how we design visual and key-value prompts to achieve superior performance with fewer parameters. In §4, we present compelling experimental results on various benchmarks, backbones, and different pretraining objectives. Specifically, our approach achieves an average improvement of **5.85%** in accuracy on VTAB-1k compared to full fine-tuning, and **1.99%** compared to VPT [34]. Moreover, our approach uses considerably fewer learnable parameters than existing methods, accounting for an average of only **0.32%** of the backbone parameters on VTAB-1k, whereas VPT on average requires 0.68% (see Fig. 1). We further demonstrate and explain the superiority of our approach over VPT with hyperbolic visualization. Finally, we demonstrate the strong algorithmic generalization of our approach to the language domain in the Appendix. We trust that this work provides valuable insights into related fields.

2. Related Work

2.1. Vision Transformers

Inspired by the remarkable success of transformers in natural language processing (NLP) [5, 11, 52, 69, 79, 83], researchers have extended the transformer architecture to various supervised vision tasks, including image classification [12, 53, 54, 56], image segmentation [46, 51, 74, 82, 84, 86, 100], object detection [4, 7, 50, 66, 93, 101] and pose estimation [29, 30, 48, 90]). Self-supervised pretraining paradigms [3, 10, 24] has also been explored, leading to state-of-the-art results. transformers dominate in visual-related disciplines due to their superior performance and scalability compared to convolutional neural networks (CNNs) [27, 34]. However, the significant computational and parameter overhead required to adapt transformers to various vision tasks cannot be ignored [15, 33, 97]. For instance, recent transformer-based models such as MViTv2-Large [45] (218M), ViT-G [95] (1.8B), SwinV2-G [53] (3.0B), and V-MoE [72] (14.7B) incur substantial computational costs. Therefore, we propose E^2VPT, which is designed to reduce the computational cost of transformer-based architectures while maintaining high performance in the *pretrain-then-finetune* paradigm.

2.2. Parameter-efficient Fine-tuning

Efficient model training has drawn much attention in the vision community, particularly with the rise of Vision Transformers [1, 8, 12, 54, 85]. However, despite their effectiveness and widespread use, these models are often too large for practical deployment and adaptation. As a result, the *pretrain-then-finetune* paradigm is commonly employed. While full fine-tuning ensures strong performance, it is an expensive approach that involves updating all network parameters [27, 75]. To overcome this challenge, researchers are exploring alternatives that balance parameter-efficiency and robust performance, which can be broadly categorized into three groups: *partial tuning*, *extra module* and *prompt tuning* methods.

Partial tuning methods are widely used for parameter-efficient fine-tuning. These methods involve freezing most of the backbone and only fine-tune a small portion of the parameters, such as linear [32] or MLP heads [9], or a few blocks/layers of the backbone [24, 65, 91, 99]. While these methods are straightforward and simple to implement [10, 35, 58], they often have a large performance gap compared to full fine-tuning. *Extra module* methods design additional learnable plug-in architecture for fine-tuning. For example, the work in [98] introduces a side structure alternatively while freezing the original network. The works in [6, 70] insert additional residual units into the backbone. However, one drawback of these methods is that the inserted modules are often customized for specific architectures and might not be generalized to others. Additionally, these modules usually consume even more parameters compared to partial tuning methods. *Prompt tuning* or prompting [28, 42, 57, 89] has been originally proposed for fast model adaptation in the language domain. These methods prepend a set of learnable vectors to the input of the backbone and only update these task-specific prompts during fine-tuning. Recently, visual-related prompting [18, 34, 88] is introduced in vision domain, which designs visual prompts in the input sequence and shows competitive performance with full fine-tuning. However, current methods do not consider the inner design of transformer-based architectures, resulting in less effective prompting solutions. In contrast, our approach is mindful of architecture and anchored on pruning, which conceptually sets it apart from the methods discussed above.

3. Our E²VPT Approach

In this section, we introduce E²VPT, a novel visual prompt tuning approach for effective and efficient large-scale transformer-based model fine-tuning. We first define the problem and notations in §3.1. The effective prompt tuning with the designing of visual and key-value prompts is presented in §3.2, followed by the efficient prompt prun-

ing in §3.3. The overall framework is shown in Fig. 2.

3.1. Problem Definition

In this section, we define the problem of E²VPT and provide the notations. Assuming we have a backbone vision transformer model $\mathbf{T}$, which is pretrained on a large set of data and tasks. The input to the vision transformer is a sequence of image patches $I = \{I_1, I_2, \ldots, I_m\}$, where m is the total number of image patches. Each patch is then projected into a d-dimensional embedding with positional encoding, *i.e.*, $E = \{E_j | 1 \leq j \leq m\}$ with $E_j = \mathrm{Emb}(I_j)$. The vision transformer $\mathbf{T}$ consists of N identical transformer layers, represented as:

$$Z^1 = L_1(E)$$
$$Z^i = L_i(Z^{i-1}) \quad i = 2, 3, \ldots, N \tag{1}$$

here each transformer layer is a stack of multi-head self-attention (MSA) and feed-forward network (FFN):

$$L(\cdot) = \mathrm{FFN}\,(\,\mathrm{MSA}\,(\cdot)\,) \tag{2}$$

Given a new vision task, the objective is to fine-tune a model $\hat{\mathbf{T}}$ that can deliver good performance on the task, while only tuning a small amount of parameters. In the context of visual prompt tuning, $\hat{\mathbf{T}} = \{\mathbf{T}, \mathbf{P}\}$ which includes a frozen backbone $\mathbf{T}$, and trainable prompts $\mathbf{P}$ with very few tunable parameters.

