This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.



Except for this watermark, it is identical to the accepted version;

the final published version of the proceedings is available on IEEE Xplore.

Conv-Adapter: Exploring Parameter Efficient Transfer Learning for ConvNets

Hao Chen1\*, Ran Tao 1, Han Zhang1, Yidong Wang2, Xiang Li1, Wei Ye2, Jindong Wang3, Guosheng Hu4, Marios Savvides 1

1 Carneige Mellon University, 2 Peking University, 3 Microsoft Research Asia, 4 Oosto

# Abstract

*While parameter efficient tuning (PET) methods have shown great potential with transformer architecture on Natural Language Processing (NLP) tasks, their effec- tiveness with large-scale ConvNets is still under-studied on Computer Vision (CV) tasks. This paper proposes Conv-Adapter, a PET module designed for ConvNets. Conv-Adapter is light-weight, domain-transferable, and architecture-agnostic with generalized performance on dif- ferent tasks. When transferring on downstream tasks, Conv- Adapter learns tasks-specific feature modulation to the in- termediate representations of backbones while keeping the*

0

-20



Conv-Adapter Linear-Probing Visual-Prompt Bias-Tuning Fine-Tuning

Relative Acc. Gain (%)

-40

-60

-80

10 1 100 101 102

Trainable Parameters (%)

*pre-trained parameters frozen. By introducing only a tiny amount of learnable parameters, e.g., only* 3*.*5% *full fine- tuning parameters of ResNet50. It can also be applied for transformer-based backbones. Conv-Adapter outperforms previous PET baseline methods and achieves comparable or surpasses the performance of full fine-tuning on* 23 *clas- sification tasks of various domains. It also presents superior performance on the few-shot classification with an average margin of* 3*.*39*%. Beyond classification, Conv-Adapter can generalize to detection and segmentation tasks with more than* 50% *reduction of parameters but comparable perfor- mance to the traditional full fine-tuning* [1](#_bookmark3)

# Introduction

As transfer learning [[54](#_bookmark73)] thrives, large-scale foundation models gradually dominate deep learning over the last few years [[3](#_bookmark24)]. Fine-tuning has become the de-facto paradigm adapting a foundation model pre-trained on a pretext task to various downstream tasks for both Computer Vision (CV) and Natural Language Processing (NLP). Albeit its sim- plicity and prominence, fine-tuning has been posing chal- lenges to development and deployment of the large-scale foundation models on downstream tasks with the drastic growth of computations and storage costs, as the parame-

\*[haoc3@andrew.cmu.edu](mailto:haoc3@andrew.cmu.edu)

1Code is available at: [https://github.com/Hhhhhhao/](https://github.com/Hhhhhhao/Conv-Adapter/tree/main) [Conv-Adapter/tree/main](https://github.com/Hhhhhhao/Conv-Adapter/tree/main)

Figure 1. Performance of Conv-Adapter compared to other trans- fer learning methods on ResNet-50 BiT-M. We compute the rela- tive performance gain w.r.t to fine-tuning and percentage of train- able parameters of the backbone (w/o linear head) on 23 im- age classification datasets from various domains to compute the results, with mean and standard deviation highlighted. Conv- Adapter achieves a superior trade-off between transfer accuracy and parameter efficiency.

ter size increases from millions [[19](#_bookmark40), [23](#_bookmark44), [46](#_bookmark65), [52](#_bookmark71)] to billions

[[5](#_bookmark26), [13](#_bookmark34), [14](#_bookmark35), [16](#_bookmark37), [34](#_bookmark55)–[36](#_bookmark56), [45](#_bookmark64)].

Parameter efficient tuning (PET), as an alternative to traditional fine-tuning, has become prevalent in NLP [[18](#_bookmark39), [22](#_bookmark43), [24](#_bookmark45), [30](#_bookmark51), [31](#_bookmark52)] for its efficiency and effectiveness. PET introduces a small number of learnable parameters to a pre-trained network, whose parameters are frozen, and learns the extra introduced parameters only. While attain- ing promising performance, especially for tasks of low-data regimes [[25](#_bookmark46), [62](#_bookmark81), [63](#_bookmark82)], PET modules for Convolutional Neu- ral Networks (ConvNets), the popular architectures for CV tasks, are still largely unstudied.

Prior arts on fine-tuning ConvNets to multiple visual do- mains are restrictive in generalization and parameter effi- ciency. Bias Tuning [[2](#_bookmark23)], which tunes only the bias terms of the backbone, might fail on domains with significant distri- bution shifts from the pre-training tasks. Residual Adapter

[[48](#_bookmark67)] and TinyTL [[7](#_bookmark28)] are mainly designed for small net-

works such as ResNet-26 [[19](#_bookmark40)] and MobileNet [[6](#_bookmark27), [23](#_bookmark44)]. It is prohibitive to scale these previous designs to larger Con- vNets [[36](#_bookmark56)] or more diverse domains [[60](#_bookmark79)]. Besides, previous PET methods [[18](#_bookmark39), [21](#_bookmark42), [24](#_bookmark45), [30](#_bookmark51), [31](#_bookmark52)] are mainly designed with

Transformer [[56](#_bookmark75)] architecture for NLP tasks [[5](#_bookmark26), [13](#_bookmark34)]. How- ever, it is not straightforward to apply Transformer-based PET to ConvNets because Transformers tokenize and se- quentialize the input and features, while ConvNets do not. Recent works [[1](#_bookmark22), [10](#_bookmark31), [25](#_bookmark46)] that attempt to use Prompt Tuning

[[30](#_bookmark51)] and Adapters [[21](#_bookmark42)] on CV tasks are also designed for Vision Transformers rather than ConvNets. Furthermore, the downstream CV tasks are usually more diverse with a larger domain gap compared with NLP [[45](#_bookmark64)]. These chal- lenges motivate us to design the architecture and adapting scheme of PET for ConvNets, which could make it trans- ferable to various CV tasks, including image classification, object detection, and semantic segmentation.

