Diversity-Aware Meta Visual Prompting Supplementary Materials

Qidong Huang[1] Xiaoyi Dong[1] Dongdong Chen[2] Weiming Zhang[1,*]
Feifei Wang[1] Gang Hua[3] Nenghai Yu[1]
[1]University of Science and Technology of China [2]Microsoft Cloud AI [3]Wormpex AI Research
{hqd0037@mail., dlight@mail., zhangwm@, wangfeifei@mail., ynh@}ustc.edu.cn
{cddlyf@, ganghua@}gmail.com

We provide supplementary materials for our manuscript "Diversity-Aware Meta Visual Prompting", including:

- Discussion on limitations and boarder impacts.
- Discussion on tunable parameters.
- Setting for models w/o task-specific heads.
- Setting for convolution networks.
- More details regarding datasets.
- More details regarding model backbones.
- More results on MoCo-v3 and ResNet-50.
- More results on VTAB-1k benchmark.
- More details regarding hyper-parameters.
- Ablation study on clustering threshold.

Each part is specified as follows respectively. Our code is available at: https://github.com/shikiw/DAM-VP.

A. Limitations and Social Impacts

Here we discuss the shortcomings and the potential social influences of the proposed diversity-aware meta visual prompting (DAM-VP), respectively.

For limitations, two aspects of concerns might be raised. First, it is obvious that DAM-VP introduces more visual prompts than VP [1] which trains the universal task-specific prompt. At the first glance, learning multiple visual prompts on a particular downstream task seems less parameter-efficient during adaption. However, we should argue that the amount of prompts introduced by our method is quite reasonable, *e.g.*, $\sim$25 for ViT-B-22K averaged on 10 datasets. This amount of extra tunable parameters brought by DAM-VP is less than that is brought by an additional linear head. The tunable parameters brought by DAM-VP is comparable with baselines methods, which is detailed in Sec.B and showcased in Table 1. Relative to tuning all of pre-trained model parameters, the amount of extra tunable parameters brought by our method is really insignificant, which has very limited threat to the storage. On the other hand, when the number of tunable parameters introduced is small enough, it makes no sense to compare the efficiency of different methods only by comparing the number of tunable parameters. **We should claim that the efficiency of our method is mainly reflected in our ability to converge faster, *e.g.*, using 10 epochs to be comparable with (or even surpass) the performances of baselines that trains for 100 epochs.**

	FT	LP	Adapter	VP	VPT	Ours
Total params	10.01×	0.43×	0.51×	0.44×	0.49×	0.63×

Table 1. Total tunable parameters needed for 10 datasets when adapting ViT-B-22K in the head-tuning scenario, where "×" the multiple of the amount of tunable parameters relative to the total amount of pre-trained ViT-B-22K encoder parameters ($\sim$85.8M). Here "FT" means fully-tuning and "LP" means linear probing.

For social impacts, it is clear that exploring more effective and efficient visual prompting methods can greatly benefit the adaption of nowadays huge pre-trained models on downstream tasks. Visual prompting provides a novel perspective for boosting transfer learning performance of pre-trained vision models. It is crucial, at least on the aspect of application, for pre-trained models that has large capacity and capability to be easily re-programmed in both industry and academia.

B. Discussion on Tunable parameters

Although keeping the pre-trained models untouched, our visual prompts are also the extra introduced parameters for

*Corresponding author.

Dataset	Usage	Meta Class	# Categories	Train	Val	Test	Diversities	Prompts
DTD [4]	Evaluation	textures	47	1,880	1,880	1,880	78.7	154
CUB200 [24]		birds	200	5,394	600	5,794	76.0	18
NABirds [10]		birds	555	21,536	2,393	24,633	74.8	22
Stanford-Dogs [14]		dogs	120	10,800	1,200	8,580	73.4	33
Oxford-Flowers [19]		flowers	102	1,020	1,020	6,149	72.7	26
Food101 [2]		food dishes	101	60,600	15,150	25,250	72.7	51
CIFAR100 [15]		all	100	40,000	10,000	10,000	70.9	79
CIFAR10 [15]		all	10	40,000	10,000	10,000	70.2	42
GTSRB [23]		traffic signs	43	21,312	2,526	12,630	67.5	6
SVHN [17]		numbers	10	58,605	14,652	26,032	61.8	3
SUN397 [25]	Meta Training	scenes	397	108,754	-	-	76.9	128
STL10 [5]		all	10	5,000	-	8,000	74.1	43
Fru92 [11]		fruits	92	9,200	4,600	55,814	74.1	42
Oxford-IIIT Pet [20]		cats,dogs	37	3,680	-	3,669	72.4	18
Veg200 [11]		vegetables	200	20,000	10,000	61,117	71.5	95
EuroSAT [9]		remote	10	27,000	-	-	64.6	12