3.2. Effective Prompting

Most existing prompt tuning approaches focus on tuning a set of visual prompts by prepending them to the input sequence in transformer layers, without considering the internal design of transformer architectures. However, to enhance the effectiveness of prompt tuning and achieve optimal fine-tuning performance, we propose a new approach that incorporates a set of key-value prompts ($\mathbf{P}_K$ and $\mathbf{P}_V$) in addition to input visual prompts ($\mathbf{P}_I$) within our visual prompt tuning framework. Intuitively, the input visual prompts are inserted to the input sequence of each encoder layer, which learn to represent of the new task. The key-value prompts are concatenated with the key and value parameter matrices in the self-attention module, which learn to capture the new attention pattern from the data.

Visual Prompts. Visual prompts are a set of d-dimensional embedding vectors that have the same dimensionality with the input visual tokens. They are prepended to the input sequence of each transformer encoder layer and interact with all the input tokens. Visual prompts play a similar role to those prompt tokens in traditional prompt tuning methods [34, 42], which learn task-specific embeddings to guide the model performing on the new task.

Formally, these visual prompts are defined as $\mathbf{P}_I = \{P_I^1, P_I^2, \ldots, P_I^N\}$, where P_I^i denotes the learnable visual
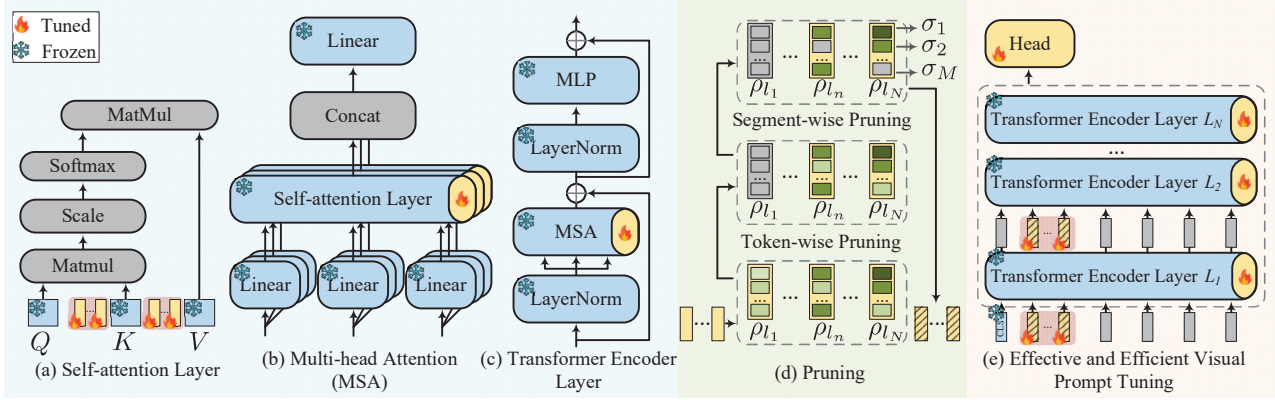
Figure 2. **Overview of our E^{2}VPT framework.** Under the *pretrain-then-finetune* paradigm, only the prompts in the transformer's input and backbone (§3.2), are updated during the fine-tuning process, while all other components remain frozen. We further introduce pruning (§3.3) at two levels of granularity (*i.e.*, token-wise and segment-wise) in (d) to eliminate unfavorable input prompts during rewinding.

prompts in the i_{th} encoder layer, and N is the total number of layers. Then the encoder layers are represented as:

$$
\begin{aligned}
Z^1 &= L_1(P_I^1,\ E) \\
Z^i &= L_i(P_I^i,\ Z^{i-1}) \quad i = 2, 3, \ldots, N
\end{aligned}
\tag{3}
$$

where Z^i represents the contextual embeddings computed by the i_{th} encoder layer. The different colors indicate trainable and frozen parameters, respectively. For the embeddings of the input image patches E, they are initialized with frozen Emb projection from the backbone.

Key-Value Prompts. Visual prompts are useful in learning knowledge about new tasks. However, they are insufficient in guiding information interaction within transformer encoder layers. The reason is that when fine-tuning on new data, the image distribution may significantly differ from those in the image examples used for pretraining the backbone model. As a result, it is crucial to enhance the model's capability to capture new information from the fine-tuning data and conduct more effective attention among input tokens to learn new patterns.

To this end, we propose a new approach by introducing a novel set of key-value prompts, P_K and P_V, which are incorporated into the attention module within each encoder layer (as shown in Fig. 2(a). These key-value prompts are small matrices that have only a few columns but share the same number of rows as the key and value matrices in the original attention module. To perform the new attention computations, the key and value matrices are concatenated with their corresponding P_K and P_V prompts, respectively. This process is defined as follows:

$$
\begin{aligned}
L(\cdot) &= \text{FFN}\left(\text{MSA}\left(\cdot\right)\right) \\
\text{MSA}(\cdot) &= \text{concat}(\text{softmax}(\frac{Q_h {K_h'}^T}{\sqrt{d}})V_h')
\end{aligned}
\tag{4}
$$

where FFN is the feed-forward network and MSA is the multi-head attention inside the encoder layer. h represents the h_{th} head. K' and V' are the new key and value embedding matrices defined as:

$$
K' = \text{concat}(K,\ P_K), \quad V' = \text{concat}(V,\ P_V) \tag{5}
$$

where K and V represent the original key and value matrices in the backbone. In this way, the key-value prompts can help guide the model adaptation to the new data. In our implementation, we take it a step further by enabling parameter sharing of the P_K and P_V prompts within each transformer layer instead of tuning separate learnable vectors. Our motivation is twofold: First, our experimental results show that with the shared prompts, the fine-tuning performance consistently improves across instances; Second, using shared prompt vectors reduces the parameter usage in the learnable transformer part by half, making it more parameter-efficient. We provide discussion on exploring the prompt locations (*i.e.*, before or after K and V) in §4.3.

