Structured Model Probing: Empowering Efficient Transfer Learning by Structured Regularization

Supplementary Material

A. Experimental Details

In this section, we provide the details of experiments performed in the main paper, including datasets used in experiments, implementation details and hyperparameter selection.

A.1. Datasets

We present detailed information on datasets used in the paper in Table 1.

A.2. Implementation Details

We utilized a multi-layer perceptron (MLP) architecture with a total of 8 fully connected layers as the non-linear part of the probing model. The hidden dimensions of all layers are set to 128. ReLU is used as activate function. Additionally, residual connections are integrated every two layers.

Further analyses on the probing model size are presented in Section C.4, where we demonstrate that the findings made in the main paper are maintained across various probing model sizes. In addition, our proposed decoupled loss function is analyzed in Section C.7 to exhibit its effectiveness compared to the traditional cross-entropy loss.

A.3. Hyperparameter Selection

Hyperparameters for SMP. For the hyperparameters involved during the training phase, we set the batch size to 128, the learning rate to 0.001, and training steps from $(500, 5000, 10000)$. Cosine learning rate annealing schedule is employed. SMP introduces two extra hyperparameters, the aggregated feature size and the regularization strength λ_1. There are two types of feature aggregation: aggregating along the token dimension (to preserve channel information) and aggregating along the channel dimension (to maintain spatial information). We apply 1D average pooling as the aggregating function and choose the aggregated feature size (token-wise size, channel-wise size) from $((768, 0), (768, 197), (768, 1970))$. We select the regularization strength λ_1 from $(5, 0.05)$. For other parameters, we simply set $\lambda_2 = 0.1$, $M_1 = 2$, $M_2 = 1$ for all experiments, which yields satisfactory performance since the strength of structured non-linearity regularizer is mainly depended on the norm of selected structures. We perform 5-fold cross validation to select these hyperparameters and provide the details of the hyperparameters used for each dataset in Table 1.

It is common for tuning or probing methods to require the extra hyperparameter selection on the validation set. For example, VPT [12] needs to select the token length and position. Head2Toe [6] requires the selection of the regularization coefficient, target feature size, and the fraction of retained features. SMP does not introduce more hyperparameters compared to these methods. As SMP only requires a single forward propagation on the pre-trained model to extract features, it is much faster than tuning-based methods when performing hyperparameter selection.

Hyperparameters for compared methods. We carefully select hyperparameters for compared methods. For Fine-tuning, we use AdamW as the optimizer and search learning rate from {5e-2, 1e-2, 5e-3, 1e-3, 5e-4, 1e-4, 5e-5, 1e-5} and weight decay from {1e-2, 1e-3, 1e-4, 0}. For VPT, we use SGD as the optimizer and follow the recommended regime of original paper [12], searching learning rate from {50, 25, 10, 5, 2.5, 1, 0.5, 0.25, 0.1, 0.05}, searching weight decay as fine-tuning, searching prompt tokens from {1, 5, 10, 50, 100, 200}. For Head2Toe [6], we choose the learning rate from {0.1, 0.01} and training steps from {500, 5000}, and search regularization coefficient from{0.001, 0.000001}, target feature sizes from {768, 15360, 32448} and the fraction of retained features from {0.0005, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1}, following the recommended regime in the paper. All results are from the best epoch selected by the validation set.

Table 1. Detailed information and selected hyperparameters on evaluated datasets.

Dataset	# Classes	Train	Val	Test	Steps	Aggregated Size	λ_1
Visual Task Adaptation Benchmark (VTAB-1k [27])							
CIFAR-100 [17]	100	800/1000	200	10,000	5,000	(768, 197)	5
Caltech101 [7]	102			6,084	5,000	(768, 197)	0.05
DTD [4]	47			1,880	5,000	(768, 197)	5
Flowers102 [21]	102			6,149	5,000	(768, 197)	0.05
Pets [22]	37			3,669	5,000	(768, 197)	5
SVHN [20]	10			26,032	5,000	(768, 1970)	5
Sun397 [26]	397			21,750	5,000	(768, 197)	5
Patch Camelyon [24]	2	800/1000	200	32,768	500	(768, 197)	5
EuroSAT [9]	10			5,400	5,000	(768, 197)	5
Resisc45 [3]	45			6,300	5,000	(768, 197)	0.05
Retinopathy [14]	5			42,670	500	(768, 197)	0.05
Clevr/count [13]	8	800/1000	200	15,000	5,000	(768, 0)	5
Clevr/distance [13]	6			15,000	5,000	(768, 197)	5
DMLab [1]	6			22,735	5,000	(768, 197)	5
KITTI/distance [8]	4			711	5,000	(768, 197)	5
dSprites/location [19]	16			73,728	5,000	(768, 197)	0.05
dSprites/orientation [19]	16			73,728	5,000	(768, 0)	0.05
SmallNORB/azimuth [18]	18			12,150	5,000	(768, 1970)	5
SmallNORB/elevation [18]	9			12,150	5,000	(768, 197)	0.05
Few-shot learning tasks							
Food101 [2]	101	1/2/4/8/16 (per class)	20,200	30,300	5,000	(768, 197)	5
CUB-200-2011 [25]	200		600	5,794			
Oxford Pets [22]	37		736	3,669			
Stanford Dogs [15]	120		1,200	8,580			
Stanford Cars [16]	196		815	8,041			
Full-size downstream tasks							
CUB-200-2011 [25]	200	5,394	600	5,794	10,000	(768,197)	5
Flowers102 [21]	102	1,020	1,020	6,149			0.05
Oxford Pets [22]	37	2,575	1,105	3,669			0.05
Stanford Dogs [15]	120	10,800	1,200	8,580			5
Stanford Cars [16]	196	7,329	815	8,041			0.05
NABirds [10]	555	21,536	2,393	24,633			5
Food101 [2]	101	75,750	/	25,250			5
DTD [4]	47	1,880	1,880	1,880			5
Magnetic [11]	6	938	/	406			0.05

