Adapting Shortcut with Normalizing Flow: An Efficient Tuning Framework for Visual Recognition

Yaoming Wang[1] Bowen Shi[1] Xiaopeng Zhang[2]
Jin Li[1] Yuchen Liu[1] Wenrui Dai[1] Chenglin Li[1] Hongkai Xiong[1] Qi Tian[2]✉
[1]Shanghai Jiao Tong University [2]Huawei Cloud
{wang_yaoming, sjtu_shibowen, deserve_lj, liuyuchen6666, daiwenrui, lcl1985, xionghongkai}@sjtu.edu.cn; zxphistory@gmail.com, tian.qi1@huawei.com

Abstract

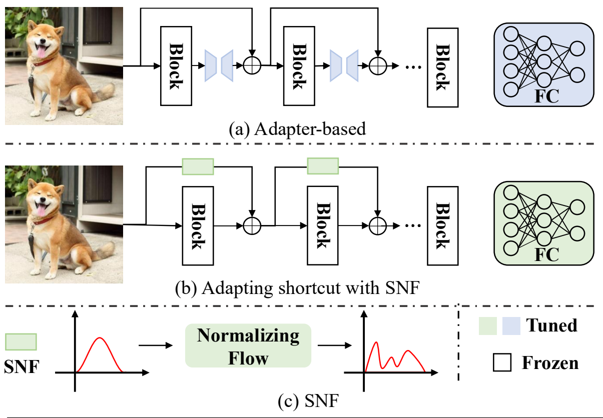
*Pretraining followed by fine-tuning has proven to be effective in visual recognition tasks. However, fine-tuning all parameters can be computationally expensive, particularly for large-scale models. To mitigate the computational and storage demands, recent research has explored Parameter-Efficient Fine-Tuning (PEFT), which focuses on tuning a minimal number of parameters for efficient adaptation. Existing methods, however, fail to analyze the impact of the additional parameters on the model, resulting in an unclear and suboptimal tuning process. In this paper, we introduce a novel and effective PEFT paradigm, named **SNF** (**S**hortcut adaptation via **N**ormalization **F**low), which utilizes normalizing flows to adjust the shortcut layers. We highlight that layers without Lipschitz constraints can lead to error propagation when adapting to downstream datasets. Since modifying the over-parameterized residual connections in these layers is expensive, we focus on adjusting the cheap yet crucial shortcuts. Moreover, learning new information with few parameters in PEFT can be challenging, and information loss can result in label information degradation. To address this issue, we propose an information-preserving normalizing flow. Experimental results demonstrate the effectiveness of SNF. Specifically, with only 0.036M parameters, SNF surpasses previous approaches on both the FGVC and VTAB-1k benchmarks using ViT/B-16 as the backbone. The code is available at https://github.com/Wang-Yaoming/SNF*

1. Introduction

The conventional paradigm in visual recognition has been to fine-tune deep neural networks that are pretrained on large-scale datasets, leveraging general visual representations to achieve impressive gains for downstream tasks [4]. However, as the size of these models continues to scale up, fine-tuning all of their parameters has become memory costly and even prohibitive. To address this issue, Parameter-Efficient Fine-Tuning (PEFT) approaches have emerged as a prevalent paradigm. These approaches [3, 14–16, 42] achieve consistent performance gains by only tuning a small set of parameters.

One pioneering PEFT approach is Visual Prompt Tuning (VPT) [16], which injects learnable prompt tokens into the input image tokens and trains these prompt tokens with pretrained model weights being frozen. Despite the improved performance compared with linear probing, VPT ig-

✉Corresponding author: Qi Tian. This work was done when Yaoming Wang worked as an intern at Huawei Cloud.



Method	Params	FGVC	VTAB-1k	ImagNet	DG
VPT-deep [16]	0.60	89.1	69.4	70.5	26.0
LoRA [15]	0.25	-	72.3	70.8	27.4
NOAH [42]	0.39	-	73.2	71.5	32.8
SNF-shallow	**0.036**	**89.8**	**73.5**	**78.5**	**34.0**

Figure 1. Illustration of (a) Adapter-based approach (b) Adapting shortcut with SNF (c) SNF. The bottom table is the overview of average accuracy vs. the number of learnable parameters on FGVC, VTAB-1k, ImageNet, and Domain Generalization (DG) tasks.

nores the data distribution shift between the pretrained and downstream data, which would mislead the feature extraction and propagate the error through the model. On the other hand, adapter-based approaches [3, 14, 15] leverage memory-efficient bottleneck design to adjust the features learned in pretrained models and can alleviate distribution shift to some extent. However, its bottleneck design sacrifices the information learned from the pretrained model and results in inferior performance. Besides, since it is hard to influence the over-parameterized model with very few parameters, these approaches inevitably require a certain amount of parameters to perform the feature adjustment for over-parameterized residual connection. This violates the original intent of PEFT, *i.e.*, tuning the pretrained model with as few parameters as possible.

In this paper, we instead pay attention to the shortcut and revisit it from the perspective of the Lipschitz [39] constraint. We reveal that most layers of the pretrained models meet the Lipschitz constraint due to the *implicit* Lipschitz regularization used in pretraining [21], *e.g.*, the weight decay loss and data augmentation, while the *implicit* regularization cannot guarantee all the layers are Lipschitz-constrained. In the close-set pretraining stage where the input distribution is fixed, models can still be robust in performance in such a limited input field. However, this robustness can be disrupted when applied to various downstream datasets, as the data distribution may differ from that of the close-set pretraining stage. In this case, layers that are not Lipschitz-constrained may propagate errors throughout the entire frozen model. Thus, to ensure a stable and robust feature extraction procedure, we need to adapt the output of these layers. However, adapting the output from the over-parameterized residual part using limited parameters can be challenging. As a result, we choose to adapt the simple shortcut feature instead, which can also help to regularize the Lipschitz constant for the model.