It is worth noting that the query matrix Q is another critical element in the self-attention mechanism. However, additional prompting on Q is not desired for two reasons: First, prompting on Q is similar to prepending on K for computing attention scores between each pair of Q and K Therefore, prompting on both Q and K is unnecessary; Second, changes in Q affect the output shape of the attention map, necessitating an additional linear projection for unmatched dimensions in the following layer. This is not affordable under the parameter-efficient design. More experiments and discussions will be provided in the Appendix.

3.3. Efficient Prompting

Our approach to effective prompting aims to enhance the performance of the fine-tuned model. However, a natural question arises: Can we reduce the number of tunable prompts without sacrificing model performance? The

lottery ticket hypothesis (LTH) [16, 102] states that there exists a sub-network that can achieve the same test performance as the original over-parameterized network for a given task, without the need for unnecessary weights. Motivated by this hypothesis, we conducted an experiment in which we masked different visual prompts and found that various prompts have varying effects on the model performance, with some even having a negative impact. This observation is consistent with previous research [43, 57].

Based on our findings, we propose a prompt pruning method on visual prompts. The primary objective of this method is to retain the most influential prompts while eliminating redundant or unnecessary ones. By removing less important prompts, we can significantly improve the efficiency of prompt tuning while maintaining performance. To achieve this goal, we design a cascade pruning strategy that operates at two levels of granularity, namely token-wise pruning and segment-wise pruning, as illustrated in Fig. 2(d). Token-wise pruning initially identifies and removes the least important visual prompts. After this step, segment-wise pruning divides each remaining prompt into multiple segments and filters out negative segments. By jointly reducing the parameter usage in learnable visual prompts, our two-level pruning approach creates soft filtered prompts that can be re-trained in the rewinding stage.

Token-wise Pruning. We introduce a learnable mask variable $\rho = \{\rho_1, \rho_2, \ldots, \rho_M\}$ (M is the length of visual prompts) and associate it with the input visual prompts in each transformer layer. Here $\rho_k \in \{0, 1\}$, where 0 means the corresponding learnable input prompt is pruned. Then the masked version of the visual prompts becomes $\widetilde{P_k} = \rho_k \cdot P_k$. To determine the pruning position, we calculate the importance score [16, 57] of each prompt token and eliminate those positions with lowest scores. The importance score is defined as the expected sensitivity of the model to the mask variables ρ_k [60]:

$$S_{P_k} = \mathbb{E}_{x \sim \mathcal{D}_x} \left| \frac{\partial \mathcal{L}(x)}{\partial \rho_k} \right| \tag{6}$$

where $\mathcal{L}$ is the loss function, and $\mathcal{D}_x$ is the training data distribution [60]. The importance score assigned to each visual prompt reflects its contribution to the fine-tuning performance. A low importance score indicates that the prompt has a minor or even negative contribution to the fine-tuning process. Conversely, a high importance score suggests that the prompt is a meaningful and useful one that significantly contributes to the fine-tuning process.

Segment-wise Pruning. We further investigate the segment-wise pruning to preclude the negative prompt segments within each prompt. The embedding of each prompt token is first equally divided into R parts. Each part is treated as an isolated unit which can be optimized jointly. Similar to the token-wise pruning, we then assign a mask

variable to each segment inside the prompt token and filter out those segments with low importance scores.

Rewinding. After performing the two-level cascade pruning, the weight rewinding stage focuses on re-training the soft filtered prompt tokens. This process involves ranking the importance scores for each layer during the pruning stage and setting the corresponding mask variables to 0 when their importance scores are relatively low. Next, the soft filtered input prompts are re-trained along with other learnable parameters using the original combination of learning rate and weight decay during fine-tuning.

4. Experiment

4.1. Experimental Setup

Datasets. Our experiments are carried out on two image classification benchmarks. **VTAB-1k** [96] collects 19 benchmarked Visual Task Adaptation, categorized into three groups: (1) *Natural* contains natural images captured by standard cameras, (2) *Specialized* includes images taken by specialized equipment, and (3) *Structured* covers tasks requiring geometric comprehension (*i.e.*, counting, distance). Each task of VTAB-1k contains 1000 training examples. Following [34, 96], we apply the 800-200 split for training set on hyperparameter tuning. The final evaluation is run on the full training data. **FGVC** contains 5 benchmarked Fine-Grained Visual Classification, including CUB-200-2011 [81], NABirds [76], Oxford Flowers [63], Stanford Dogs [39] and Stanford Cars [19]. Following [34], the training set is randomly split into 90% `train` and 10% `val`. We use `val` for hyperparameter tuning.

Baselines. For fair comparison, we follow [34] and compare E^2VPT with other widely applied parameter-efficient fine-tuning methods. Results of two vision transformer architectures, Vision transformer [12] (ViT) and Swin transformer [54] (Swin), on image classification are discussed in §4.2. We also apply E^2VPT to two self-supervised objectives: MAE [24] and MoCo v3 [10].

Training. Following [34, 58], we conduct grid search to match the best tuning hyperparameters, learning rate (*i.e.*, [50, 25, 10, 5, 2.5, 1, 0.5, 0.25, 0.1, 0.05]), and weight decay (*i.e.*, [0.01, 0.001, 0.0001, 0.0]) on `val` set for each task. Notably, E^2VPT **does not require** specific-designed large learning rate in [34]. For all models, the learning rate is scheduled following a cosine decay policy and trained for 100 epochs (including 10 warm-up epochs). We follow the same batch size setting in [34]: 64/128 for ViT-Base/16 and 80 for Swin-Base, respectively. The number of segments for each token (§3.3) is set to 8. The percentages for the pruning stage are searched linearly between 10% and 90% with 10% intervals. The rewinding stage applies once to re-train the pruned input prompts.