B. Experiments on More Transfer Scenarios

In this section, we present full results of experiments on more transfer scenarios, including performance on larger downstream datasets, different pre-trained models, and different architecture.

B.1. Performance on Larger Downstream Datasets

We evaluate our method on 9 full-size downstream datasets, including CUB-200-2011 [25], Flowers102 [21], Oxford Pets [22], Stanford Dogs [15], Stanford Cars [16], NABirds [10], Food101 [2], DTD [4], and Magnetic [11]. The results are presented in Table 2, and suggest that even in the larger data regime, SMP still achieves superior or competitive performance compared to tuning methods in most cases. It can be observed that probing-based methods perform worse than tuning-based methods on Stanford Cars, which is a fine-grained classification dataset. This indicates there is room for improvement for the dataset that has low inter-class variance. However, even in this case, SMP improves the accuracy by 25.9% with respect to Linear. Overall, the competitive performance of SMP on full-size downstream datasets suggests that it can be a strong alternative to tuning methods.

B.2. Performance on Different Pre-trained Models

To verify the effectiveness of SMP on different pre-trained models, we conduct VTAB-1k experiments on various models, i.e., ViT-B/16 and ViT-L/14, which are pre-trained via CLIP method [23]. The results are presented in Table 3. As the results indicate, the employment of stronger pre-trained model results in better performance of SMP, and using larger models further boosts its performance. Across the three data groups, SMP consistently outperforms Fine-tuning, Linear and Head2Toe. While VPT performs better than SMP on the Natural group, SMP outperforms VPT significantly on the Structured group. By using more detailed hyperparameter selection such as exploring more aggregated feature sizes and varying regularization strength, the performance of SMP on the Natural group can be further improved. However, here we simply maintain the similar hyperparameter searching regime as the main paper. Overall, despite the usage of stronger pre-trained models, SMP continuously outperforms compared methods in terms of mean accuracy, owing to the incorporation of diverse features and non-linear transformation, as well as its flexible framework.

B.3. Performance on Different Architecture

Our proposed SMP method is versatile and can be applied not only to vision transformers but also to convolutional neural networks. In the case of convolutional neural networks, the features are extracted from each layer and then aggregated by 2D average pooling. We perform experiments on ImageNet pre-trained ResNet-50, and present the results in Table 3. It is evident from the results that SMP outperforms other methods on pre-trained convolutional neural networks and achieves the highest mean accuracy across all three data groups. This suggests that SMP is effective for convolutional neural networks as well.

Table 2. Test accuracy (%) on 9 full-size downstream datasets. **Bold** represents the best results and underline represents the 2nd best results.

	CUB-200-2011	Flowers102	Oxford Pets	Stanford Dogs	Stanford Cars	NABirds	Food101	DTD	Magnetic	Average
Fine-tuning	87.3	98.8	92.9	89.4	**84.5**	82.7	**89.9**	72.2	**98.0**	88.4
VPT	88.5	99.0	93.5	90.2	83.6	**84.2**	88.8	74.2	96.5	**88.7**
Linear	85.3	97.9	92.0	86.2	51.3	75.9	87.2	66.5	93.4	81.7
SMP	**89.2**	**99.4**	**93.9**	**92.1**	77.2	81.1	88.9	**79.5**	97.3	**88.7**

Table 3. Test accuracy (%) on the VTAB-1k benchmark using different pre-trained models. * indicates results are obtained from [12] and † indicates results are obtained from [6].