When the data distribution changes, the feature extracted by the frozen model becomes biased. In addition to regularizing the Lipschitz constant, we need to adjust the biased distribution to improve the feature extraction capability. However, learning new information from downstream data with limited learnable parameters is challenging, and conducting adaptation in an information-losing way like adapter-based approaches [3, 14, 15] is not a feasible option. To this end, we propose a novel and effective PEFT paradigm, namely Shortcut adaption via Normalization Flow (SNF), that leverages normalizing flows to adapt the shortcut. Specifically, we freeze the residual part and compute the transformation of the shortcut using planar flow layers. The planar flow has a wonderful property that its Jacobian determinant is easy to compute, allowing us to constrain the Lipschitz constant by regularizing the log Jacobian determinant.

Experimental results demonstrate that SNF achieves remarkable performance improvements compared to linear probing and even outperforms full fine-tuning with only 0.036M learnable parameters. Our approach also outperforms state-of-the-art PEFT approaches, achieving better performance with fewer parameters. Specifically, on FGVC datasets, our 1-layer flow model outperforms VPT-deep with only 5% learnable parameters (0.036M vs. 0.66M). Additionally, extensive experiments on the VTAB benchmark and few-shot learning demonstrate that SNF can surpass existing approaches in the low-data regime. Furthermore, SNF is a general approach that can be applied to various pretrained backbones, including both vision transformers and convolutional neural networks.

The contributions of this paper are summarized below:

- We reveal that the unconstrained Lipschitz in the pretrained model will cause error propagation when transferred to downstream datasets and directly adapting the over-parameterized residual part is hard.
- We analyze that the loss of feature information undermines the learning of label information when adjusting the feature distribution in PEFT.
- We propose a novel PEFT framework that leverages normalizing flows to learn an invertible transformation of shortcuts. The proposed method boosts PEFT with extremely few parameters.

2. Related Work

Parameter-Efficient Fine-Tuning. Parameter-Efficient Fine-Tuning (PEFT) has dominated recent advances in Natural Language Processing (NLP) [10,14,15,19,22,43]. The pioneer works can be categorized into the token-based [19, 22] and adapter-based [14, 15, 31, 32]. Recently, PEFT has gradually gained attention in visual recognition tasks as the growing model size and the growing downstream tasks. Similar to PEFT in NLP, existing visual PEFT approaches can also be divided into token-based and adapter-based. For token-based, Visual Prompt Tuning (VPT) tries to replicate the success of NLP in visual recognition. Specifically, VPT injects prompt tokens to the input sequence of vision transformers (ViTs) and then trains these prompt tokens while freezing the origin ViT. Pro-tuning [26] further replaces prompt tokens with learnable small networks.

Adapter-based approaches leverage memory-efficient bottleneck architecture after each block [14] or as the additional residual [3,15] of the frozen model and can achieve much better performance compared with token-based approaches. However, the bottleneck design will sacrifice the information learned by the pre-trained model and result in poor performance. Besides, adapter-based approaches pay attention to adapting residuals and need sufficient parameters, which violates the principle of PEFT: tuning the model with as few parameters as possible. Recently, some work [42] proposes to combine the token-based and adapter-based approaches with neural architecture search. However, the costly search process still hinders the development of PEFT. Concurrent work [20] performs scaling and shifting on each operation in pretrained models, which is memory-costly and rough. In this paper, we propose to utilize normalizing flow to adapt shortcuts and achieve impressive performance with extremely few parameters.

Normalizing Flow. Generative methods encounter significant challenges due to the high dimensionality and complex structure of the underlying data distribution. Normalizing Flow based models [5, 6, 18, 35] provide a promising solution, that parameterizes a complex distribution via multiple layers of invertible transformations. In this paper, we focus on how to transform a distribution into the desired form without loss of information. Thus we introduce invertible normalizing flow to parameter efficient fine-tuning.

3. Methodology

3.1. Preliminary: Shortcut Connection

Given a model $F(\cdot)$ that consists of L residual layers as $\mathcal{F}(\cdot, W_i), i = 1, \cdots, L$. Denote the input of layer l as x_l, then the output of the lth layer can be defined as:

$$x_{l+1} = h(x_l) + \mathcal{F}(x_l, W_l). \tag{1}$$

Here $h(\cdot)$ is the shortcut connection function. To avoid gradient vanishing, skip connection is usually adopted for $h(\cdot)$, the final output y of model F is:

$$y = x_l + \sum_{i=l}^{L} \mathcal{F}(x_i, W_i). \tag{2}$$

Considering a little distortion δ for feature x_l, which is natural in neural network training. Let x_l^1 and x_l^2 to be any two points in the δ-neighborhood of the input x_l. Then we have the deviation of the output of l layer as:

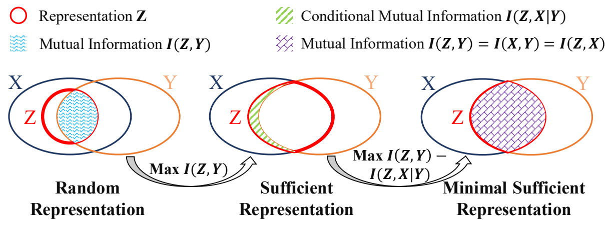
$$||x_{l+1}^2 - x_{l+1}^1|| = ||x_l^2 - x_l^1 + \mathcal{F}(x_l^2, W_l) - \mathcal{F}(x_l^1, W_l)|| \tag{3}$$

Since the model parameters are pretrained on large datasets with data augmentation and weight decay, which implies an implicit Lipschitz constraint [9], we assume that most layers of the model meet the K-Lipschitz constraint:

$$||\mathcal{F}(x_l^2, W_l) - \mathcal{F}(x_l^1, W_l)|| \leq K_l ||x_l^2 - x_l^1|| \tag{4}$$

Where K_l is usually the Frobenius norm of the parameter W_l, which is the upper bound of the spectral norm [39]. Then the output of layer l also meets the Lipschitz constraint as:

$$||x_{l+1}^2 - x_{l+1}^1|| \leq (1 + K_l) ||x_l^2 - x_l^1|| \tag{5}$$



○ Representation Z
▨ Mutual Information $I(Z,Y)$
▨ Conditional Mutual Information $I(Z,X|Y)$
▨ Mutual Information $I(Z,Y) = I(X,Y) = I(Z,X)$
X
Y
Z
Max $I(Z,Y)$
Max $I(Z,Y) - I(Z,X|Y)$
Random Representation
Sufficient Representation
Minimal Sufficient Representation

Figure 2. The illustration of Assumption 1, *i.e.*, the Venn diagram for input X, label Y, and representation Z (Best viewed in color).

Bring Eq. 5 into Eq. 2, we can get that the output meets

$$||y^2 - y^1|| \leq \left(\prod_{i=l}^{L} (1 + K_i) \right) ||x_l^2 - x_l^1|| \tag{6}$$

When layer j cannot meet K-Lipschitz constraint, *i.e.*, K_j is not a constant, then the output is not stable with distortion.

3.2. Adapting Shortcut for Lipschitz Constraint

For the pre-trained dataset, the unconstrained layers usually result in a limited impact on performance, as the model is robust in a with-in dataset training manner. However, in parameter efficient fine-tuning, the model needs to fit the unseen downstream dataset when most of the model parameters are frozen. Then the unconstrained layers will cause catastrophic error propagation in the model. To resolve this problem, an intuitive approach is to adapt the residual layer $\mathcal{F}(\cdot, W)$ to reduce error propagation. Unfortunately, since the residual layer is frozen, it is hard to adapt the over-parameterized residual layer with few parameters.

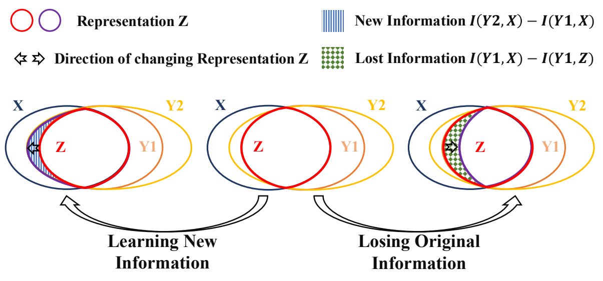
Different from the over-parameterized residual layer, the shortcut is simple but can be used to alleviate error propagation. Specifically, if we map the shortcut with function $g_\phi(\cdot)$. Then Eq. 6 can be rewritten as:

$$||y^2 - y^1|| \leq \left(\prod_{i=l}^{L} (K_i^\phi + K_i) \right) ||x_l^2 - x_l^1|| \tag{7}$$

Regularizing K_i^ϕ can gradually mitigate error propagation throughout the model.

3.3. Adapting Shortcut with Normalizing Flow

As the data distribution is changed, the feature extracted by the frozen model is also biased. Then we need not only to regularize the Lipschitz constant but also adapt the feature distribution. In this subsection, we first analyze that the biased posterior distribution hinders migration performance. Then we shed light on the property that needs to be satisfied for the transformation of the biased posterior distribution. Specifically, the transformation should not lose the information of the pre-trained representation. Finally,



Representation Z
New Information $I(Y2, X) - I(Y1, X)$
Direction of changing Representation Z
Lost Information $I(Y1, X) - I(Y1, Z)$
X
Y2
Z
Y1
X
Y2
Z
Y1
X
Y2
Z
Y1
Learning New Information
Losing Original Information

Figure 3. The illustration of Proposition 1. Learning new representation information is hard with few learnable parameters while losing representation information leads to losing information about the label (Best viewed in color).

we propose that the normalizing flow meets this property, and use it to adjust the biased distribution for PEFT.

Given the feature z, the classification model predicts the distribution $p(y|z)$ that indicates the probability of the input belongs to the ground truth label. Thus, the loss function of maximum likelihood estimation (MLE) is $\mathbb{E}_{q(z)} \log(p(y|z))$, where $q(z)$ is the posterior distribution learned from data. In traditional fine-tuning, the distribution can be directly learned by a powerful neural network. However, in parameter-efficient tuning, the powerful neural network is trained on other datasets and is frozen, then the learned posterior distribution is biased. Therefore, how to adjust the biased posterior distribution is the key to parameter-efficient tuning.