Reproducibility. E^2VPT is implemented in Pytorch [67].

Table 1. **Image classification accuracy for ViT-Base/16 [12]** pretrained on supervised ImageNet-21k. Following [34], we report the average test accuracy (three runs) on FGVC [34] and VTAB-1k [96] benchmarks, and "Number of Wins" in [·] compared to full fine-tuning (Full) [32]. "Tuned/Total" is the average percentage of tuned parameters required by 24 tasks. "Scope" indicates the tuning scope of each method. "Additional parameters" is the existence of parameters in addition to the pretrained backbone and linear head. The highest accuracy among all approaches except FULL are shown in **bold**. E^{2}VPT outperforms the full fine-tuning in **19 of 24** instances with far fewer trainable parameters. More impressively, we further report "Number of Wins to VPT" in {·}. Our method beats VPT in **21 of 24** cases with considerably lower parameters. Per-task results are available in Appendix. Same for Table 2 and 3.

ViT-Base/16 [12] (85.8M)	Tuned/ Total	Scope Input	Scope Backbone	Extra params	FGVC [34] [5]	VTAB-1k [96] [19] Natural [7]	Specialized [4]	Structured [8]
Full [CVPR22][32]	100.00%		✓		88.54%	75.88%	83.36%	47.64%
Linear [CVPR22][32]	0.08%				79.32% [0]	68.93% [1]	77.16% [1]	26.84% [0]
Partial-1 [NeurIPS14][91]	8.34%				82.63% [0]	69.44% [2]	78.53% [0]	34.17% [0]
MLP-3 [CVPR20][9]	1.44%			✓	79.80% [0]	67.80% [2]	72.83% [0]	30.62% [0]
Sidetune [ECCV20][98]	10.08%		✓	✓	78.35% [0]	58.21% [0]	68.12% [0]	23.41% [0]
Bias [NeurIPS17][70]	0.80%		✓		88.41% [3]	73.30% [3]	78.25% [0]	44.09% [2]
Adapter [NeurIPS20][6]	1.02%		✓	✓	85.66% [2]	70.39% [4]	77.11% [0]	33.43% [0]
VPT [ECCV22][34]	0.73%	✓		✓	89.11% [4]	78.48% [6]	82.43% [2]	54.98% [8]
Ours	0.39%	✓	✓	✓	**89.22%** [4] {4}	**80.01%** [6] {5}	**84.43%** [3] {4}	**57.39%** [8] {7}

Table 2. **Image classification accuracy for Swin-Base [54]** pretrained on supervised ImageNet-21k.

Swin-Base [54] (86.7M)	Tuned/ Total	VTAB-1k [96] [19] Natural [7]	Specialized [4]	Structured [8]
Full [ICLR23][71]	100.00%	79.10%	86.21%	59.65%
Linear [ICLR23][71]	0.06%	73.52% [5]	80.77% [0]	33.52% [0]
Partial-1 [NeurIPS14][91]	14.58%	73.11% [4]	81.70% [0]	34.96% [0]
MLP-3 [CVPR20][9]	2.42%	73.56% [5]	75.21% [0]	35.69% [0]
Bias [NeurIPS17][70]	0.29%	74.19% [2]	80.14% [0]	42.42% [0]
VPT [ECCV22][34]	0.25%	76.78% [6]	83.33% [0]	51.85% [0]
Ours	0.21%	**83.31%** [6] {6}	**84.95%** [2] {3}	**57.35%** [3] {7}

Experiments are conducted on NVIDIA A100-40GB GPUs. To guarantee reproducibility, our full implementation will be publicly released.

4.2. Comparison with State-of-the-Arts

We respectively examine the performance and robustness of E^{2}VPT on ViT [12], Swin [54], and two self-supervised objectives — MAE [24] and MoCo v3 [10]. For reference, we provide the individual per-task results for Table 1, 2 and 3 in Appendix.

E^{2}VPT on ViT. We report the average accuracy score on VTAB-1k and FGVC benchmarks across four diverse task groups for three runs in Table 1, considering E^{2}VPT to the other eight tuning protocols under *pretrain-then-finetune* paradigm. Specifically, Full [32] updates both backbone and classification head; Linear [32], Parital-1 [91] (top layer) and MLP-3 [9] (3 MLP layers) are partial tuning methods that only update partial parameters. Sidetune [98], Bias [70] and Adapter [6] are extra module methods which add new trainable parameters to backbone for adaptation; VPT [34] is a most recent visual prompt tuning method. There are several key observations from these results. **First**, E^{2}VPT is able to outperform the full fine-tuning method in most cases, 21 out of 24 tasks. For example, our model achieves **0.68%** improvement on FGVC and **9.75%** improvements on VTAB-1k Structured respectively. This ob-

servation demonstrates the effectiveness of our approach for fast large-scale vision model adaptation. On the other hand, our model only trains **0.39%** parameters in the backbone, which is much more parameter efficient than the full fine-tuned model. **Second**, it is not surprising to see that the prompt tuning based approaches generally outperform the other parameter efficient methods, such as partial fine-tuning (Partial-1) and extra module (Adapter), indicating the superior adaptability of prompt tuning methods on large-scale vision models. Again, the number of tunable parameters in prompt tuning methods is also smaller compared to the other methods. **Third**, our approach consistently outperforms the strong VPT model with less tunable prompts, demonstrating the effective design of the key-value prompting and the efficient prompt pruning. The reason is that VPT only focus on design input visual prompts, which fail to capture the accurate interactions between image patches in the new data. In contrast, the key-value prompts in E^{2}VPT effectively bridge this gap.