In this work, we narrow the gap of PET between NLP and CV with the proposal of **Conv-Adapter** – an adap- tion module that is light-weight, domain-transferable, and architecture-agnostic. Conv-Adapter learns task-specific knowledge on downstream tasks and adapts the intermedi- ate features of each residual block in the pre-trained Con- vNets. It has a bottleneck structure consisting of depth-wise separable convolutions [[23](#_bookmark44)] and non-linearity. Due to the variety of CV network architectures and tasks, we explore four adapting schemes of Conv-Adapter combining two de- sign perspectives - adapted representations and insertion form to verify the optimal tuning paradigm on ConvNets. We find it is essential for Conv-Adapter to maintain the lo- cality relationship when adapting intermediate feature maps for transferability. More importantly, Conv-Adapter can be formulated under the same mathematical framework as the PET modules used in the NLP field [[18](#_bookmark39)]. Conv-Adapter outperforms previous PET baselines and achieves similar or even better performance to the traditional *full* fine-tuning on 23 cross-domain classification datasets with an average of 3*.*5% of the backbone parameters using ResNet-50 BiT- M [[27](#_bookmark48)], as shown in Fig. [1](#_bookmark2). Conv-Adapter also well gener- alizes to object detection and semantic segmentation tasks with same-level performance to fully fine-tuning. To fur- ther understand Conv-Adapter, in addition, we empirically analyze the performance of Conv-Adapter with both the do- main shifting of datasets and the network weights shifting brought by fine-tuning. The core contributions of this work can be summarized as:

* To our knowledge, we are the first to *systematically* inves- tigate the feasible solutions of general parameter-efficient tuning (PET) for ConvNets. This investigation can nar- row the gap between NLP and CV for PET.
* We propose Conv-Adapter, a light-weight and plug-and- play PET module, along with four adapting variants fol- lowing two design dimensions - transferability and pa-

rameter efficiency. Meanwhile, we empirically justify several essential design choices to make Conv-Adapter effectively transferred to different CV tasks.

* Extensive experiments demonstrate the effectiveness and efficiency of Conv-Adapter. It achieves comparable or even better performance to full fine-tuning with only around 5% backbone parameters. Conv-Adapter also well generalizes to detection and segmentation tasks that re- quire dense predictions.

# Related Work

## Parameter Efficient Tuning for Transformers

Pre-trained Transformer models in NLP are usually of the size of billions of parameters [[5](#_bookmark26), [13](#_bookmark34), [16](#_bookmark37)], which makes fine- tuning inefficient as one needs to train and maintain a sepa- rate copy of the backbone parameters on each downstream task. Adapter [[21](#_bookmark42)] is first proposal to conduct transfer with light-weight adapter modules. It learns the task-specific knowledge and composes it into the pre-trained backbone [[43](#_bookmark62), [44](#_bookmark63)] when adapting to a new task. Similarly, LoRA in- troduces trainable low-rank matrices to each layer of the backbone model to approximate parameter updates. Differ- ent from inserting adaption modules to intermediate layers, Prefix Tuning [[31](#_bookmark52)] and Prompt Tuning [[30](#_bookmark51)], inspired by the success of textual prompts [[5](#_bookmark26), [33](#_bookmark54), [45](#_bookmark64)], prepend learnable prompt tokens to input and only train these tokens when transferring to a new task. More recently, a unified formu- lation of Adapter, LoRA, and Prefix Tuning is proposed in [[18](#_bookmark39)], where their core function is to adapt the intermedi- ate representation of the pre-trained model by residual task- specific representation learned by tuning modules.

Visual Prompt Tuning [[25](#_bookmark46)] is a recent method adapt- ing Prompt Tuning from NLP to Vision Transformers [[25](#_bookmark46)]. Bahng et. al. [[1](#_bookmark22)] also explores visual prompts in input pixel space for adapting CLIP models [[45](#_bookmark64)] and makes connec- tion with [[15](#_bookmark36)]. While showing promising results on Trans- formers, visual prompts on ConvNets presents much worse transfer results [[1](#_bookmark22), [25](#_bookmark46)], possibly due to the limited capacity of input space visual prompts. Conv-Adapter can adapt the intermediate features thus has larger capacity.

## Transfer Learning for ConvNets

While there is no straightforward approach to applying pre- vious PET methods designed for Transformers directly on ConvNets, several attempts have been made in prior re- search. BatchNorm Tuning [[40](#_bookmark59)] and Bias Tuning [[2](#_bookmark23)] only tune the batchnorm related terms or the bias terms of the pre-trained backbone. Piggyback [[39](#_bookmark58)] instead learns weight masks for downstream tasks while keeping the pre-trained backbone unchanged. They all have limited transferability and update partial parameters of the backbone.