Assumption 1. *For pretrained model, the learned feature z is the minimal sufficient representation for learning information of label y from input x as $I(x, y) = I(x, z) = I(y, z)$. The intuitive diagram is illustrated in Fig. 2.*

This assumption is natural for well pre-pretrained models as the bottleneck architecture and the regularization tricks, *e.g.*, drop-path are employed during pre-training.

Assumption 2. *For downstream datasets, the label information y_2 is assumed to contain the label information y_1 of the pretrained large dataset as $y_1 \subseteq y_2$. And the mutual information $I(x, y_1) \leq I(x, y_2)$ for input x.*

This assumption is natural in visual recognition tasks. As downstream datasets are usually subdivisions under existing categories of the large pretrained dataset.

Proposition 1. *For downstream datasets, we can hardly learn new information about the input when most features are frozen, then if we learn representation z_{new} in a way that information is lost, i.e., $I(z_{new}, x) < I(z, x)$, the representation will also lose information about the label y_2 as $I(z_{new}, y_2) < I(z, y_2)$.*

Proof. Please refer to the supplementary material. □

We also illustrate this proposition in Fig. 3. This means we need to adjust the posterior distribution without loss of information. To this end, We leverage normalizing flow, an ideal family of variational distributions that is flexible enough to map any complex distribution with invertible mapping, to adjust the distribution.

Proposition 2. *For $z_1, z_2 \in \mathbb{R}^N$, the mutual information $I(z_1, z_2)$ equals to $I(z_1, z_1)$ when the mapping $z_2 = f_\psi(z_1)$, $f_\psi : \mathbb{R}^N \to \mathbb{R}^N$ is invertible and smooth.*

Proof. Please refer to the supplementary material. □

The normalizing flow describes the transformation of a probability density $z^0 \to z^J$ with a chain of K invertible and smooth mapping f_j as:

$$z^J = f_J \circ \cdots \circ f_2 \circ f_1(z^0) \tag{8}$$

And the new probability density z_j is:

$$\ln q(z^J) = \ln q(z^0) - \sum_{j=1}^{J} \ln \left| \det \frac{\partial f_j}{\partial z_{j-1}} \right| \tag{9}$$

Here, $\ln \left| \det \frac{\partial f_j}{\partial z_{j-1}} \right|$ is the log Jacobian and we can regularize the log Jacobian to control the K-Lipschitz constraint (details in Sec. 3.5) as elaborated in Sec. 3.2.

3.4. Implementation Details

In this subsection, we introduce the implementation details of the proposed paradigm that utilizes normalizing flows for adapting shortcuts. For each layer of the proposed paradigm, we adopt planar flow for adapting shortcuts Given the feature $z \in \mathbb{R}^{N*d}$, we consider a family of transformations of the form:

$$f(z) = z + \lambda \cdot h(\gamma^T \cdot z + \beta) \tag{10}$$

And the log Jacobian for planar transformation is

$$\ln \left| \det \frac{\partial f}{\partial z} \right| = \ln \left| I + \lambda^T \cdot h'(\gamma^T \cdot z + \beta) \cdot \gamma \right| \tag{11}$$

where $\lambda, \gamma, \beta \in \mathbb{R}^d$, $\cdot$ is the Hadamard product and $h(\cdot)$ is a smooth non-parametric non-linearity. In practice, the parameters λ, γ, β would be broadcast to keep the same shape with z. Following the practice of [35], we also employ one affine transformation at the beginning as $f(z) = \gamma^T \cdot z + \beta$, where $\gamma, \beta \in \mathbb{R}^d$ and $\cdot$ is still the Hadamard product. The log Jacobian for affine transformation is $\ln \left| \det \frac{\partial f}{\partial z} \right| = \ln \gamma$.

During the training procedure, we freeze the residual part and compute the transformation of the shortcut through the proposed flow in each layer. Specifically, for the l-th layer, we transform the shortcut feature from $q(z_l^0)$ to $q(z_l^J)$

through J flow layers. Then, for the L layer model, the posterior distribution is factorized as:

$$q(z) = \prod_{i=1}^{L} q(z_i^J) \tag{12}$$

The final objective function is:

$$\mathbb{E}_{q(z)} \log(p(y|z)) = \mathbb{E}_{q(z_i^J)\cdots q(z_L^J)} \log(p(y|z_i^J, \cdots, z_L^J)) \tag{13}$$

Given the ground truth of the label y, the objective function can be calculated like common visual recognition tasks.

3.5. Analysis on the Jacobian Determinant

In this subsection, we come back to the Lipschitz constraint. We hope the Lipschitz constraint of the flow model is tighter than that of shortcuts as discussed in Sec. 3.2. Specifically, the Lipschitz constant of the flow layer is expected to be no more than 1. For traditional neural networks, this constraint is difficult to apply because the Lipschitz constant is related to the spectral norm of the parameter matrix, which is difficult to compute. Fortunately, our flow model has an easy-computed Jacobian determinant, which is helpful for regularizing the Lipschitz constant.

Proposition 3. *When* $\ln\left|\det \frac{\partial f}{\partial z}\right| \leq 0$ *holds, the Lipschitz constant for f will meet the constrain as* $K(f(\cdot)) \leq 1$.

Proof. Please refer to the supplementary material. □

From this way, we can regularize the log Jacobian determinant to be negative to meet the K-Lipschitz constraint. Specifically, we add the regularization loss as $\mathrm{ReLU}(\ln\left|\det \frac{\partial \mathrm{f}}{\partial \mathrm{z}}\right|)$ after the objective function in Eq. 13. Here ReLU is the ReLU activation.