Table 2. Basic information of the datasets used in our work. "Prompts" shows the prompt numbers used on ViT-B-1K in the head-freezing/missing scenario.

transfer learning. We compare the amount of tunable parameters of different methods on ViT-B-22K in the head-tuning scenario, showcased in Table 1. Apparently, our method DAM-VP uses the similar amount of tunable parameters with previous visual prompting methods, indicating the comparable parameter efficiency. Compared with VPT [13], the slightly more tunable parameters introduced by DAM-VP is relatively tolerable and acceptable since they are both far away less than FT. However, it can not reflect the efficiency during adaption. As we stated in limitations, our method is more efficient than other methods thanks to its faster converging, using 10 epochs to be comparable with (or even surpass) the previous methods that use 100 epochs.

C. Setting for Models w/o Task-Specific Heads

In the head-freezing/missing scenario, the task-specific is discarded so that it is necessary to design an approach to map the output feature to our desired classification logits. Previous VP [1] applies a hard-coded mapping method to tackle with this, *i.e.*, directly using the first N_c channels of feature output as the classification probability output of N_c categories. However, we argue that this method is too straightforward that it ignores the important property of neural networks, *i.e.*, usually, some neurons in the intermediate layer might be not sufficiently active and relatively robust to the different inputs. This denotes that some of the selected feature channels selected by hard-coded mapping probably have very limited space for their variation, since their corresponding neurons are more "robust". In other words, the optimization of visual prompts might be seriously hindered by these less active channels.

To alleviate this issue, we propose active-based mapping, a simple but effective method for converting features to logits. Specifically, given a pre-trained vision encoder $\mathcal{M}$, we input it with a batch of randomly generated Gaussian noises to observe each channel's variance of the output visual feature. By sorting these variances, we can obtain the ranking of the sensitivities of output feature channels and select the largest N_c channels as our desired active channels. After normalized, these N_c channels can construct the output probabilities of any input image.

D. Setting for Convolution Networks

Different from previous methods such as VPT [13] and Adapter [12, 21], our method is universal for both Vision Transformer and convolution networks since our prompt design is consistent with VP [1] that applies pixel-level visual prompts. The prompt is actually the learnable pixel patches, which looks like a photo frame with the width of 30 and can be added on the original image as input. We choose this design mainly because: 1) it naturally suits all kinds of vision models since directly crafting pixels guarantees that only the input space is considered to be modified. 2) The photo-frame-like structure can greatly inherent the main content of the input image, which usually allocates at the center of the image. In this supplementary, we also provide the prompting results on ResNet-50 [8] that is pre-trained on ImageNet-1k in Table 4.

E. Dataset Specification

We adopt total 16 datasets in experiments, in which 10 for evaluation and 6 for meta training. The basic information regarding these datasets is given in Table 2 and image examples of evaluation datasets are showcased in Figure 1.

F. Backbone Specification

There are total 6 backbones are used in our experiments,

Figure 1. Image examples for each dataset in our evaluation, where the data diversity score decreases from top to bottom.

	Extra Head	DTD [4]	CUB200 [24]	NABirds [10]	Dogs [14]	Flowers [19]	Food101 [2]	CIFAR100 [15]	CIFAR10 [15]	GTSRB [23]	SVHN [17]	Average
Data diversity	-	78.7	76.0	74.8	73.4	72.7	72.7	70.9	70.2	67.5	61.8	-
Fully-Tuning	✓	71.3	78.8	72.8	89.5	95.1	83.3	84.0	97.1	96.8	90.6	85.9
Linear	✓	68.5	78.3	70.3	89.4	87.1	79.4	80.6	94.3	79.5	43.5	77.1
Adapter [12, 21]	✓	69.2	81.5	73.9	83.2	90.8	65.6	73.3	95.0	90.7	73.5	79.7
VP [1]	✓	65.9	75.4	69.0	91.0	84.5	77.7	79.1	95.1	89.8	91.3	81.9
VPT [13]	✓	67.2	72.1	65.3	80.5	88.5	65.2	72.8	94.4	88.5	61.8	75.6
DAM-VP (10 epochs)	✓	68.6	77.0	70.5	93.2	86.9	79.6	79.6	95.1	90.1	85.4	82.6
DAM-VP (50 epochs)	✓	71.2	79.7	71.4	93.9	89.6	80.1	81.8	95.3	92.8	89.3	84.5

Table 3. Head-tuning adaption performance of different methods on MoCo-v3-B-1K, where we report image classification accuracy and all of baseline methods are trained for **100 epochs**.