E^{2}VPT on Hierarchical Transformer. To prove the effectiveness and generalization of our architectural design, we further extend E^{2}VPT to a hierarchical transformer — Swin [54], where the MSA layer is employed in local shifted windows and patch embeddings are merged at deeper layers. For generality, we follow the same settings in ViT [12] architecture to prepend K-V learnable pairs and [34] for altering input vectors (*i.e.*, these learnable vectors are attended within the local windows and ignored during patch merging). For pruning, we notice performance drop when incorporating within the deeper local windows. We therefore assign pruning stage only to the first stage. As Swin does not use `[CLS]` and apply the global pooling as input for classification head [34, 54], we follow this design when adapting our method. The exclusive experiments are deployed on the ImageNet-21k supervised pretrained Swin-Base [54]. E^{2}VPT consistently outperform all the other

Table 3. **Image Classification accuracy for different pretrained objectives** — MAE [24] and MoCo v3 [10] with ViT-Base [12] as backbone. Our method enjoys significant performance gains to VPT [34] while having lower parameter usage.

Pretrained objectives	MAE [24]				MoCo v3 [10]			
Parms & Data	Tuned/	VTAB-1k [96] [19]			Tuned/	VTAB-1k [96] [19]		
Methods	Total	*Natural* [7]	*Specialized* [4]	*Structured* [8]	Total	Natural [7]	Specialized [4]	Structured [8]
Full [CVPR22][32]	100.00%	59.31%	79.68%	53.82%	100.00%	71.95%	84.72%	51.98%
Linear [CVPR22][32]	0.04%	18.87% [0]	53.72% [0]	23.70% [0]	0.04%	67.46% [4]	81.08% [0]	30.33% [0]
Partial-1 [NeurIPS14][91]	8.30%	58.44% [5]	**78.28%** [1]	47.64% [1]	8.30%	72.31% [5]	84.58% [2]	47.89% [1]
Bias [NeurIPS17][70]	0.16%	54.55% [1]	75.68% [1]	**47.70%** [0]	0.16%	72.89% [3]	81.14% [0]	53.43% [4]
Adapter [NeurIPS20][6]	0.87%	54.90% [3]	75.19% [1]	38.98% [0]	1.12%	74.19% [4]	82.66% [1]	47.69% [2]
VPT [ECCV22][34]	0.10%	36.02% [0]	60.61% [1]	26.57% [0]	0.06%	70.27% [4]	83.04% [0]	42.38% [0]
Ours	0.07%	**59.52%** [4] {6}	77.80% [1] {2}	44.65% [3] {8}	0.13%	**76.47%** [4] {7}	**87.28%** [2] {4}	**54.91%** [6] {8}

Table 4. **Impact of different components** in E^2VPT on two instances: VTAB-1k *Natural* SVHN [62] and FGVC NABirds [77].

Fine-tuning Techniques			VTAB-1k *Natural* SVHN [62]			FGVC NABirds [77]		
Visual Prompts	Key-Value Prompts	Pruning & Rewinding	Pruning	Tuned / Total	Accuracy	Pruning	Tuned / Total	Accuracy
✓			0.0%	0.54%	78.1%	0.0%	1.02%	84.2%
✓	✓		0.0%	0.55%	83.8%	0.0%	1.05%	84.5%
✓		✓	56.3%	0.42%	79.0%	34.4%	0.63%	84.2%
✓	✓	✓	62.5%	0.43%	**85.3%**	40.0%	0.65%	**84.6%**

parameter-efficient methods on all three VTAB-1k problem classes and for the first time surpasses full fine-tuning on VTAB-1k *Specialized* and *Structured* using significantly fewer parameters (*i.e.*, **0.21%**).

Different Pretraining Methods. We conducted experiments with two self-supervised objectives, MAE [24] and MoCo v3 [10], on backbones pretrained without labeled data, following the approach of VPT [34]. While VPT yielded inconclusive results on these objectives, our proposed method, E^2VPT, outperformed other methods and achieved competitive performance to full fine-tuning (**8 of 19** instances under MAE, and **12 of 19** instances under MoCo v3), using significantly fewer model parameters (**0.07%** on MAE and **0.13%** on MoCo v3). Our method also outperformed VPT by a large margin (**59.52%** vs. 36.02% under MAE on VTAB-1k *Natural*). We leveraged the gap discussed in VPT, which indicates that self-supervised ViTs are fundamentally different from the supervised ones, and demonstrated the generality of our method to both pretraining objectives.

4.3. Diagnostic Experiments

Impact of Different Components. To investigate the impact of different components in E^2VPT, including visual prompts, key-value prompts, and pruning and rewinding, we conducted experiments on two tasks in the benchmarks. The results are summarized in Table 4. For SVHN [62], we found that the model with visual prompts alone achieved an accuracy of 78.1%. Adding key-value prompts and applying pruning and rewinding techniques individually led to additional gains (**5.7%** and **0.9%**), demonstrating the effectiveness of our key-value prompt tuning technique in the self-attention module as well as the pruning mechanism. Finally, combining all components together yielded the best performance, with an accuracy of 85.3%. We observed similar trends on FGVC NABirds [77].

Table 5. **Prompt location and Initialization** results on VTAB-1k [96] in three runs. Per-task results are available in Appendix.