More related to our work, Residual Adapter [[48](#_bookmark67)] ex-

plores inserting an extra convolutional layer of kernel size 1 to each convolutional layer in pre-trained ResNet-26 [[19](#_bookmark40)], either in parallel or in sequential, to conduct the multi- domain transfer. Similarly, TinyTL introduces extra resid- ual blocks to MobileNet [[6](#_bookmark27), [23](#_bookmark44)] for memory efficient on- device learning. Guo et. al. [[17](#_bookmark38)] proposes re-composing a ResNet with depth-wise and point-wise convolutions, and re-training only the depth-wise part during fine-tuning. RepNet [[59](#_bookmark78)] exploits a dedicated designed side network to re-program the intermediate features of pre-trained Con- vNets. Conv-Adapter differs from previous methods with a design that considers parameter efficiency and transferabil- ity from the internal architectures and adapting schemes. Besides, the proposed Conv-Adapter does not require tun- ing any backbone parameters to achieve comparable perfor- mance to fine-tuning.

# Method

## Preliminaries

Parameter efficient tuning (PET) methods [[21](#_bookmark42), [24](#_bookmark45), [25](#_bookmark46), [30](#_bookmark51), [31](#_bookmark52)] introduce learnable adapting modules plugged into the backbone that is frozen during tuning. From a unified point of view, the core function of the adaption modules is to learn task-specific feature modulations on originally hidden rep- resentations in the pre-trained backbone [[18](#_bookmark39)]. Specifically, considering an intermediate hidden representation **h** gener- ated by a layer or a series of layers with input **x** in a pre- trained network, the PET adaption module learns ∆**h** and updates **h** as:

**h** ←− **h** + *α* · ∆**h***,* (1)

where *α* could be a scalar [[24](#_bookmark45)] or a gating function [[31](#_bookmark52)]. Previous PET methods in NLP mainly follow a similar functional form for constructing ∆**h** – down-sampling pro- jection, non-linearity, and up-sampling projection. How- ever, they differ in 1) implementation (architecture) - the form of the projections and non-linearity, and 2) the adapt- ing scheme - which **h** in the model to adapt and compute

∆**h** from which representation. These differences charac- terize the adaptation to new tasks and robustness to out-of- distribution evaluation [[31](#_bookmark52)].

It is non-trivial to design effective PET methods for Con- vNets because previous PET modules are mainly devel- oped on Transformers rather than ConvNets. Besides, the components of the architecture and computation dynam- ics of ConvNets and Transformers are inherently different. Following the unified formulation of PET methods in Eq. ([1](#_bookmark6)), we propose **Conv-Adapter**. We construct the ∆**h** of Conv-Adapter similarly to previous PET methods and de- sign the adaption architecture and scheme on ConvNets from the perspective of transferability and parameter effi- ciency.

## Motivation

Before delving into the details of our design, we identify the essential difficulty that prevents utilizing prior arts directly on ConvNets as an adaption module and thus inspires us to propose Conv-Adapter. Conventionally, for ConvNets, **h** and ∆**h** are usually 3-dimensional structural features maps belonging to R*C×H×W* with *C* being the channel dimen-

sion and *H* × *W* being the spatial size of the feature maps. The difference in intermediate feature and processing

dynamics poses obstacles to transferability. For Trans- formers, **h** is whereas 2-dimensional sequential features in R*L×D* where *L* is the sequence length and *D* is the feature dimension. Previous PET modules for Transformers com- pute ∆**h** in various forms, e.g., linear layers over **h** [[21](#_bookmark42)] and self-attention over additional input prompts [[25](#_bookmark46), [30](#_bookmark51), [31](#_bookmark52)]. They can all process the sequential features globally with long-range dependencies as the computing blocks in Trans-

formers. Although it is possible to apply linear layers, or equivalently 1 × 1 convolutional layers [[48](#_bookmark67)], to adapt the feature maps of ConvNets, it is yet intuitive that this might produce inferior transfer performance due to the *loss of lo- cality*, which is encoded in the structural features maps by

convolutions of kernel size larger than 1. The *loss of lo- cality* results in a radical mismatch of the receptive field in ∆**h** and **h**, which might be destructive when adapting ConvNets on tasks with significant domain shifts. Apart from the receptive field mismatch, the spatial size of feature maps in ConvNets also significantly affects the transferabil- ity of adaption. Earlier attempts to use adapters to transfer ConvNets usually downsample the feature’s spatial size for memory and parameter efficiency. However, for CV tasks beyond image classification like segmentation, the spatial size matters for achieving good results [[9](#_bookmark30), [49](#_bookmark68)].

In summary, it is crucial to design the architecture and adapting scheme of the PET module computing ∆**h** for ConvNets to have the same spatial size of feature maps and the same receptive field of convolutions for transferability.

## Architecture of Conv-Adapter

Given the above challenges, we design our **Conv-Adapter** as a bottleneck structure, which is also widely used by PET methods of NLP tasks [[19](#_bookmark40), [21](#_bookmark42)]. However, our **Conv-Adapter** designs the bottleneck, particularly for Con- vNets. Precisely, it consists of two convolutional layers with a non-linearity function in-between. The first convo- lution conducts channel dimension down-sampling with a kernel size similar to that of the adapted blocks, whereas the second convolution projects the channel dimension back. For simplicity, we adopt the same activation function used in the backbone as the non-linearity at the middle of the bot- tleneck. The effective receptive field of the modulated fea- ture maps produced by Conv-Adapter is thus similar to that of the adapted blocks in the backbone. We do not change the

*Conv-Adapter*

𝐶𝑖𝑛 × 𝐻 × 𝑊

𝑾𝑑𝑜(𝑛

Depth-wise Conv.

Nonlinearity

𝐶𝑖𝑛

𝛾 × 𝐻 × 𝑊

𝑾𝑢𝑝

Point-wise Conv.