	Extra Head	DTD [4]	CUB200 [24]	NABirds [10]	Dogs [14]	Flowers [19]	Food101 [2]	CIFAR100 [15]	CIFAR10 [15]	GTSRB [23]	SVHN [17]	Average
Data diversity	-	78.7	76.0	74.8	73.4	72.7	72.7	70.9	70.2	67.5	61.8	-
Fully-Tuning	✓	62.1	76.5	73.7	75.8	88.1	84.0	81.2	95.8	95.2	96.5	83.6
Linear	✓	64.8	68.1	58.7	88.5	81.0	71.8	71.4	89.9	79.4	45.3	71.9
VP [1]	✓	63.4	64.3	56.4	80.7	78.7	64.2	62.2	82.1	84.8	78.1	71.5
VPT [13]	✓	63.5	69.8	58.4	87.3	81.2	70.0	70.2	88.6	82.9	60.4	73.2
DAM-VP (10 epochs)	✓	68.4	65.3	57.4	88.0	76.1	69.4	71.6	89.4	83.7	75.6	74.5
DAM-VP (50 epochs)	✓	68.5	67.8	58.4	88.5	83.7	71.4	72.5	90.2	85.6	78.0	76.5

Table 4. Head-tuning adaption performance of different methods on ResNet50-1K, where we report image classification accuracy and all of baseline methods are trained for **100 epochs**.

Name	Backbone	Pre-trained Paradigm	Pre-trained Dataset	Params (M)	Feature Dim
ViT-B-1K	ViT-B/16	Supervised	ImageNet-1k	85	768
ViT-B-22K	ViT-B/16	Supervised	ImageNet-22k	85	768
CLIP-ViT-B	ViT-B/16	CLIP	400M web data	85	768
Swin-B-22K	Swin-B	Supervised	ImageNet-22k	88	1024
MoCo-B-1K	ViT-B/16	Contrastive	ImageNet-1k	85	768
ResNet50-1K	ResNet-50	Supervised	ImageNet-1k	23	2048

Table 5. Basic information of the pre-trained vision backbones used in our experiment.

Backbone	ViT-B/16			ViT-L/16		
VTAB-1k	Natural	Specialized	Structured	Natural	Specialized	Structured
Fully-Tuning	75.88	83.36	47.64	75.99	84.68	50.71
Linear	68.93	77.16	26.84	71.17	73.50	26.44
VPT	78.48	82.43	**54.98**	82.80	84.63	55.85
Ours	**81.29**	**83.78**	54.33	**83.53**	**85.24**	**56.35**

Table 6. Results on VTAB benchmark (19 datasets) for ViT-B-22K and ViT-L-22K.

shown in Table 5. We report the results of ViT-B-1K [7], ViT-B-22K [7], CLIP-ViT-B [22] and Swin-B-22K [16] in our manuscript and report the results of MoCo-B-1K [3] and ResNet50-1K [8] in this supplementary.

G. More Prompting Results

Threshold	33	32	31	30	29
Flowers Acc (%)	64.3	75.7	84.1	88.0	91.1
Prompt params (M)	0.35	0.98	1.82	3.50	4.90

Table 7. **Configure clustering threshold for scaling the prompting performance. Introducing more prompts for DAM-VP benefits the accuracy when the storage is not constrained.** We test ViT-B-1K on Oxford-Flowers in the head-freezing/missing scenario. We trade-off between the accuracy and extra parameters, finally selecting 31 as the default threshold.

For the self-supervised pre-trained model, we verify our DAM-VP on ViT-B/16 [7] pre-trained by MoCo v3 [3] and show the results in Table 3. We can find that VPT performs not good to adapt MoCo-v3 pre-trained model, whereas our DAM-VP is able to achieve comparable downstream accuracy with Full-tuning.