ViT-Base/16 [12] (85.8M)		VTAB-1k [96] [19]		
		Natural [7]	*Specialized* [4]	*Structured* [8]
(a)	After	**80.67%** [6]	84.30% [3]	56.76% [8]
	Before	80.01% [6]	**84.43%** [3]	**57.39%** [8]
(b)	*Trunc. Norm.* [67]	79.77% [6]	84.30% [3]	56.36% [8]
	He [25]	**80.01%** [6]	**84.43%** [3]	**57.39%** [8]

Prompt Location. An fundamental distinction between E^2VPT and other methods is the learnable key-value prompts introduced to self-attention. In our implementation, we prepend the key-value prompts to the sequence of Key and Value matrices. Further investigation is required to determine the appropriate placement of the learnable prompts. We provide ablation results on VTAB-1k exhaustively in Table 5(a). We show that both prepending learnable prompts before or after Key and Value matrices show competitive results, validating the robustness of our approach on prompt locations. We choose "Before" as our baseline method in all our experiments since it achieves slightly better results on average (*i.e.*, 73.94% *vs* 73.91%).

Initialization. Table 5(b) reports the performance of our approach with respect to two widely adopted initialization methods: *truncated normal* [61, 67] and *He initialization* [25] on VTAB-1k benchmark. The results show that *He initialization* generally provides more stable and preferable performances on average, though we observe that in some specific tasks (*i.e.*, *truncated normal* is 1.1% higher in accuracy over *He* on VTAB-1k *Specialized* Diabetic Retinopathy Detection [13]) *truncated normal* gets slightly better results. In conclusion, E^2VPT shows robustness on different initialization methods and is able to achieve consistent performance with full fine-tuning.

Prompt Length. Prompt length is the only hyper-parameter needed to tune in E^2VPT. To further analyze the impact of different prompt lengths on the model performance, we
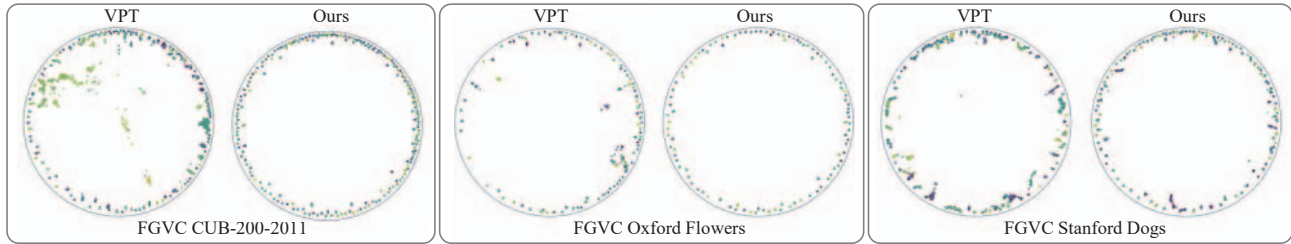
Figure 3. **Hyperbolic visualization results** from VPT [34] and ours on 3 FGVC tasks (*i.e.*, FGVC CUB-200-2011 [81], Oxford Flowers [63] and Stanford Dogs [39]). Our method shows consistently better clustering pushed to the border of the Poincaré disk.
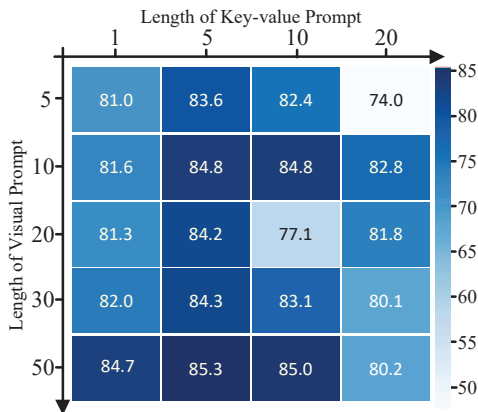

Figure 4. **Sensitivity of input prompt and key-value prompt lengths.** We vary the number of prompts for different combinations, and show their results on VTAB-1k *Natural* SVHN [62].

conducted a comprehensive study on the lengths of visual prompts and key-value prompts for a better understanding of their characteristics on VTAB-1k *Natural* SVHN [62]. The length of visual prompts is typically limited to [5, 10, 20, 30, 50], while the length of key-value prompts is restricted to [1, 5, 10, 50], which is a standard configuration for most datasets. The model performance results on different prompt length combinations are reported in Fig. 4. It can be seen that, when using 50 visual prompts, a relative shorter key-value prompt can benefit performance notably (*i.e.*, 84.7% when introducing one key-value prompt *vs* 78.1% without key-value prompts), while further increasing the length of the key-value prompt yields a small performance gain (*i.e.*, 85.3% when using 5 key-value prompts). We also notice that using a large number of key-value prompts lead to subpar results (*i.e.*, 80.2% with 20 key-value prompts). Similar patterns are observed with other visual prompt lengths. We argue that a heavy parameter engineering in self-attention layer might distort the original attention map and does harm to adaptation.

4.4. Visualization

Following [2, 14, 17, 40, 68], we show hyperbolic visualizations results on training set for VPT and ours on three tasks in FGVC (*i.e.*, CUB-200-2011 [81], Oxford Flowers [63], and Stanford Dogs [39]). Hyperbolic space, to be specific, is a Riemannian manifold of constant negative curvature. While there are several isometric models of hyperbolic space, we follow previous work [14, 17] and stick to the Poincaré ball model. Similar to [14], we use UMAP [59] with the "hyperboloid" distance metric to reduce the dimensionality to 2D. ViT-Base plays as an encoder with two types of pretraining (*i.e.*, tuned models under VPT, and ours after rewinding, respectively). We freeze the models during fine-tuning and output embeddings are mapped to hyperbolic space. Adam optimizer [55] with a learning rate of 3×10^{-5} is applied to all settings. The weight decay is 0.01 with batch size equals to 900. All models are trained for 50 steps for fair comparison, with a gradient clip by norm 3.