𝛼

× 𝐶𝑜𝑢𝑡 × 𝐻 × 𝑊

Figure 2. Architecture of Conv-Adapter, which has a bottleneck composed of depth-wise separable convolutions with non-linearity activation. *Cin*, *Cout*, *H*, *W* is set to keep the same as in back- bone. ***α*** and *γ* are hyper-parameters to tune.

spatial size of the feature maps for better transferability on dense prediction tasks. We adopt the depth-wise separable convolutions [[23](#_bookmark44)] for Conv-Adapter to reduce the parameter size further.

Figure [2](#_bookmark7) illustrates our Conv-Adapter architecture. For- mally, let the input feature map to the adapted blocks of the ConvNets be **z** ∈ R*Cin×H×W* and the output feature

maps be **h** ∈ R*Cout×H×W* , where *Cin* and *Cout* are the

channel dimension of the input and output to the adapted blocks respectively. Assuming the spatial size *H* × *W* of

the feature maps does not change along these blocks, we set

which consists of a series of convolutional layers (and sometimes pooling layers) and a residual identity branch, making it more difficult to use a single adapting scheme to various architectures.

To explore the effective adapting schemes of using Conv- Adapter to tune a ConvNet, we study it mainly from two perspectives, similar to [[18](#_bookmark39)], 1) the location of adaptation in pre-trained ConvNets – which intermediate representation **h** to adapt, and 2) the insertion form of Conv-Adapter – how to set the input **z** to Conv-Adapter to compute ∆**h**. From the location perspective, we study plugging Conv-Adapter to each (inverted) residual block [[7](#_bookmark28)] or to each functioning

*K* × *K* convolutional layer within a residual block [[17](#_bookmark38)]. From the insertion perspective, Conv-Adapter can be in-

serted either in parallel or in sequential to the modified com- ponents, with the input to Conv-Adapter being **x**, the input to the modified components, or being **h** itself, respectively. Combining the design dimension from these two perspec- tives, we propose 4 variants of adapting schemes with Conv- Adapter: **Convolution Parallel**, **Convolution Sequential**, **Residual Parallel**, and **Residual Sequential**.

Taking the bottleneck residual block of ResNet-50 [[19](#_bookmark40)] as an example, we demonstrate the proposed designs in Fig.

[3](#_bookmark9). As 1 × 1 convolution layer can only transfer channel- wise information, we thus design the adapting of functional convolutions, i.e., intermediate *K*×*K* convolutions, to keep locality sensitive. On the contrary, adapting the whole resid-

ual block considers the transferring of pre-trained knowl- edge carried by 1 × 1 convolutions. Intuitively, adapting

the learnable weight as **W**

*down*

*C*

∈ R *γ* for the

*in ×γ×K×K*

*γ*

the whole residual blocks has a larger capacity for mod-

ulating task-specific features than adapting only *K* × *K*

depth-wise convolution and **W***up*

∈ R*Cout× Cin ×*1*×*1 for

convolution but may introduce more parameters. Plugging

the point-wise convolution in Conv-Adapter, with the non- linearity denoted as *f* . We use a compression factor of *γ* to denote the down-sampling in the channel dimension, where *γ* is a hyper-parameter tuned for each task. Mathematically,

Conv-Adapter computes ∆**h** ∈ R*Cout×H×W* as:

∆**h** = **W***up* ⊗ *f* (**W***down*⊗ˆ**z**) *,* (2)

where ⊗ and ⊗ˆ denotes point-wise and depth-wise convolu- tion, respectively. To allow the modulation ∆**h** to be more flexibly composed into **h**, we set ***α*** in Eq. ([1](#_bookmark6)) as a learnable scaling vector in R*Cout* , which is initialized as ones. The ablation study on design choices is presented in Sec. [4.5](#_bookmark17).

## Adapting ConvNets with Conv-Adapter

After setting the architecture of Conv-Adapter, we dis- cuss the scheme to adapt a variety of ConvNets. Previous PET methods insert the adapting modules to Self-Attention blocks, Feed-Forward blocks, or both [[18](#_bookmark39)] of Transformers, which have a relatively unified architecture. In contrast, modern ConvNets usually stacks either residual blocks [[19](#_bookmark40), [51](#_bookmark70), [61](#_bookmark80)] or inverted residual blocks [[23](#_bookmark44), [36](#_bookmark56), [52](#_bookmark71), [53](#_bookmark72)],

Conv-Adapter stage-wisely is not considered as it is imprac- tical to make the receptive field of Conv-Adapter similar to the adapted stage with only two convolutions. It needs a more sophisticated design on not only the Conv-Adapter ar- chitecture but also the adaptation location [[59](#_bookmark78)], and we em- pirically find that stage-wise adaptation produces inferior performance and requires much more parameters. Conv- Adapter is flexible to be inserted into every residual block of the ConvNet backbone for transferability of features from different depths, as in [[39](#_bookmark58), [48](#_bookmark67)]. Other backbones such as ConvNext [[36](#_bookmark56)], and even Swin-Transformer [[34](#_bookmark55)] can be adapted following the same guideline (see experiments).

# Experiments

This section verifies the transferability and parameter effi- ciency of Conv-Adapter from various aspects, including im- age classification, few-shot classification, object detection, and semantic segmentation. Additionally, we provide an ab- lation study of Conv-Adapter for its design choices and an analysis of its performance.