	Natural								Specialized					Structured									
	CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Mean (Natural)	Camelyon	EuroSAT	Resisc45	Retinopathy	Mean (Specialized)	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Elev	Mean (Structured)	Mean (All)
ViT-B/16 (CLIP pre-trained)																							
Tuning Methods																							
Fine-tuning	43.1	84.4	62.1	84.7	74.6	90.4	35.5	67.8	74.4	95.6	80.0	73.6	80.9	60.1	57.5	45.3	56.4	62.6	25.7	30.2	27.6	45.7	61.2
VPT	**70.7**	**92.5**	75.5	**95.5**	**89.8**	91.4	**54.0**	**81.4**	77.9	94.4	**88.9**	74.6	84.0	76.5	61.2	40.8	**79.8**	**85.0**	41.1	31.7	39.7	57.0	**71.6**
Probing Methods																							
Linear	64.0	90.5	73.4	95.6	86.2	57.1	53.9	74.4	81.1	92.6	87.3	74.1	83.8	50.2	41.1	40.6	63.6	32.0	41.5	15.7	29.7	39.3	61.6
Head2Toe	63.2	90.2	77.6	94.6	79.2	77.8	49.2	76.0	81.1	93.1	87.7	**74.9**	84.2	54.0	59.2	**42.8**	78.2	50.6	49.7	25.0	43.0	50.3	66.9
SMP	62.9	91.6	**78.5**	95.3	86.9	74.8	49.9	77.2	**81.2**	**95.8**	88.1	72.0	**84.3**	**84.1**	**63.0**	42.7	74.8	76.2	**51.2**	**37.1**	**54.6**	**60.5**	**71.6**
ViT-L/14 (CLIP pre-trained)																							
Tuning Methods																							
Fine-tuning	55.6	88.3	64.2	90.5	81.9	92.6	37.5	72.9	87.4	95.8	86.4	73.6	85.8	74.8	58.9	52.8	**81.3**	69.0	23.8	24.2	31.6	52.0	66.9
VPT	**80.7**	**94.3**	79.8	**98.6**	**94.1**	**94.1**	**59.7**	**85.9**	**83.2**	95.5	**92.7**	**76.3**	**86.9**	62.1	60.7	34.1	66.1	**84.8**	50.2	25.0	33.6	52.1	71.9
Probing Methods																							
Linear	72.4	92.8	78.9	98.2	91.0	64.4	55.6	79.0	82.2	95.3	91.2	74.5	85.8	51.6	44.2	42.2	63.7	30.8	45.1	14.2	28.6	40.1	64.1
Head2Toe	71.9	92.6	79.5	97.7	84.5	78.9	52.4	79.7	**83.2**	96.9	91.4	74.5	86.5	54.5	52.9	45.3	75.8	49.0	**54.3**	27.0	41.4	50.0	68.6
SMP	72.0	93.5	**80.5**	98.3	91.3	83.8	55.3	82.1	83.1	**97.0**	91.3	74.4	86.5	**89.4**	**63.5**	**45.5**	75.7	75.7	52.2	**35.5**	**53.2**	**61.3**	**74.3**
ResNet-50 (ImageNet-1k pre-trained)																							
Tuning Methods																							
Scratch†	11.0	37.7	23.0	40.2	13.3	59.3	3.9	26.9	73.5	84.8	41.6	63.1	65.8	38.5	54.8	35.8	36.9	87.9	37.3	20.9	36.9	43.6	42.1
Fine-tuning†	33.2	84.6	54.5	85.2	79.1	87.8	16.6	63.0	82.0	92.5	73.3	73.5	80.3	54.6	63.7	46.3	72.1	**94.8**	47.1	35.0	33.3	55.9	63.6
VPT*	49.5	87.7	63.4	80.9	**88.3**	60.3	33.7	66.3	72.5	90.4	72.8	73.6	77.3	39.9	51.4	36.3	62.5	43.2	23.2	17.0	26.8	37.5	56.5
Probing Methods																							
Linear†	48.5	86.0	67.8	84.8	87.4	47.5	34.4	65.2	**83.2**	92.4	73.3	73.6	80.6	39.7	39.9	36.0	66.4	40.4	37.0	19.6	25.5	38.1	57.0
Head2Toe†	47.1	**88.8**	67.6	85.6	87.6	**84.1**	32.9	70.5	82.1	94.3	76.0	74.1	81.6	55.3	59.5	**43.9**	**72.3**	64.9	**51.1**	**39.6**	43.1	53.7	65.8
SMP	**84.7**	85.1	**69.2**	**86.0**	86.8	79.6	**34.9**	**75.2**	82.4	**94.5**	**76.5**	**74.3**	**82.0**	**82.5**	**63.9**	40.0	66.9	80.9	47.3	37.6	**47.9**	**58.4**	**69.5**

C. Extended Analysis

In this section, we present further analyses to provide a more comprehensive understanding of the proposed method.

C.1. Informativeness of Extracted Features

Figure 1 demonstrates the informativeness of the extracted features on the VTAB-1k benchmark, ordered from easy to difficult, according to the domain similarity defined in the main paper. The results indicate that, for easier tasks, deeper layers of the pre-trained model yield more informative features. This observation justifies the superior performance of linear probing for easy tasks. However, for the more difficult tasks, the informativeness of features is uniform across various layers. Thus, it is inadequate to rely only on features from the final layer for adapting to difficult tasks, as it fails to capture the diverse information contained in the intermediate layers. Consequently, incorporating additional features into the probing model becomes crucial for difficult tasks.