Fig. 3 illustrates how learned embeddings are arranged on the Poincaré disk. We can see that in E^2VPT, samples are clustered according to labels, and each cluster is pushed closer to the border of the disk, indicating that the encoder separates class well. On the other hand, we observe in VPT that some of the samples move towards the center and intermix [14], indicating possible confusion during projection. We also follow [14, 68, 40] and present the Recall@K metric in Appendix for reference. These visualization results further validate the effectiveness of the proposed E^2VPT approach in generating separatable embeddings from the input images in the new tasks.

5. Conclusion and Discussion

The vast majority of current efforts under the *pretrain-then-finetune* paradigm seek to reduce parameter usage while overlooking the inner design of transformer-based architecture. In light of this view, we present E^2VPT, a new parameter-efficient visual prompt tuning approach to model the transformer architecture during adaptation. It enjoys several advantages: **i)** consider self-attention mechanism during tuning for superior performance to current parameter-efficient fine-tuning; and **ii)** apply pruning and rewinding stages to reduce parameter usage in input visual prompts. The systemic merits enable an effective yet efficient algorithm. As a whole, we conclude that the outcomes elucidated in this paper impart essential understandings and necessitate further exploration within this realm.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, 2021. 3

[2] Mina Ghadimi Atigh, Julian Schoep, Erman Acar, Nanne Van Noord, and Pascal Mettes. Hyperbolic image segmentation. In *CVPR*, 2022. 8

[3] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022. 2

[4] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. *arXiv preprint arXiv:2012.09958*, 2020. 2

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020. 1, 2

[6] Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. Tinytl: Reduce memory, not parameters for efficient on-device learning. In *NeurIPS*, 2020. 1, 2, 3, 6, 7

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[8] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *ICCV*, 2021. 3

[9] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. In *CVPR*, 2020. 3, 6

[10] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021. 2, 3, 5, 6, 7

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2018. 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2, 3, 5, 6, 7

[13] Will Cukierski Emma Dugas, Jared Jorge. Diabetic retinopathy detection, 2015. 7

[14] Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic vision transformers: Combining improvements in metric learning. In *CVPR*, 2022. 8

[15] Quentin Fournier, Gaétan Marceau Caron, and Daniel Aloise. A practical survey on faster and lighter transformers. *arXiv preprint arXiv:2103.14636*, 2021. 2

[16] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019. 2, 5

[17] Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *NeurIPS*, 2018. 8

[18] Yunhe Gao, Xingjian Shi, Yi Zhu, Hao Wang, Zhiqiang Tang, Xiong Zhou, Mu Li, and Dimitris N Metaxas. Visual prompt tuning for test-time domain adaptation. *arXiv preprint arXiv:2210.04831*, 2022. 3

[19] Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. Fine-grained car detection for visual census estimation. In *AAAI*, 2017. 5

[20] Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. In *ICML*, 2021. 1

[21] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE TPAMI*, 2022. 2

[22] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *NeurIPS*, 2015. 2

[23] Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *NeurIPS*, 1992. 2

[24] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 2, 3, 5, 6, 7

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 7

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[27] Xuehai He, Chunyuan Li, Pengchuan Zhang, Jianwei Yang, and Xin Eric Wang. Parameter-efficient fine-tuning for vision transformers. *arXiv preprint arXiv:2203.16329*, 2022. 2, 3

[28] Yun He, Steven Zheng, Yi Tay, Jai Gupta, Yu Du, Vamsi Aribandi, Zhe Zhao, YaGuang Li, Zhao Chen, Donald Metzler, et al. Hyperprompt: Prompt-based task-conditioning of transformers. In *ICML*, 2022. 3

[29] Lin Huang, Jianchao Tan, Ji Liu, and Junsong Yuan. Hand-transformer: non-autoregressive structured modeling for 3d hand pose estimation. In *ECCV*, 2020. 2

[30] Lin Huang, Jianchao Tan, Jingjing Meng, Ji Liu, and Junsong Yuan. Hot-net: Non-autoregressive transformer for 3d hand-object pose estimation. In *ACMMM*, 2020. 2

[31] Mike Innes, Alan Edelman, Keno Fischer, Chris Rackauckas, Elliot Saba, Viral B Shah, and Will Tebbutt. A differentiable programming system to bridge machine learning and scientific computing. *arXiv preprint arXiv:1907.07587*, 2019. 1

[32] Eugenia Iofinova, Alexandra Peste, Mark Kurtz, and Dan Alistarh. How well do sparse imagenet models transfer? In *CVPR*, 2022. 3, 6, 7

[33] Khawar Islam. Recent advances in vision transformer: A survey and outlook of recent work. *arXiv preprint arXiv:2203.01536*, 2022. 2

[34] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022. 1, 2, 3, 5, 6, 7, 8

[35] Menglin Jia, Zuxuan Wu, Austin Reiter, Claire Cardie, Serge Belongie, and Ser-Nam Lim. Exploring visual engagement signals for representation learning. In *ICCV*, 2021. 2, 3

[36] Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *ECCV*, 2022. 2

[37] Lukasz Kaiser, Aidan N Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. In *ICML*, 2017. 1

[38] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM Computing Surveys*, 54(10s):1–41, 2022. 2

[39] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *CVPR Workshop*, 2011. 5, 8

[40] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *CVPR*, 2020. 8

[41] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In *NeurIPS*, 1989. 2

[42] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, 2021. 3

[43] Changlin Li, Bohan Zhuang, Guangrun Wang, Xiaodan Liang, Xiaojun Chang, and Yi Yang. Automated progressive learning for efficient training of vision transformers. In *CVPR*, 2022. 2, 5

[44] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016. 2

[45] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *CVPR*, 2022. 2

[46] James Liang, Tianfei Zhou, Dongfang Liu, and Wenguan Wang. Clustseg: Clustering for universal segmentation. *arXiv preprint arXiv:2305.02187*, 2023. 2