*1 x 1 Conv.*

*x*

*K x K Conv. Conv-Adapter*

+

*1 x 1 Conv.*

+



*1 x 1 Conv.*

*K x K Conv.*

*h*

*Conv-Adapter*

+

*1 x 1 Conv.*

+



*1 x 1 Conv.*

*1 x 1 Conv.*

*x*

*K x K Conv. Conv-Adapter*

*K x K Conv.*

*1 x 1 Conv.*

*1 x 1 Conv.*

*h*

*Conv-Adapter*

+ +

Conv. Parallel

Conv. Sequential

Res. Parallel

Res. Sequential

Figure 3. Four adapting schemes of Conv-Adapter to ResNet50: Convolution Parallel, Convolutional Sequential, Residual Parallel, and Residual Sequential. The schemes differ regarding the position of of the modified representation and corresponding insertion form. Other networks can be adapted similarly following the illustration. Green modules are frozen during fine-tuning.

## Transferability of Conv-Adapter

### Setup

We first evaluate the transferability of Conv-Adapter on classification tasks. We experiment on two benchmarks: VTAB-1k [[60](#_bookmark79)] and FGVC. VTAB-1k includes 19 diverse visual classification tasks, which are grouped into three cat- egories: *Natural*, *Specialized*, and *Structured* based on the

Table 1. Performance of Conv-Adapter adapting schemes on ResNet-50 BiT-M. Each setting includes three runs and averaged top-1 accuracy (%) over datasets and the averaged total trainable parameters (M) over all datasets are reported. We compare pro- posed variants of Conv-Adapter (in gray) with full Fine-Tuning (FT), Linear Probing (LP), Bias Tuning (Bias), and Visual Prompt Tuning (VPT). We report the number of wins of (*·*) for each method compared to FT. **Bold** and underline refer to the top and second result separately.

domain of the images. Each task in VTAB-1k contains

1,000 training examples. FGVC consists of 4 *Fine-Grained Visual Classification* tasks: CUB-200-2011 [[57](#_bookmark76)], Stanford Dogs [[26](#_bookmark47)], Stanford Cars [[29](#_bookmark50)], and NABirds [[55](#_bookmark74)].

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tuning | # Param. | FGVC | VTAB-1k | | |
| Natural | Specialized | Structured |
| # Tasks | - | 4 | 7 | 4 | 8 |
| FT | 23.89 | 83.46 | 72.19 | **85.86** | **66.72** |
| LP | 0.37 | 75.44 (1) | 67.42 (4) | 81.42 (0) | 37.92 (0) |
| Bias | 0.41 | 64.98 (0) | 66.06 (4) | 80.34 (0) | 32.18 (0) |
| VPT | 0.42 | 74.79 (1) | 65.43 (2) | 80.35 (0) | 37.64 (0) |
| Conv. Par. | 0.85 | 83.77 (3) | **72.60 (5)** | 84.21 (1) | 56.70 (1) |
| Conv. Seq. | 0.87 | 79.68 (2) | 72.28 (4) | 83.85 (0) | 58.50 (1) |
| Res. Par. | 8.21 | **84.24 (3)** | 71.75 (4) | 84.70 (0) | 61.34 (1) |
| Res. Seq. | 3.53 | 83.45 (2) | 71.74 (4) | 84.84 (0) | 61.33 (2) |

For evaluation, we compare the 4 variants of Conv- Adapter with full fine-tuning (FT) and 3 baseline meth- ods: linear probing (LP), bias tuning (Bias) [[7](#_bookmark28)], and vi- sual prompt tuning (VPT) [[1](#_bookmark22), [25](#_bookmark46)]. We test each method on ResNet50 [[19](#_bookmark40), [27](#_bookmark48)] with ImageNet21k pre-training. To find the optimal hyper-parameters of Conv-Adapter (and base- line methods), we conduct a grid search of the learning rate, weight decay, and compression factor *γ* for each dataset

using the validation data split from training data for both

benchmarks. For VTAB-1k, we use the recommended op- timal data augmentations in [[60](#_bookmark79)], rather than solely Resize and Centre Crop as in [[1](#_bookmark22), [63](#_bookmark82)]. We find the recommended augmentations produces better results for full-tuning. For FGVC, we use RandomResized Crop with a minimum scale of 0.2 and Horizontal Flip [[50](#_bookmark69)] as augmentation. Mores de- tails of the hyper-parameters are shown in Appendix.

### Results and Discussion

Results are reported in Tab. [1](#_bookmark11). Conv-Adapter not only demonstrates significant improvements over the baseline methods, but also achieves the same level of performance

or even surpasses their fine-tuning counterparts on all do- mains evaluated, by introducing only around **3.5%** of full fine-tuning parameters for ResNet-50. Notably, there is a considerable performance gap, i.e., an improvement of **23.44%**, of Conv-Adapter over previous baseline methods on *Structured* datasets of VTAB-1k.

One can observe that the proposed four variants of Conv- Adapter all achieve comparable performance compared to full fine-tuning. Among the four variants, **Convolution Parallel** achieves the best trade-off between performance and parameter efficiency. On the evaluated classification tasks, inserting Conv-Adapter in parallel generally outper- forms inserting sequentially. In terms of the modified repre-

sentation, one can find that, on most of the datasets, adapt- ing only the *K* × *K* convolutions of ResNet-50 can achieve performance close to fine-tuning. However, on *Structured* datasets, adapting whole residual blocks is far better than adjusting only the middle convolutions with more parame-

ters, demonstrating the superior capacity of adjusting resid- ual blocks when there is a more significant domain gap.