C.2. Visualization of Group Norms

In Figure 2, we show the norm of each structure group obtained from the weight matrix of the linear classifier, which is regularized by structured sparsity regularizer. The results validate the observation of the informativeness of extracted features.

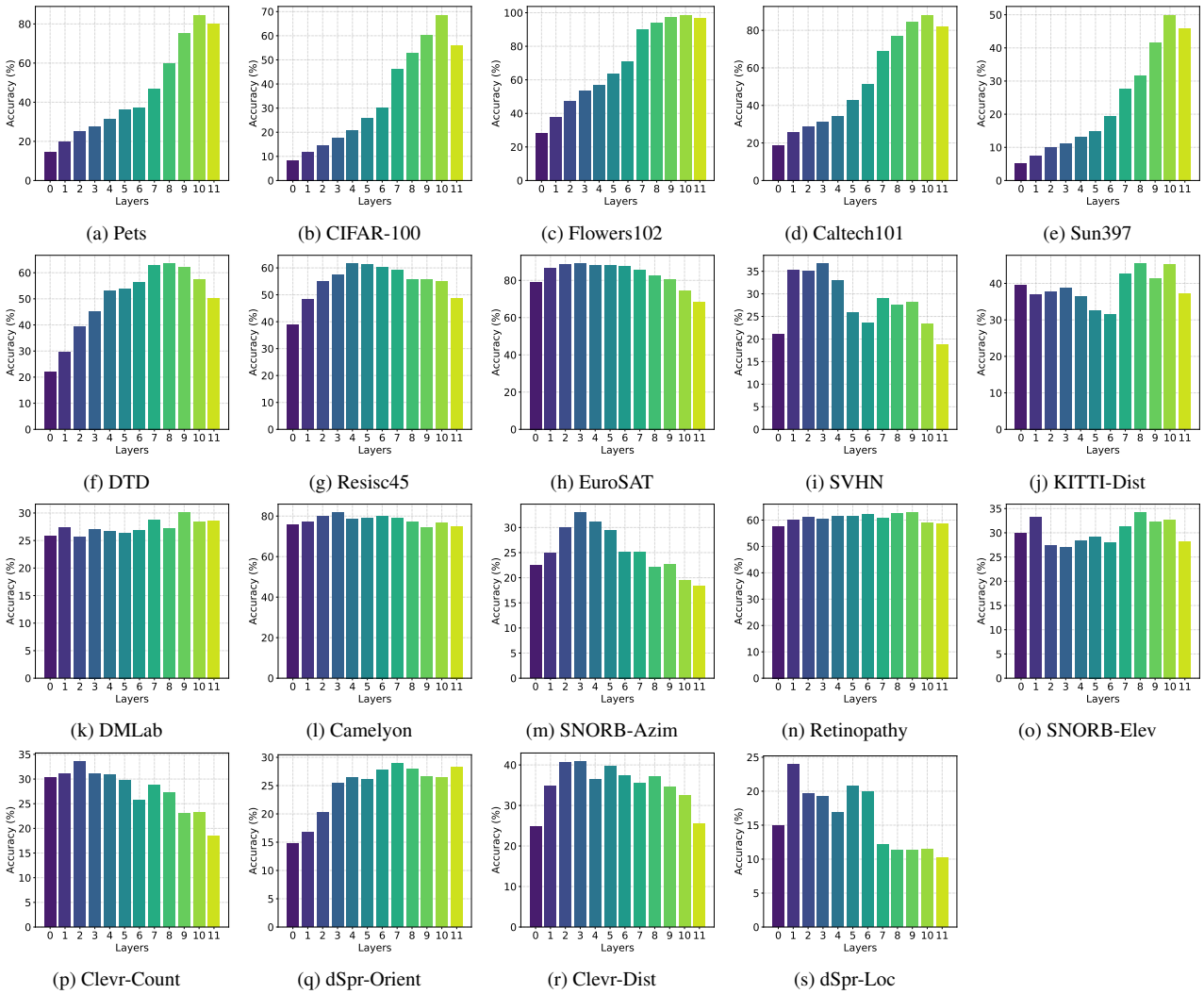
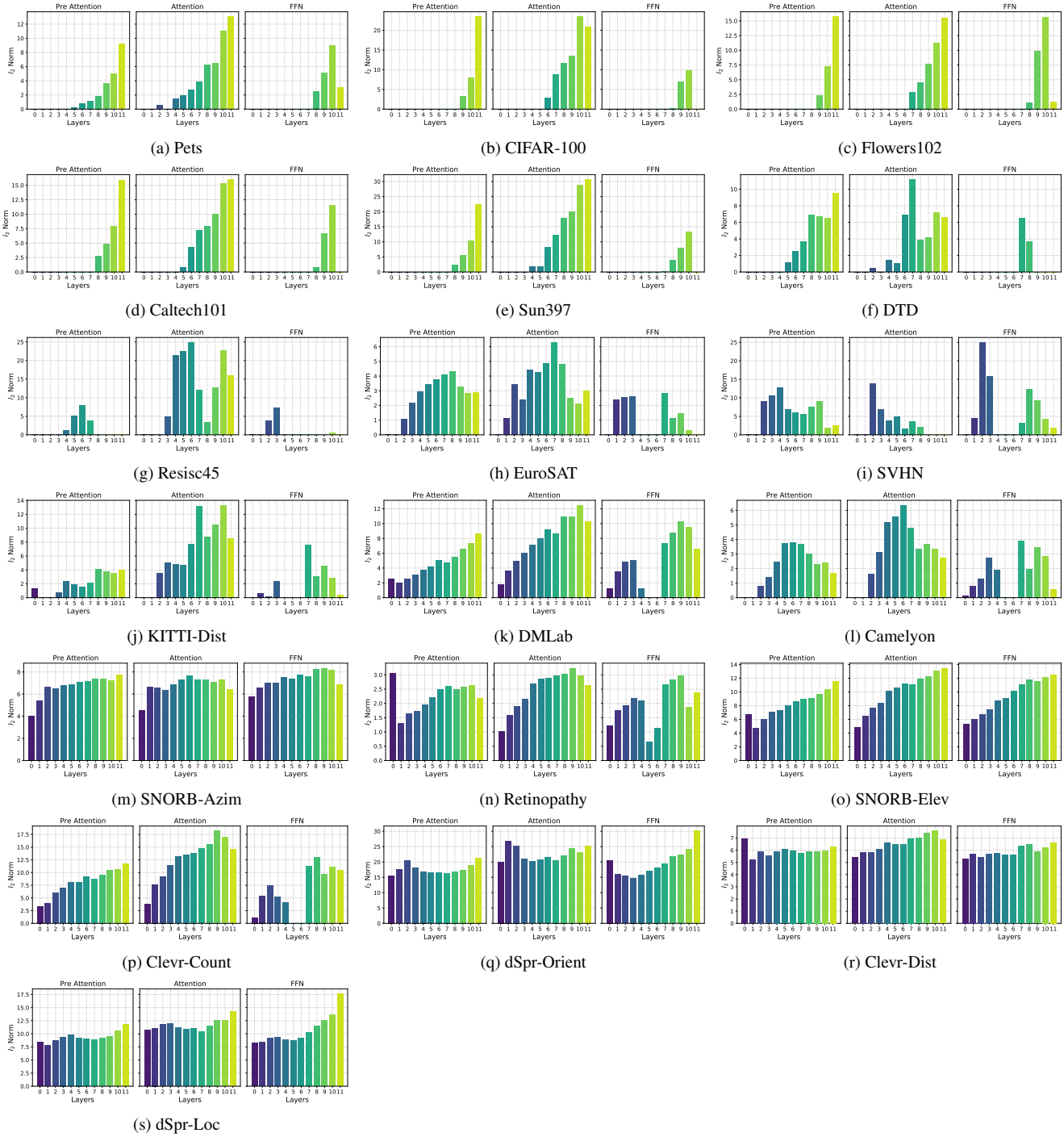