[47] Jinfeng Lin, Yalin Liu, Qingkai Zeng, Meng Jiang, and Jane Cleland-Huang. Traceability transformed: Generating more accurate links with pre-trained bert models. In *ICSE*, 2021. 1

[48] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 2

[49] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. A survey of transformers. *AI Open*, 2022. 2

[50] Dongfang Liu, Yiming Cui, Yingjie Chen, Jiyong Zhang, and Bin Fan. Video object detection for autonomous driving: Motion-aid feature calibration. *Neurocomputing*, 409:1–11, 2020. 2

[51] Dongfang Liu, Yiming Cui, Wenbo Tan, and Yingjie Chen. Sg-net: Spatial granularity network for one-stage video instance segmentation. In *CVPR*, 2021. 2

[52] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. In *ICLR*, 2020. 2

[53] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *CVPR*, 2022. 2

[54] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1, 2, 3, 5, 6

[55] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICML*, 2017. 8

[56] Yawen Lu, Qifan Wang, Siqi Ma, Tong Geng, Yingjie Victor Chen, Huaijin Chen, and Dongfang Liu. Transflow: Transformer as flow learner. In *CVPR*, 2023. 2

[57] Fang Ma, Chen Zhang, Lei Ren, Jingang Wang, Qifan Wang, Wei Wu, Xiaojun Quan, and Dawei Song. Xprompt: Exploring the extreme of prompt tuning. In *EMNLP*, 2022. 3, 5

[58] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018. 2, 3, 5

[59] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. 8

[60] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *NeurIPS*, 2019. 5

[61] Meenal V Narkhede, Prashant P Bartakke, and Mukul S Sutaone. A review on weight initialization strategies for neural networks. *Artificial Intelligence Review*, 2022. 7

[62] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 7, 8

[63] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing*, 2008. 5, 8

[64] Rohit Nishant, Mike Kennedy, and Jacqueline Corbett. Artificial intelligence for sustainability: Challenges, opportunities, and a research agenda. *International Journal of Information Management*, 2020. 1

[65] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016. 3

[66] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. In *CVPR*, 2021. 2

[67] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An

imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5, 7

[68] Wei Peng, Tuomas Varanka, Abdelrahman Mostafa, Henglin Shi, and Guoying Zhao. Hyperbolic deep neural networks: A survey. *IEEE TPAMI*, 2021. 8

[69] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 2020. 2

[70] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *NeurIPS*, 2017. 2, 3, 6, 7

[71] Yi Ren, Shangmin Guo, Wonho Bae, and Danica J Sutherland. How to prepare your task head for finetuning. In *ICLR*, 2023. 6

[72] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *NeurIPS*, 2021. 2

[73] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 1

[74] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *ICCV*, 2021. 2

[75] Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 2016. 1, 3

[76] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, 2015. 5

[77] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, 2015. 7

[78] Aimee Van Wynsberghe. Sustainable ai: Ai for sustainability and the sustainability of ai. *AI and Ethics*, 2021. 1

[79] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[80] Ricardo Vinuesa, Hossein Azizpour, Iolanda Leite, Madeline Balaam, Virginia Dignum, Sami Domisch, Anna Felländer, Simone Daniela Langhans, Max Tegmark, and Francesco Fuso Nerini. The role of artificial intelligence in achieving the sustainable development goals. *Nature Communications*, 2020. 1

[81] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5, 8

[82] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021. 2

[83] Wenguan Wang, Cheng Han, Tianfei Zhou, and Dongfang Liu. Visual recognition with deep nearest centroids. In *ICLR*, 2022. 2

[84] Wenguan Wang, James Liang, and Dongfang Liu. Learning equivariant segmentation with instance-unique querying. In *NeurIPS*, 2022. 2

[85] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021. 3

[86] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *CVPR*, 2021. 2

[87] Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, et al. Sustainable ai: Environmental implications, challenges and opportunities. *Proceedings of Machine Learning and Systems*, 2022. 1

[88] Yinghui Xing, Qirui Wu, De Cheng, Shizhou Zhang, Guoqiang Liang, and Yanning Zhang. Class-aware visual prompt tuning for vision-language pre-trained model. *arXiv preprint arXiv:2208.08340*, 2022. 3

[89] Li Yang, Qifan Wang, Jingang Wang, Xiaojun Quan, Fuli Feng, Yu Chen, Madian Khabsa, Sinong Wang, Zenglin Xu, and Dongfang Liu. Mixpave: Mix-prompt tuning for few-shot product attribute value extraction. In *ACL*, 2023. 3

[90] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Keypoint localization via transformer. In *ICCV*, 2021. 2

[91] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NeurIPS*, 2014. 1, 3, 6, 7

[92] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *ICLR*, 2020. 1

[93] Zhenxun Yuan, Xiao Song, Lei Bai, Zhe Wang, and Wanli Ouyang. Temporal-channel transformer for 3d lidar-based video object detection for autonomous driving. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 2

[94] Yuhang Zang, Wei Li, Kaiyang Zhou, Chen Huang, and Chen Change Loy. Unified vision and language prompt learning. *arXiv preprint arXiv:2210.07225*, 2022. 2

[95] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *CVPR*, 2022. 2

[96] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation

learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 1, 5, 6, 7

[97] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. Patchformer: An efficient point transformer with patch attention. In *CVPR*, 2022. 2

[98] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In *ECCV*, 2020. 2, 3, 6

[99] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016. 3

[100] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021. 2

[101] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2

[102] Bohan Zhuang, Jing Liu, Zizheng Pan, Haoyu He, Yuetian Weng, and Chunhua Shen. A survey on efficient training of transformers. *arXiv preprint arXiv:2302.01107*, 2023. 2, 5