## Universality of Conv-Adapter

### Setup

We evaluate the universality of Conv-Adapter on classi- fication tasks in this section, where Conv-Adapter is in- serted to various ConvNets architectures with different pre- training. We adopt the simple yet effective adapting scheme – Convolution Parallel, and mainly compare it with full fine-tuning. More specifically, we adopt ImageNet-21k pre- trained ResNet50 [[27](#_bookmark48)], ConvNext-B and ConvNext-L [[36](#_bookmark56)], and even Swin-B and Swin-L [[34](#_bookmark55)]. Apart from ImageNet- 21k, we evaluate ImageNet-1K, CLIP [[20](#_bookmark41)], and MoCov3

[[11](#_bookmark32)] pre-training. Similarly, we conduct a hyper-parameter search on the validation set, and report the accuracy on the test set of FGVC and VTAB-1k. Model details are shown in Appendix.

### Results and Discussion

We present the results in Tab. [2](#_bookmark13). On various ImageNet-21k pre-trained ConvNets, Conv-Adapter demonstrates its uni- versality with comparable performance to fine-tuning. For large models such as ConvNext-L and Swin-L, conducting traditional fine-tuning requires training nearly 196M param- eters, whereas Conv-Adapter improves the parameter effi- ciency with only 7.8% and 4.5% of the fine-tuning param- eters on ConvNext-L and Swin-L respectively. Although the transfer performance of Conv-Adapter on ImageNet-1k pre-trained models is more limited, compared to ImageNet- 21k pre-training, Conv-Adapter still demonstrates its supe- rior parameter efficiency and shows improvement over fine- tuning on several tasks. For the CLIP vision models, Conv- Adapter consistently outperforms fine-tuning on Structured tasks of VTAB-1k. We observe a performance gap of Conv- Adapter on MoCov3 pre-trained [[11](#_bookmark32)], and we argue this is possibly due to the difference in feature space of self- supervised and supervised models in CV [[25](#_bookmark46)].

## Few-Shot Classification

### Setup

PET methods usually present superior performance for tasks with low-data regimes [[18](#_bookmark39), [31](#_bookmark52)]. We thus evaluate Conv-Adapter on few-shot classification using ImageNet- 21k pre-trained ResNet50 Bit-M [[27](#_bookmark48)] and ConvNext-B [[36](#_bookmark56)]. We evaluate 5 FGVC datasets using 1, 2, 4, 8 shots for

Table 2. Comparing Conv-Adapter (CA) with full Fine-Tuning (FT) using various backbone architectures of different pre- training. Each setting includes three runs and averaged top-1 accu- racy (%) over datasets and the averaged total trainable parameters

(M) over all datasets are reported. We report the number of wins of (*·*) for each method in compared to FT. **Bold** indicates the best results.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Pre-train | Backbone | Tuning | # Param. | FGVC | Natural | VTAB-1k  Specialized | Structured |
| # Tasks |  |  |  | 4 | 7 | 4 | 8 |
|  | ResNet50 | FT | 23.89 | 83.46 | 72.19 | **85.86** | **66.72** |
|  | BiT-M |
| CA | 0.85 | **83.77 (3)** | **72.60 (5)** | 84.21 (1) | 56.70 (1) |
|  | ConvNext-B | FT | 87.75 | **89.48** | **81.59** | **87.32** | **65.77** |
| ImageNet | CA | 6.83 | 89.28 (1) | 80.62 (4) | 86.29 (0) | 64.88 (2) |
| ConvNext-L | FT | 196.50 | 90.64 | **82.25** | **87.94** | **67.65** |
| 21k |
| CA | 15.52 | **90.69 (3)** | 81.7 (2) | 86.85 (0) | 64.98 (3) |
|  | Swin-B | FT | 86.92 | **90.01** | 78.65 | **87.59** | **64.69** |
|  | CA | 4.98 | 88.55 (1) | **80.00 (4)** | 85.84 (0) | 62.57 (2) |
|  | Swin-L | FT | 195.27 | **91.04** | 80.64 | **87.85** | **66** |
|  | CA | 8.86 | 90.54 (2) | **81.39 (3)** | 86.29 (1) | 63.19 (2) |
|  | ResNet50 | FT | 23.87 | **85.84** | **67.15** | **83.53** | **53.32** |
| ImageNet | CA | 0.72 | 83.48 (0) | 64.20 (0) | 81.33 (1) | 52.74 (2) |
| ConvNext-B | FT | 87.75 | **88.95** | 74.51 | **85.33** | 61.34 |
| 1k |
| CA | 10.82 | 87.84 (1) | **74.72 (4)** | 84.29 (0) | **63.77 (2)** |
| CLIP | ResNet50 | FT | 38.50 | **81.38** | **58.53** | **80.8** | 57.18 |
| CA | 2.23 | 76.64 (0) | 56.33 (3) | 79.12 (0) | **58.96 (4)** |
| ResNet50x4 | FT | 87.17 | **84.23** | **65.71** | **82.22** | 58.84 |
| CA | 6.14 | 82.71 (0) | 62.54 (2) | 80.72 (1) | **59.10 (4)** |
| MoCov3 | ResNet50 | FT | 23.87 | **83.92** | **66.25** | **83.89** | **60.26** |
| CA | 0.89 | 79.69 (0) | 65.31 (3) | 81.59 (0) | 53.87 (1) |

each class following following previous studies [[25](#_bookmark46), [45](#_bookmark64), [63](#_bookmark82)] including Food101 [[4](#_bookmark25)], Oxford Flowers [[41](#_bookmark60)], Oxford Pets [[42](#_bookmark61)], Stanford Cars [[29](#_bookmark50)], and Aircraft [[38](#_bookmark57)]. Averaged top-1 accuracy is reported in Tab. [3](#_bookmark14). We search from the same range as before and adopt the same augmentations as for FGVC tasks. The detailed hyper-parameters and more re- sults for each dataset are in Appendix.