4. Experiments

4.1. Experimental Setup

Pre-trained Backbones. Following VPT [16], we choose ViT-B/16 [7] pretrained on ImageNet-21k as the backbone in most experiments. Besides, we also generalize our method to several different shortcut-equipped backbones, including CNN-based ResNet [11] backbone, and Swin Transformer(Swin-B) [23]. We also explore the effectiveness of tuning large models using ViT-L/16. The details of the different backbones are shown in Sec. 4.6.
Amount of Parameters. Since the linear layer is used for all approaches, including linear probing and fine-tuning, *we do not consider the linear layer when calculating the amount of parameters for a more intuitive comparison.*
Implementation Details. We implement the codes with the Pytorch [30] framework and use 1 Nvidia-V100 GPU for training. An AdamW [24] optimizer and a cosine decay learning rate schedule are used with 2×10^{-3} initial learning rate and 1×10^{-4} weight-decay for the default setting, except for VTAB-1k benchmark. The total training epochs are set to 100 with 10 warm-up epochs. We employ random resized crop and random horizontal flip as data augmentation for all the datasets following VPT [16]. For most experiments, we perform both SNF-shallow, which use 1-layer planar flow for adaption and SNF-deep, which use 5-layer planar flow. More details are in supplementary materials.

4.2. Experiments on FGVC Datasets

Datasets. FGVC Benchmark consists of 5 Fine-Grained Visual Classification tasks including CUB-200-2011 [37], NABirds [36], Oxford Flowers [28], Stanford Dogs [17] and Stanford Cars [8]. The details of these datasets are provided in supplementary materials.
Results. Illustrated in Tab. 1, SNF-shallow achieves better average accuracy (89.78% vs. 89.11%) with *only 5% parameters* compared to VPT-deep. Besides, our SNF-deep can further boost the average accuracy to 90.74% and achieve 4 sota results among five datasets.

4.3. Experiments on VTAB-1k Benchmark

Datasets. VTAB-1k [40] benchmark is composed of 19 visual classification tasks, which are clustered into three groups: Natural, Specialized, and Structured. The Natural group contains natural images that come from various kinds of visual common concepts, including generic and fine-grained objects. The Specialized group is composed of images captured via specialized equipment, such as medical imaging equipment and remote sensing imaging equipment. The structured group is uncommon and often abstract classification concepts, essentially depth estimation or object counting. Top-1 accuracy is the performance metric.
Results. Illustrated in Tab. 2, our SNF-shallow achieves 73.5% averaged top-1 accuracy with only 0.036M params, which already outperforms other sota approaches like NOAH (73.2% accuracy, 0.39M params) and VPT-deep (69.4% accuracy, 0.60M params). Note that it is extremely challenging to tune the pretrained backbone with less than 0.1 M params on difficult VTAB-1k Benchmark, *e.g.*, VPT-shallow uses 0.07M params to tune the model but only achieves 64.9% averaged top-1 accuracy. While SNF-shallow exhibits very impressive performance, which proves that adaption on shortcut with flow model is direct to the essence of PEFT. Besides, our SNF-deep can further improve the performance and achieves a new state-of-art (sota) performance of 74.1% averaged top-1 accuracy and achieves 10 sota and 6 sub-sota out of 19 tasks.

4.4. Experiments on Few-Shot Learning

Datasets. Following [42], we choose five fine-grained visual recognition datasets including Food101 [1], Stand-

Table 1. Per-task fine-tuning results on FGVC datasets. The backbone is ViT-B/16, and we ignore the linear layer when calculate the amount of learnable parameters. Bold represents the best performance, underlined represents the second best performance.

Methods	Params(M)	CUB-200-2011	NABirds	Oxford Flowers	Stanford Dogs	Stanford Cars	Mean
Linear probing	0	85.3	75.9	97.9	86.2	51.3	79.32
Full-tuning	85.8	87.3	82.7	98.8	89.4	84.5	88.54
Sidetune [41]	-	84.7	75.8	96.9	85.8	48.6	78.35
Adapter [14]	0.23	87.1	84.3	98.5	89.8	68.6	85.67
VPT-shallow [16]	<u>0.08</u>	86.7	78.8	98.4	**90.7**	68.7	84.62
VPT-deep [16]	0.66	88.5	84.2	99.0	<u>90.2</u>	<u>83.6</u>	89.11
Results of our methods							
SNF-shallow	**0.036**	<u>90.0</u>	<u>86.7</u>	<u>99.6</u>	89.3	83.3	<u>89.78</u>
SNF-deep	0.25	**90.2**	**87.4**	**99.7**	89.5	**86.9**	**90.74**

Table 2. Per-task fine-tuning results on VTAB-1k benchmark. The backbone is ViT-B/16, and we ignore the linear layer when calculate the amount of learnable parameters. Bold represents the best performance, underlined represents the second best performance.