Figure 1. Informativeness of features extracted from different layers. Datasets are ordered from easy tasks (left) to difficult tasks (right) according to domain similarity.



(a) Pets (b) CIFAR-100 (c) Flowers102

(d) Caltech101 (e) Sun397 (f) DTD

(g) Resisc45 (h) EuroSAT (i) SVHN

(j) KITTI-Dist (k) DMLab (l) Camelyon

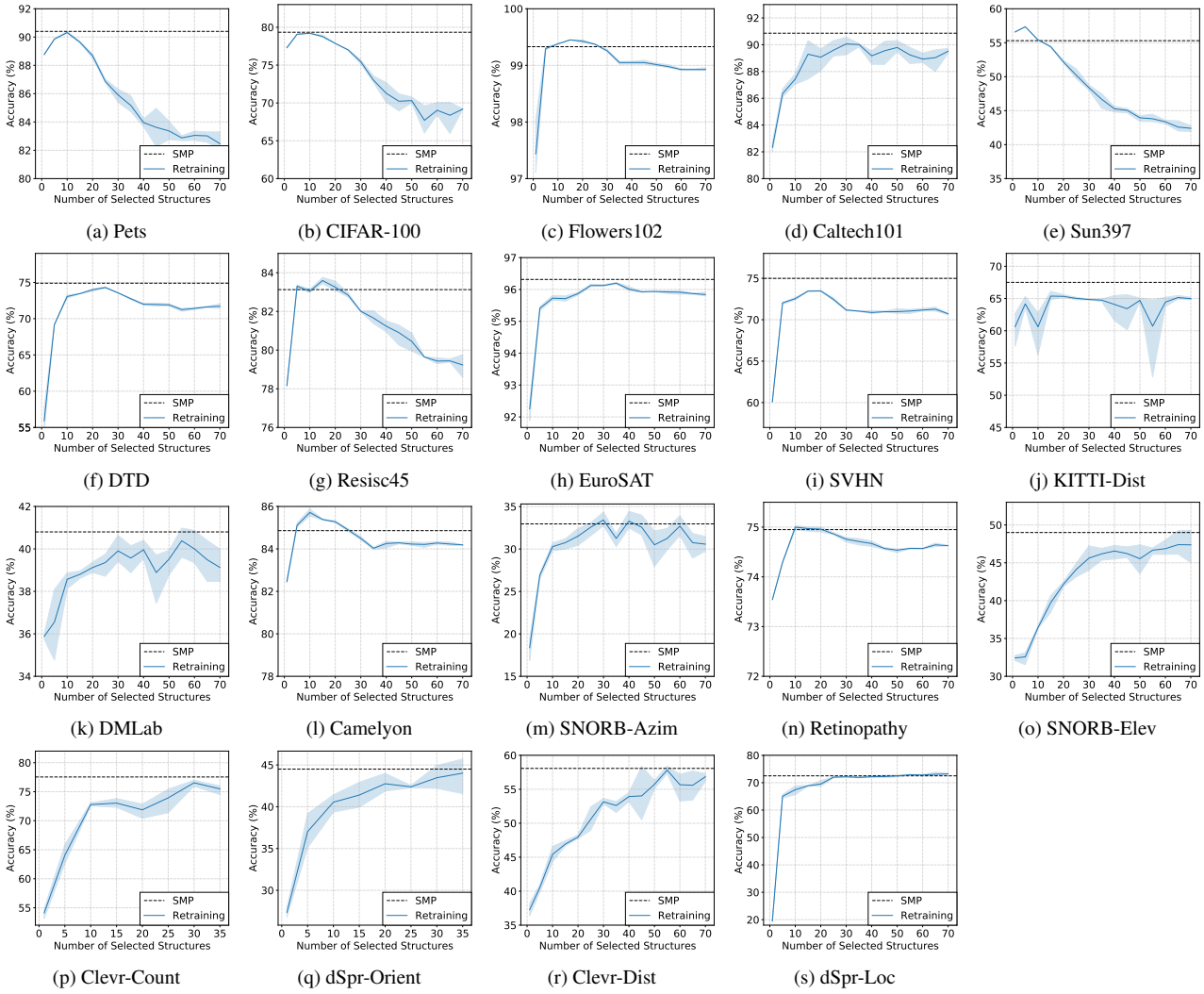
(m) SNORB-Azim (n) Retinopathy (o) SNORB-Elev

(p) Clevr-Count (q) dSpr-Orient (r) Clevr-Dist

(s) dSpr-Loc

Figure 2. ℓ_2 norm of each structure. Datasets are ordered from easy tasks (left) to difficult tasks (right) according to domain similarity.

For easy tasks, only deeper layers contain useful information, thus structures from these layers exhibit non-zero norms. As for difficult tasks, the information contained in intermediate layers is also crucial, and the norms of these structure groups are similar, resulting in larger average norms. This further verifies our idea of structured non-linearity regularizer. In particular, for tasks with a few selected features, the regularizer induces strong regularization effects, resulting in a mostly linear model. On the other hand, for tasks that require diverse features, the regularizer produces a more complex model to be more non-linear.



(a) Pets (b) CIFAR-100 (c) Flowers102 (d) Caltech101 (e) Sun397

(f) DTD (g) Resisc45 (h) EuroSAT (i) SVHN (j) KITTI-Dist

(k) DMLab (l) Camelyon (m) SNORB-Azim (n) Retinopathy (o) SNORB-Elev

(p) Clevr-Count (q) dSpr-Orient (r) Clevr-Dist (s) dSpr-Loc

Figure 3. Retraining performance by gradually incorporating structures with larger ℓ_2 norm obtained by SMP. Datasets are ordered from easy tasks (left) to difficult tasks (right) according to domain similarity.

C.3. Retraining Probing Model

We present the complete results of gradually incorporating selected features in a retrained probing model in Figure 3. The structures are selected by their ℓ_2 norms. During the retraining stage, we control the non-linearity of probing model through validation set, and choose the best probing model, to verify the effect of varying selected structure numbers. The results further confirm our motivation. For easy tasks, there are redundant features that could lead to overfitting, and a few selected features are sufficient to achieve satisfactory performance; incorporating more features results in performance degradation. In contrast, difficult tasks require diverse features extracted from different layers, and incorporating more features consistently shows performance improvement. This validates the efficacy of the proposed structured sparsity regularizer: in general, SMP can perform embedded structure selection and obtain high performance without retraining a probing model.

C.4. Impact of Different Probing Model Sizes

We investigate the effect of changing the size of probing model. We vary the probing model size by changing the hidden layers of the MLP, and using MLP with all extracted features as probing model (like $\text{MLP}_{\textit{Struc.}}$ in the main paper). The results are presented in Table 4. It can be verified that with different sizes of the probing model, the trends are still consistent with our observation in the main paper: on easy tasks, the non-linear MLP model yields worse performance compared with linear

Table 4. Performance analyses on the VTAB-1k benchmark using ViT/B-16 pretrained on ImageNet-21k.