For the pre-trained convolution network, we verify our DAM-VP on ImageNet-1k [6] supervised pre-trained ResNet-50 [8] and show the results in Table 4. Note that Adapter is hard to be extended to convolution networks. For VPT, we follow the extending approach of its paper. Though obtaining lower accuracy than Full-tuning, our method still outperforms previous visual prompting methods and linear probing.

lr / wd	DTD [4]	CUB200 [24]	NABirds [10]	Dogs [14]	Flowers [19]	Food101 [2]	CIFAR100 [15]	CIFAR10 [15]	GTSRB [23]	SVHN [17]	
Fully-Tuning	1e-3/1e-4	5e-4/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	ViT-B-1K
Linear	1e-1/0	5e-1/0	1e-3/0	2.5e+2/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	
Adapter [12, 21]	5e-3/1e-4	1e-2/1e-1	5e-2/1e-2	5e-3/1e-2	1e-2/1e-2	5e-3/1e-4	5e-3/1e-4	1/1e-4	5e-1/1e-4	5e-1/1e-4	
VP [1]	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	
VPT [13]	5/1e-4	5e-2/1e-3	5/1e-4	5e+1/0	5/1e-4	0.25/1e-4	1e-2/1e-4	2.5/1e-2	5e-1/1e-4	2/1e-4	
DAM-VP	8e+3/0	5e+4/0	1e+4/0	1e+4/0	8e+3/0	5e+3/0	5e+3/0	5e+3/0	5e+3/0	5e+3/0	
Fully-Tuning	1e-3/1e-4	5e-3/1e-4	5e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	CLIP-ViT-B
Linear	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1e-1/0	
TP [22]	-	-	-	-	-	-	-	-	-	-	
VP [1]	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	1e+4/0	
DAM-VP	5e+4/0	2e+4/0	2e+4/0	1.5e+4/0	1e+4/0	5e+3/0	8e+3/1e-4	5e+3/0	7e+3/0	5e+4/0	

Table 8. Learning rate and weight decay specification for our experiments in **head-freezing/missing** adaption.

lr / wd	DTD [4]	CUB200 [24]	NABirds [10]	Dogs [14]	Flowers [19]	Food101 [2]	CIFAR100 [15]	CIFAR10 [15]	GTSRB [23]	SVHN [17]	
Fully-Tuning	5e-4/1e-4	5e-3/0	5e-3/0	5e-3/0	1e-3/1e-2	5e-4/1e-4	1e-3/1e-4	1e-3/1e-4	5e-4/1e-4	1e-3/1e-3	ViT-B-22K
Linear	1/0	5/1e-4	10/0	1e-1/1e-4	1e+1/1e-4	1e-3/0	1e-1/0	1e-2/0	1e-2/0	0.25/1e-2	
Adapter [12, 21]	5e-3/1e-4	1e-3/1e-2	5e-3/1e-3	1e-3/1e-4	5e-3/1e-4	5e-3/1e-4	5e-3/1e-2	5e-4/1e-4	5e-3/1e-4	5e-3/1e-4	
VP [1]	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	
VPT [13]	5/1e-4	1e+1/1e-3	5/1e-4	5e+1/1e-4	25/1e-3	5/1e-3	5/1e-3	2.5/1e-2	1e+1/1e-4	2.5/0	
DAM-VP	5/1e-1	1/1e-1	5/1e-2	1/1e-1	1e+1/5e-2	1/1e-2	5e-1/2e-3	1e-1/5e-3	5e+2/0	3e+2/0	
Fully-Tuning	1e-4/1e-4	1e-4/1e-4	1e-4/1e-4	1e-4/1e-4	1e-4/1e-4	1e-4/1e-4	5e-4/1e-4	1e-4/1e-4	1e-4/1e-4	1e-3/1e-2	Swin-B-22K
Linear	2.5/1e-2	5e-1/0	5e-1/0	5e-1/0	5e-1/0	5e-1/0	1e-1/1e-2	5e-1/0	5e-1/0	1e-1/1e-3	
Adapter [12, 21]	5e-1/1e-4	5e-2/1e-1	5e-2/1e-2	5e-3/1e-2	5e-2/1e-2	5e-1/1e-4	5e-3/1e-4	1/1e-4	5e-1/1e-4	5e-2/1e-4	
VP [1]	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	
VPT [13]	0.25/1e-2	5e-2/1e-3	5e-2/1e-3	5e+1/0	5e-2/1e-2	5e-3/1e-4	5/1e-3	2.5/1e-2	5/1e-4	0.25/1e-2	
DAM-VP	1e-1/5e-2	1e-1/1e-1	1/1e-2	1e-1/1e-1	1/1e-4	1e-1/5e-2	5e-2/1e-2	5e-2/1e-2	5e+2/0	1e+1/0	
Fully-Tuning	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-2	MoCo-v3-B-1K
Linear	1/0	1e-1/0	1e-1/0	1e-1/0	2.5/1e-4	1e-1/0	1e-1/0	1e-1/0	1e-1/0	1/0	
Adapter [12, 21]	5e-3/1e-2	5e-2/1e-1	5e-2/1e-2	5e-3/1e-2	5e-3/1e-4	5e-1/1e-4	5e-3/1e-4	1e-2/1e-4	5e-1/1e-4	5e-3/1e-4	
VP [1]	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	4e+1/0	
VPT [13]	5e+2/0	5e-2/1e-3	5e-1/1e-3	5/1e-4	1e+2/1e-4	1e-2/1e-4	1e+2/1e-4	1e-1/1e-3	2/1e-4	5e+1/1e-4	
DAM-VP	5e-1/1e-2	1/5e-1	5/5e-2	1/5e-1	1/1e-1	5e-1/1e-1	1e-1/5e-2	1e-1/5e-2	2.5e+2/0	1e+1/0	
Fully-Tuning	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	1e-3/1e-4	ResNet50-1K
Linear	1e-1/1e-2	1e-1/0	1e-1/0	1e-1/0	5e-2/1e-2	1e-1/0	1e-1/0	1e-1/0	1e-1/0	5/0	
VP [1]	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	
VPT [13]	1/1e-2	1e-1/1e-1	1/1e-2	1/5e-2	5e-1/1e-2	1e-2/1e-4	1e-1/1e-3	1e-1/1e-3	1e-1/1e-4	5e-1/0	
DAM-VP	5e-1/5e-1	1e-1/1e-1	1/1e-2	1/5e-2	1/5e-1	5e-1/5e-1	5e-1/1e-2	2e-1/1e-2	2.5e+1/0	5/0	