		Natural							**Specialized**				**Structured**								
Methods	#Params(M)	Cifar100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	Average
Traditional Fine-tuning																					
Full	85.8	68.9	87.7	64.3	97.2	86.9	87.4	38.8	79.7	95.7	84.2	73.9	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	65.57
Linear	0	63.4	85.0	63.2	97.0	86.3	36.6	51.0	78.5	87.5	68.5	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	52.94
Other approaches																					
Bias [2]	0.10	72.8	87.0	59.2	97.5	85.3	59.9	51.4	78.7	91.6	72.9	69.8	61.5	55.6	32.4	55.9	66.6	40.0	15.7	25.1	62.1
VPT-shallow [16]	<u>0.07</u>	77.7	86.9	62.6	97.5	87.3	74.5	51.2	78.2	92.0	75.6	72.9	50.5	58.6	40.5	67.1	68.7	36.1	20.2	34.1	64.9
VPT-deep [16]	0.60	78.8	90.8	65.8	98.0	88.3	78.1	49.6	81.8	96.1	83.4	68.4	68.5	60.0	46.5	72.8	73.6	47.9	**32.9**	37.8	69.4
LoRA [15]	0.25	67.1	91.4	69.4	98.8	<u>90.4</u>	85.3	54.0	84.9	95.3	<u>84.4</u>	73.6	**82.9**	**69.2**	<u>49.8</u>	78.5	75.7	47.1	31.0	<u>44.0</u>	72.3
NOAH [42]	0.39	69.6	92.7	70.2	<u>99.1</u>	<u>90.4</u>	86.1	53.7	84.4	95.4	83.9	**75.8**	<u>82.8</u>	<u>68.9</u>	**49.9**	81.7	**81.8**	48.3	32.8	**44.2**	73.2
Ours																					
SNF-shallow	**0.036**	**84.3**	<u>93.5</u>	**72.7**	**99.3**	**91.3**	<u>89.5</u>	<u>54.3</u>	**85.7**	<u>96.2</u>	**85.5**	74.1	81.1	61.0	48.9	<u>82.3</u>	75.4	**49.3**	31.1	41.7	<u>73.5</u>
SNF-deep	0.25	<u>84.0</u>	**94.0**	**72.7**	**99.3**	**91.3**	**90.3**	**54.9**	**87.2**	**97.3**	**85.5**	<u>74.5</u>	82.3	63.8	<u>49.8</u>	**82.5**	<u>75.8</u>	<u>49.2</u>	<u>31.4</u>	42.1	**74.1**

fordCars [8], OxfordFlowers102 [27], OxfordPets [29], and FGVCAircraft [25] for few-shot evaluation. We follow [33,42,45] and use 1, 2, 4, 8, and 16 samples from each class for training and perform the evaluation on the whole test sets. We use the few-shot sample partition file provided by NOAH [42] for fair comparisons.

Results. Our few-shot experiment *is based on SNF-shallow (0.036M params)* and the results are summarized in Fig. 4. Overall, SNF shows clear advantages in all settings over other PEFT methods and almost achieves the best performance on each dataset under each few-shot setting, except for 2-shot, 4-shot, and 8-shot on FGVCAircraft. Note that when the training examples are extremely few, *e.g.*, 1-shot, SNF-shallow achieves more significant performance gains (more than 10% performance gain on Flowers102, Food101, and OxfordPets), while Adapt-based approaches, such as NOAH, LoRA, and Adapter, exhibit very poor performance. This phenomenon proves that the information loss in adapt-based approaches injures the classification ability significantly, while SNF adapts the feature distribution without losing information, thus can leverage a few samples for performance improvement.

4.5. Experiments on Domain Generalization

Datasets. The DG experiment is designed to investigate whether the generalized features can be learned without the labels of the downstream datasets. Following [42, 44, 45], We first tune our model on the ImageNet-1k dataset with