### Results and Discussion

Compared with Fine-tuning, Conv-Adapter boosts few-shot classifications with an average 3.39% margin over differ- ent shots using only around 5% trainable parameters. Es- pecially for 1/2-shot cases, Conv-Adapter shows supreme performance compared with Fine-tuning and VPT [[25](#_bookmark46)] (11.07% on 1-shot and 6.99% on 2-shot with larger archi- tecture ConvNext-B). Meanwhile, Conv-Adapter provides a better accuracy-efficiency trade-off than Visual Prompt Tuning on few-shot classifications. It surpasses VPT with an average margin of 1.35% with ResNet50 Bit-M and 3.69% with ConvNext-B. In the 8-shot case, VPT drops around 8% performance compared with Fine-tuning due to limited capacity, while Conv-Adapter can achieve compa- rable or better performance to Fine-tuning and maintain pa- rameter efficiency.

Table 3. Few-shot classification: the average Top-1 accuracy over 5 FGVC datasets, with 1, 2, 4, 8 shots. We compare Conv-Adapter (CA), Visual Prompt Tuning (VPT), and full Fine-Tuning (FT). **Bold** indicates the best results.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Backbone | Tuning | # Param | 1 | 2 | 4 | 8 |
| ResNet50 Bit-M | FT VPT | 23.72  0.24 | 29.30  32.56 | 38.96  42.18 | 50.09  52.21 | 61.27  59.37 |
| CA | 1.02 | **34.31** | **43.55** | **52.43** | **61.42** |
|  | FT | 87.68 | 36.34 | 48.83 | **63.69** | **76.91** |
| ConvNext-B | VPT | 0.13 | 42.25 | 51.85 | 62.89 | 69.04 |
|  | CA | 4.6 | **47.41** | **55.82** | 63.25 | 74.29 |

Table 4. Object detection & Semantic Segmentation results. We report the results of fine-tuning and Conv-Adapter with the Resid- ual Parallel scheme.

Object Detection with Faster-RCNN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Backbone | Tuning | # Param | AP | AP50 | AP75 |
| ResNet50 | FT | 41.53 | 38.1 | 59.7 | 41.5 |
| CA | 35.72 | **38.4** | **61.1** | **41.5** |
| ConvNeXt-B | FT | 67.09 | **45.2** | **67.2** | **49.9** |
| CA | 24.62 | 41.9 | 64.5 | 45.7 |

Semantic Segmentation with UPerNet

|  |  |  |  |
| --- | --- | --- | --- |
| Backbone | Tuning | # Param (M) | mIoU |
| ResNet50 | FT | 66.49 | 42.1 |
| CA | 45.65 | **43.0** |
| ConvNeXt-B | FT | 81.87 | **48.7** |
| CA | 39.40 | 46.9 |

## Object Detection and Semantic Segmentation

### Setup

Beyond image classification tasks, we also validate the gen- eralization of Conv-Adapter on dense prediction tasks, in- cluding object detection and semantic segmentation. We use ImageNet-21k pre-trained ResNet50 and ConvNeXt- S as backbones. For object detection, we implement Conv-Adapter with Faster-RCNN using the MMDetection

[[8](#_bookmark29)] framework compared with fine-tuning. We report the average precision (AP) results on the validation split of the MS-COCO dataset [[32](#_bookmark53)]. For semantic segmentation, we implement Conv-Adapter with UPerNet [[58](#_bookmark77)] using MM- Segmentation framework [[12](#_bookmark33)] and conduct experiments on the ADE20K dataset [[64](#_bookmark83)], with mIoU reported on the vali- dation split.

For object detection, we compare all four schemes of Conv-Adapter with the fine-tuning baseline. Specifically, we follow a standard 1x training schedule: all models are trained with a batch size of 16 and optimized by AdamW with an initial learning rate of 0.0002 for Faster RCNN and 0.0001 for RetinaNet, which are then dropped by a factor of 10 at the 8-th and 11-th epoch. The shorter side of the input image is resized to 800 while maintaining the original aspect ratio. For segmentation, we train all models for 80k

iterations with an random cropping augmentation of 512 ×

512 input resolution. For ConvNeXt models, we use a larger input resolution of 640 × 640 and train the models for 160k iterations. We apply AdamW optimizer with a polynomial learning rate decay schedule. More detailed training setting and hyper-parameters are shown in Appendix.

### Results and Discussion

The dense prediction results are summarized in Tab. [4](#_bookmark16). We observe a different effect of Conv-Adapter on two types of backbones. On ResNet50, Conv-Adapter surpasses fine- tuning with fewer trainable parameters (including the dense prediction heads) for object detection and semantic segmen- tation. On ConvNeXt-S, the performance is lower than their fine-tuning counterparts. We argue that the inferior per- formance of Conv-Adapter on ConvNeXt-S on dense pre- diction tasks is due to severely reduced model capacity as the number of trainable parameters is reduced by more than 50%. Nevertheless, they can still outperform the ResNet50 with fewer total parameters. This indicates there might be overfitting issues, and we encourage more future studies on this topic.

## Ablation Study

### Setup

We provide an ablation study on the design choices of Conv-Adapter, where we explore different architectures and adapting schemes. In this section, we mainly report the Top- 1 accuracy on the validation set of VTAB-1k.