Table 9. Learning rate and weight decay specification for our experiments in **head-tuning** adaption.

H. More Results on VTAB-1k

VTAB-1k [26] benchmarks transfer learning methods with total 19 different task datasets, which contains three splits named "Natural", "Specialized" and "Structured", respectively. We report the comparison results in Table 6.

I. Hyper-Parameter Specification

Here we mainly specify the detailed configuration of hyper-parameters in our experiments. By default, we use AdamW optimizer for fully-tuning, Adapter and SGD optimizer for linear probing, VP, VPT and our DAM-VP during adaption. Following VPT [13], we adopt cosine decay scheduler and unify the warm up epochs as 10. The configuration about learning rate and weight decay are listed in Table 8 and 9 for head-freezing/missing and head-tuning scenarios, respectively. During meta training, we use Reptile [18] as the basic solution and adopt Adam optimizer, with the unified meta learning rate (meta step size) as 0.5, the learning rate for fast update as 0.5, the unified fast update step as 4. The weight decay rate is set as 0 for the head-freezing/missing case and 1e-4 for the head-tuning case.

J. Ablation Study on Clustering Threshold

We further analyse the impact of different threshold of agglomerative clustering used in our diversity-adaptive data

partition. By default, we set the threshold as 31 for ViT-B-1K, 10 for ViT-B-22K, 20 for Swin-B-22K, 18 for MoCo-v3-B-1K and 21 for ResNet50-1K. Usually, the lower threshold represents the more clusters obtained by clustering. In Table 7, we surprisingly found that in the head-freezing/missing case, the prompting performance can be greatly boosted with the decreasing of threshold, whereas the introduced extra tunable parameters are also growing. It is inspiring that, especially in some cases when the storage is not a big deal, we can easily scale up the tunable parameters to get the better downstream accuracy in the head-freezing/missing scenario (almost to be closer to full-tuning performance).

References

[1] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 2022. 1, 2, 4, 5

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 - mining discriminative components with random forests. In *ECCV*, 2014. 2, 4, 5

[3] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021. 4

[4] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014. 2, 4, 5

[5] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011. 2

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 4

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 4

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 4

[9] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE J-STARS*, 2019. 2

[10] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge J. Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, 2015. 2, 4, 5

[11] Saihui Hou, Yushan Feng, and Zilei Wang. Vegfru: A domain-specific dataset for fine-grained visual categorization. In *ICCV*, 2017. 2

[12] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *ICML*, 2019. 2, 4, 5

[13] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022. 2, 4, 5

[14] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *CVPR Workshops*, 2011. 2, 4, 5

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2, 4, 5

[16] 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. 4

[17] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 2, 4, 5

[18] Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2018. 5

[19] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008. 2, 4, 5

[20] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, 2012. 2

[21] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulic, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. In *EMNLP*, 2020. 2, 4, 5

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 4, 5

[23] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 2012. 2, 4, 5

[24] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 2, 4, 5

[25] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 2

[26] 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. 5