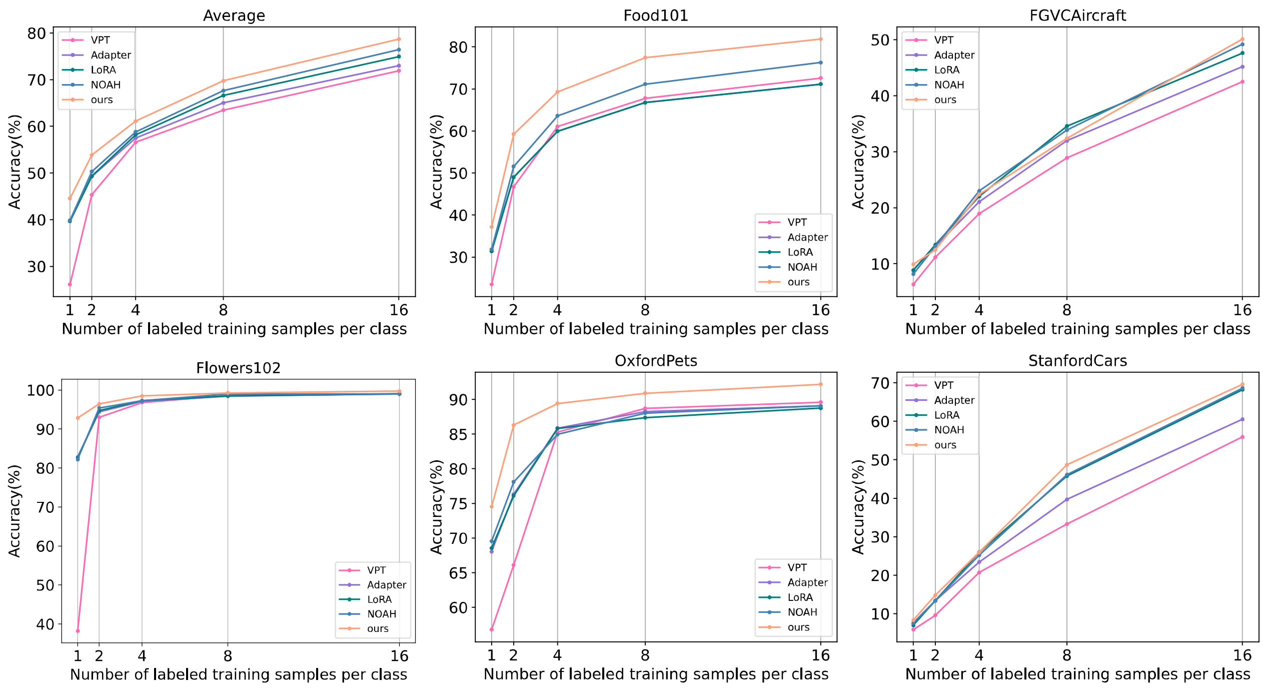


Figure 4. Results of few-shot learning on five daily visual recognition datasets (Best viewed in color).

Table 3. Results on domain generalization, the source domain is ImageNet-1k. The backbone is ViT-B/16.

Method	Source	$\rightarrow \mathcal{D}_{V2}$	$\rightarrow \mathcal{D}_{Sketch}$	$\rightarrow \mathcal{D}_{A}$	$\rightarrow \mathcal{D}_{R}$
Adapter [14]	70.5	59.1	16.4	5.5	22.1
VPT [16]	70.5	58.0	18.3	4.6	23.2
LoRA [15]	70.8	59.3	20.0	6.9	23.3
NOAH [42]	71.5	66.1	24.8	11.9	28.5
Ours	**78.5**	**66.4**	**27.0**	**12.2**	**30.4**

16-shot examples per class, and then directly evaluate the model on four variants of ImageNet: ImageNetV2 [34], ImageNet-Sketch [38], ImageNet-A [13], and ImageNet-R [12], processed with various domain shifts.

Results. Illustrated in Tab. 3, we implement SNF-shallow using ViT-B/16 as the backbone. SNF-shallow achieves a nearly 10% performance improvement on ImageNet-1k compared with other PEFT approaches, and when directly tested on four variants of ImageNet, SNF-shallow outperforms other approaches on all the target datasets. This indicates that SNF can help the model to adapt the feature distribution and the adaption is also general for domain shift.

4.6. Experiments on Different Backbones

We perform experiments on different backbones including ViT-B/16, ViT-L/16, Swin-B and CNN-based ResNet-50, ResNet-101 to verify the generalization of our method. We choose DTD and EuroSAT for comparisons.

Results. Illustrated in Tab. 4, our SNF-shallow can outperform the full-tuning performance on all the transformer-based backbone, *i.e.*, ViT-B, ViT-L and Swin-B, and SNF-deep can further improve the performance on both two datasets. Note that on large-size ViT-L/16, which has 306 M parameters, our SNF-shallow only tunes 0.1 M parameters and outperforms full-tuning significantly. Since fine-tuning becomes more expensive as model size increases, the generation ability of our SNF on large models is more important and the results are more impressive.

For CNN models, *i.e.*, ResNet-50 and ResNet-101, SNF can also achieve competitive results towards full-tuning with only a few parameters. The experiments prove that our approach is generalizable to all kinds of backbones with short-cut designs, while approaches like VPT and NOAH are only suitable for transformer-based frameworks.

4.7. Ablation Study

The length of the flow model. We first perform ablation studies on NABirds and Stanford Cars to explore the influence of the length of flow layers. Illustrated in Tab. 5, we experiment with 1, 3, 5, and 7 layers flow model respectively. Results show that as the number of layers increases, the flow model would have stronger distribution learning abilities, thus the accuracy of the two datasets is gradually improving. We finally choose the 1-layer flow for SNF-shallow and the 5-layer flow for SNF-deep for better parameters and accuracy trade-off.

Shortcut or Residual. To further verify the effectiveness of

Table 4. Results on different backbones.SNF-s† and SNF-d† are the abbreviation of SNF-shallow and SNF-deep, respectively.

Methods	ViT-B			ViT-L			Swin-B			ResNet-50			ResNet-101		
	DTD	EuroSAT	#Params(M)	DTD	EuroSAT	#Params(M)	DTD	EuroSAT	#Params(M)	DTD	EuroSAT	#Params(M)	DTD	EuroSAT	#Params(M)
Linear	63.2	87.5	0	67.7	94.6	0	77.1	94.0	0	60.5	88.4	0	58.7	87.8	0
Full	64.3	95.7	85.8	68.5	95.7	306	76.5	96.6	87.1	61.1	**95.9**	24.5	60.3	**95.6**	42.6
SNF-s†	**72.7**	96.2	0.036	73.5	96.6	0.10	77.4	96.8	0.05	62.8	95.6	0.03	62.5	**95.6**	0.06
SNF-d‡	**72.7**	**97.3**	0.25	**74.7**	**97.0**	0.69	**78.1**	**97.3**	0.34	**63.5**	95.7	0.18	**63.4**	95.1	0.43

Table 5. Ablation study on the length of flow model.

Methods	Params(M)	NABirds	Stanford Cars	Mean
1-layer	**0.036**	86.7	83.3	85.00
3-layer	0.14	87.2	84.9	86.05
5-layer	0.25	**87.4**	86.9	87.15
7-layer	0.36	**87.4**	**87.5**	**87.45**

Table 6. Experiments on adapting on shortcut or residual. Here SNF-s is the abbreviation of SNF-shallow and SNF-d is the abbreviation of SNF-deep.