	Natural								Specialized					Structured									
	CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Mean (Natural)	Camelyon	EuroSAT	Resisc45	Retinopathy	Mean (Specialized)	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Elev	Mean (Structured)	Mean (All)
Impact of Different Probing Model Sizes																							
Linear$_{Struc.}$	72.9	89.5	72.5	98.9	83.6	71.6	42.7	76.0	84.3	96.0	81.8	75.0	84.3	48.2	52.5	39.4	66.7	48.7	36.7	32.3	37.7	45.3	64.8
MLP$_{Struc.}$ (MLP-2)	69.7	88.9	70.7	98.6	83.2	70.5	34.6	73.8	84.0	95.6	76.3	74.2	82.5	59.1	52.8	39.5	60.9	51.1	41.3	31.9	37.6	46.8	64.2
MLP$_{Struc.}$ (MLP-4)	69.5	88.5	70.0	98.0	84.3	70.4	32.8	73.3	83.0	95.5	72.4	73.4	81.1	73.0	54.1	40.2	42.8	65.3	42.8	32.1	40.8	48.9	64.7
MLP$_{Struc.}$	66.8	88.7	70.3	97.4	82.1	71.1	33.3	72.8	81.6	95.9	77.0	73.5	82.0	77.2	57.7	39.3	66.5	72.3	44.1	32.7	47.7	54.7	67.1
Impact of Feature Aggregation Designs																							
SMP (*token-wise*)	79.2	90.8	74.7	99.3	90.2	55.4	55.4	77.9	83.9	96.4	83.2	75.2	84.7	77.8	57.9	40.1	67.7	62.5	43.1	23.6	47.5	52.5	68.6
SMP (*channel-wise*)	24.1	65.0	29.2	58.1	30.3	58.7	11.8	39.6	72.9	76.5	39.5	72.1	65.2	37.1	51.5	32.5	53.7	72.0	23.5	28.6	39.3	42.3	46.1
SMP (*both*)	79.3	90.9	74.9	99.3	90.4	60.1	55.3	78.6	84.8	96.3	83.1	75.0	85.1	74.1	58.0	40.8	67.5	72.5	35.3	27.2	49.0	53.5	69.4
Impact of Different Structures																							
SMP w/o Pre Attn	79.5	91.2	74.5	99.3	90.1	73.1	54.7	80.3	84.0	96.3	82.7	74.5	84.4	77.6	56.5	40.0	66.9	72.3	42.4	32.9	48.5	54.6	70.4
SMP w/o Attn	78.1	90.7	74.5	99.1	89.7	72.9	54.2	79.9	83.9	96.3	81.8	74.7	84.1	78.3	57.1	36.3	65.0	70.4	39.5	33.7	48.8	53.6	69.7
SMP w/o FFN	78.4	90.3	74.4	99.2	90.1	74.0	54.4	80.1	84.1	96.4	83.1	75.0	84.7	78.0	54.2	40.5	68.6	73.5	43.7	32.2	49.5	55.0	70.5
Impact of Loss Function																							
SMP w/ CE	69.5	91.2	72.0	99.5	87.1	74.6	38.8	76.1	84.4	96.1	82.3	73.4	84.0	57.8	52.2	39.8	64.4	69.7	40.7	32.7	45.4	50.4	66.9
SMP	79.3	90.9	74.9	99.3	90.4	75.0	55.3	80.7	84.8	96.3	83.1	75.0	84.8	77.5	58.0	40.8	67.5	72.5	44.5	33.0	49.0	55.4	70.9

model; while on difficult tasks, the non-linear model has better performance. We can also observe that the performance degradation of simple MLP models on easy tasks is less than complex MLP models. However, the simple MLP models achieve less performance improvement on difficult tasks. Our proposed structured non-linearity regularizer enables us to leverage a complicated probing model without worrying about performance degradation.

C.5. Impact of Feature Aggregation Designs

The main goal of performing feature aggregation is to maintain the diversity in aggregated features while minimizing the redundancy in raw features. In this section, we conduct ablation studies on feature aggregation designs adopted in SMP, to validate their impact on different datasets. The results are presented in Table 4. *Token-wise* refers to performing 1D average pooling through the token dimension, aiming to preserve channel information, and resulting in 768-dimensional features. *Channel-wise* represents performing 1D average pooling through the channel dimension, intending to retain spatial information, thereby generating 197-dimensional features. *Both* means performing 1D average pooling through both dimensions, and concatenating the aggregated 768- and 197-dimensional features.

Generally, we find that channel information (token-wise aggregation) is important for all datasets. While for datasets that are sensitive to spatial location, such as SVHN, dSpr-Loc, and sNORB-Azim, it is crucial to incorporate spatial information (channel-wise aggregation), to further boost the performance. However, this degrades the performance on Clevr-Count and dSpr-Ori. This may be because these extra location-based features lead to overfitting on these datasets. Therefore, we determine the aggregated feature size through the validation set.