Method	Caltech101		Cifar100	
	SNF-s	SNF-d	SNF-s	SNF-d
on shortcut	**93.5**	**94.0**	**84.3**	**84.0**
on residual	90.8	92.2	83.2	83.7

Table 7. Experiments on bottleneck. The backbone is ViT-B/16.

Method	Caltech101	Cifar100	#Params
SNF-shallow	93.5	**84.3**	**0.036**
SNF-deep	**94.0**	84.0	0.25
Bottleneck on shortcut	90.9	79.6	0.29

adapting on shortcut rather than the residual connection, we perform ablation study in Tab. 6. Overall, adapting on shortcut outperforms adapting on residual significantly. Specifically, when there are only a few learnable parameters (SNF-shallow), the gap between the adaptation shortcut and the residual is large. Introducing a large amount of learnable parameters (SNF-deep) can reduce the gap but it will reduce the parameter efficiency. This trend proves that we can hardly adapt residual connection especially when the learnable parameters are few.

Bottleneck or invertible flow. Adapter-based approaches tend to apply a bottleneck design, where one down-up architecture along with ReLU activation is used. We argue that this information-losing design is harmful to PEFT. To verify, we conduct ablations in Tab. 7. For a fair comparison, the middle dimension of the bottleneck is settled as 8 and the amount of learnable parameters is 0.29 M. Obviously, the bottleneck design is much inferior compared with

Table 8. Experiments on performing SNF in different location.

Methods	Params(M)	SNF-shallow	Params(M)	SNF-deep
First	0.003	90.8	0.021	91.3
Middle	0.003	88.5	0.021	89.3
Last	0.003	87.6	0.021	87.8
All	0.036	**93.5**	0.25	**94.0**

SNF (more than 3% deficiencies).

Error propagation in PEFT. We further perform experiments to explore whether there exists error propagation in PEFT. Illustrated in Tab. 8, we perform SNF in different locations of the backbone. Here, 'First', 'Middle', and 'Last' indicates using SNF in the first, middle, and last layer of the backbone, respectively. All the experiments are performed on Caltech-101. As shown, the performance is worse when SNF is used in the later layers of the backbone, which proves that error propagation exists in PEFT.

5. Conclusion

In this paper, we focus on Parameter-Efficient Fine-Tuning (PEFT) and identify the issues arising from the data distribution shift and Lipschitz unconstrained layers, which can lead to error propagation. Additionally, we demonstrate that information loss during adaption can harm PEFT performance. To address these challenges, we propose SNF, which leverages normalization flows with information-keeping transformations to adapt the shortcut. We conduct extensive experiments on various datasets, including FGVC, VTAB-1k, and Domain Generalization, and demonstrate the effectiveness of our approach.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant 62250055, Grant 61932022, Grant 61931023, Grant 61971285, Grant 61831018, Grant 62120106007, Grant 61972256, Grant T2122024, Grant 62125109, and in part by the Program of Shanghai Science and Technology Innovation Project under Grant 20511100100.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pages 446–461. Springer, 2014. 5

[2] Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. Tinytl: Reduce memory, not parameters for efficient on-device learning. *Advances in Neural Information Processing Systems*, 33:11285–11297, 2020. 6

[3] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022. 1, 2

[4] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014. 1

[5] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 3

[6] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 3

[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 *International Conference on Learning Representations*, 2021. 5

[8] Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. Fine-grained car detection for visual census estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017. 5, 6

[9] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110(2):393–416, 2021. 3

[10] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021. 2

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[12] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021. 7

[13] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021. 7

[14] 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 *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. 1, 2, 6, 7

[15] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 1, 2, 6, 7

[16] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022. 1, 5, 6, 7

[17] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, volume 2. Citeseer, 2011. 5

[18] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018. 3

[19] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 2

[20] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *arXiv preprint arXiv:2210.08823*, 2022. 3

[21] Hsueh-Ti Derek Liu, Francis Williams, Alec Jacobson, Sanja Fidler, and Or Litany. Learning smooth neural functions via lipschitz regularization. *arXiv preprint arXiv:2202.08345*, 2022. 2

[22] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021. 2

[23] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 5

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[25] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 6

[26] Xing Nie, Bolin Ni, Jianlong Chang, Gaomeng Meng, Chunlei Huo, Zhaoxiang Zhang, Shiming Xiang, Qi Tian, and Chunhong Pan. Pro-tuning: Unified prompt tuning for vision tasks. *arXiv preprint arXiv:2207.14381*, 2022. 2

[27] M-E Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1447–1454. IEEE, 2006. 6

[28] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008*

Sixth Indian Conference on Computer Vision, Graphics & Image Processing, pages 722–729. IEEE, 2008. 5

[29] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 6

[30] 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 H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[31] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*, 2020. 2

[32] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. *arXiv preprint arXiv:2007.07779*, 2020. 2

[33] 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 *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 6

[34] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019. 7

[35] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. 3, 4

[36] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 595–604, 2015. 5

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

[38] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32, 2019. 7

[39] GR Wood and BP Zhang. Estimation of the lipschitz constant of a function. *Journal of Global Optimization*, 8(1):91–103, 1996. 2, 3

[40] 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

[41] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In *European Conference on Computer Vision*, pages 698–714. Springer, 2020. 6

[42] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *arXiv preprint arXiv:2206.04673*, 2022. 1, 3, 5, 6, 7

[43] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*, 2021. 2

[44] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16825, 2022. 6

[45] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 6