C.6. Impact of Different Structures

To enhance the diversity of features, we integrate features extracted from various structures of the transformer blocks, including features before self-attention, features after self-attention, and features after Feed Forward Network (FFN). Despite these structures are located in nearby positions, they still contribute to the diversity of features since these features are generated by different parameters. Our designed structured sparsity regularizer allows us to conduct model fitting and feature selection simultaneously, thereby obviating the need for the costly manual selection of candidate structures. We present a thorough

Table 5. Domain similarity measurements on the VTAB-1k benchmark using ViT/B-16 pretrained on ImageNet-21k.

	Natural							Specialized				Structured							
	CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Elev
Domain Similarity	69.6	67.7	53.5	69.5	82.1	16.3	54.3	7.4	18.6	42.9	1.7	-0.6	-22.6	8.0	15.8	-54.4	-2.0	2.7	-0.4
Label-Feature Correlation ($\times 10^{-2}$)	47.7	71.9	37.5	66.6	61.0	2.4	46.3	11.4	37.1	26.1	2.7	4.5	4.7	5.1	23.1	0.9	3.5	4.2	4.5

ablation study on different structures in Table 4. The results demonstrate that incorporating all structures yields the best mean results, since our flexible framework automatically selects the most suitable candidate structures via structured regularization.

C.7. Impact of Loss Function

We propose a decoupled cross-entropy loss by decomposing the loss function into two components, as shown in Eq.(9) of the main paper. We apply gradient stop on the linear part of the second component because our model, $f(x) = \theta^\top x + f_W(x)$, consists of both a linear and a non-linear part. Directly employing cross-entropy loss on the output of the model may cause the structured sparsity regularizer to lose its effectiveness in controlling the complexity of $\|\theta\|$, under the influence of the non-linear part. We present the experimental results of using single cross-entropy loss on the output of the model in Table 4 (represented as w/ CE). The results show that performance degradation occurs without the proposed decoupled loss, thereby validating the effectiveness of our design.

C.8. Label-Feature Correlation as Domain Similarity Measure

Except for the domain similarity defined as Eq.(1) in the main paper, we also utilize label-feature correlation (LFC) proposed in [5] as domain similarity measure:

$$\text{LFC} = \frac{(K_X - \mu_X) \cdot (K_Y - \mu_Y)}{\|K_X - \mu_X\|_2 \|K_Y - \mu_K\|_2} \tag{1}$$

where X represents feature matrix, Y represents labels, $K_X \in \mathbb{R}^{N \times N}$ is the feature similarity matrix, $K_Y \in \mathbb{R}^{N \times N}$ is the label similarity matrix where $(K_Y)_{i,j} = 1$ if y_i equals y_j, and -1 otherwise. μ_X denotes the mean of the entries of K_X and μ_Y represents the mean of K_Y. We calculate LFC on the VTAB-1k benchmark using ViT/B-16 pre-trained on ImageNet-21k, and report the results in Table 5. LFC exhibits a high Spearman rank correlation coefficient (0.854) with the domain similarity score adopted in the main paper, showing the high correlation between these two measures.

References

[1] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv:1612.03801*, 2016. 2

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 - mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, pages 446–461, 2014. 2, 3

[3] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 2017. 2

[4] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3606–3613, 2014. 2, 3

[5] Aditya Deshpande, Alessandro Achille, Avinash Ravichandran, Hao Li, Luca Zancato, Charless C. Fowlkes, Rahul Bhotika, Stefano Soatto, and Pietro Perona. A linearized framework and a new benchmark for model selection for fine-tuning. *arXiv:2102.00084*, 2021. 9

[6] Utku Evci, Vincent Dumoulin, Hugo Larochelle, and Michael C. Mozer. Head2Toe: Utilizing intermediate representations for better transfer learning. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 6009–6033, 2022. 1, 4

[7] Li Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPR Workshop)*, pages 178–178, 2004. 2

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research*, 2013. 2

[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 Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 595–604, 2015. 2, 3

[11] Yibin Huang, Congying Qiu, Yue Guo, Xiaonan Wang, and Kui Yuan. Surface defect saliency of magnetic tile. In *14th IEEE International Conference on Automation Science and Engineering (CASE)*, pages 612–617, 2018. 2, 3

[12] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision (ECCV)*, 2022. 1, 4

[13] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[14] Kaggle and EyePacs. Kaggle diabetic retinopathy detection, 2015. 2

[15] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition (CVPR Workshop)*, 2011. 2, 3

[16] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshop (ICCV Workshop)*, pages 554–561, 2013. 2, 3

[17] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009. 2

[18] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004. 2

[19] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset, 2017. 2

[20] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning (NIPS Workshop)*, 2011. 2

[21] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *6th Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 2, 3

[22] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3498–3505, 2012. 2, 3

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8748–8763, 2021. 3

[24] Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2018. 2

[25] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011. 2, 3

[26] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3485–3492, 2010. 2

[27] 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:1910.04867*, 2019. 2