### Architecture and Adapting Schemes

We first compare the performance of Conv-Adapter using depth-wise separable, regular, and 1×1 convolutions (linear layers). As shown in Tab. [5](#_bookmark18), depth-wise separable convolu- tion introduces the minimal parameter budget while achiev-

ing the best results. Apart from 4 adapting variants pro- posed in this work, we also explore other design choices used in previous works. We experiment on spatial down- sampling of feature maps [[7](#_bookmark28)]. Compared to channel down- sampling with a bottleneck in Conv-Adapter, spatial down- sampling introduces nearly 27 times of parameters with in- ferior accuracy. We also validate the adapting scheme of

applying 1 × 1 convolution to all convolutional layers [[48](#_bookmark67)], which introduces nearly 16 times of parameters to Conv-

Adapter with -12.27% accuracy gain. Finally, we evalu- ate the adapting scheme that inserts Conv-Adapter stage- wisely, which is less effective in both parameter size and performance than the proposed schemes.

Table 5. Ablation study on more adapting scheme and more archi-

ResNet50 ConvNext-B

tectures of Conv-Adapter. The different schemes and architectures 75

mainly come from previous works. The proposed adaptation and 74

Accuracy

architecture achieve the best results. 73

72

Adapting Scheme Down-sample # Convs Type of Conv. # Param VTAB-1k 71

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *K* × *K* Conv. Par. | Channel | 2 | Depth-wise | 0.67 | **71.03** |
| *K* × *K* Conv. Par.  *K* × *K* Conv. Par. | Channel Channel | 2  2 | Regular Linear | 5.66  1.22 | 70.52  68.32 |

70

69

*K* × *K* Conv. Par. Spatial

All Conv. Par -

Stage Par. Channel

71

2 Depth-wise

1. Linear
2. Depth-wise

71

18.45

10.74

1.90

68.54

58.75

65.06

1 3 5 7

Kernel Size

Figure 5. Sensitivity to kernel size of depth-wise convolution in Conv-Adapter, for both ResNet50 and ConvNext-B.

70

Accuracy

69

68

67

0.0 1.0 2.5 5.0 10.0

Init. of 

70

69

Accuracy

68

67

1 2 4 8 16 32

1

0

Relative Acc Gain(%)

-1

-2

-3

-4

1.0

0.8

ResNet50 ConvNext-B CLIP

ResNet50 ConvNext-B CLIP

Relative CKA Sim.

0.6

0.4

0.2

0.0

Figure 4. Sensitivity to hyper-parameters of initialization of learn- able scaling vector ***α*** and compression factor *γ*.

Cars Cub200 Dogs

Cars Cub200 Dogs

### Sensitivity to *γ* and initialization of *α*

We explicitly study the sensitivity of the transfer perfor- mance to the initialization of the learnable scaling vector ***α*** and compression factor *γ* in Conv-Adapter, as shown in Fig.

[4](#_bookmark21). When initializing ***α*** as ones, Conv-Adapter achieves the best performance on the validation set of VTAB-1k. Com- pared to ***α***, Conv-Adapter is more robust to the compression factor *γ*, achieving similar performance with the compres- sion factor of 1, 2, and 4. Setting *γ* with a larger value results in inferior performance with a more limited capacity of Conv-Adapter.

### Kernel size in Conv-Adapter

We show the performance of Conv-Adapter on VTAB-1k validation set in Fig [5](#_bookmark20), of using different kernel size for the depth-wise convolution to verify our argument of the *loss of locality*. One can observe that, for both ResNet50 and ConvNext-B, using smaller kernel size results in infe- rior performance. When setting the kernel size larger to that of the residual blocks, i.e., 5 and 7 for ResNet50, the perfor- mance is further boosted, with more parameters introduced.

### CKA Similarity of Conv-Adapter

We observe from Tab. [1](#_bookmark11) and Tab. [2](#_bookmark13) that, on datasets with large domain shifts, Conv-Adapter (and baseline methods) may fail to generalize well. To investigate the reason, we compute the CKA similarity [[28](#_bookmark49), [47](#_bookmark66)] between weights of

Figure 6. CKA similarity and accuracy gap between Conv-Adapter and fully fine-tuning for FGVC datasets.

convolutional filters for the pre-trained and fine-tuned back- bone. The lower the CKA similarity, the larger capacity is required for good transfer performance. We plot the CKA similarity and the relative accuracy gain of Conv-Adapter to fine-tuning in Fig. [7](#_bookmark0), where the same trends over datasets exhibit for different architectures. When fully fine-tuning only leads to small changes in filter weights (larger CKA similarities), Conv-Adapter is more likely to surpass the performance of fully fine-tuning. More detail on CKA sim- ilarity comparison is in Appendix.

# Conclusions

In this work, we propose Conv-Adapter, a parameter efficient tuning module for ConvNets. Conv-Adapter is light-weight, domain-transferable, and model-agnostic. Extensive experiments on classification and dense prediction tasks show it can achieve performance comparable to full fine-tuning with much fewer pa- rameters. We find Conv-Adapter might fail on tasks with large domain shifts and subject to feature qual- ity determined by pre-training. Future work includes more exploration of Conv-Adapter on domain robust- ness and dense predictions and NAS for Conv-Adapter.

# 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. [2](#_bookmark4), [5](#_bookmark10)

1. Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bit- fit: Simple parameter-efficient fine-tuning for transformer- based masked language-models. *Proceedings of the 60th Annual Meeting of the Association for Computational Lin-* *guistics (Volume 2: Short Papers)*, 2022. [1](#_bookmark1), [2](#_bookmark4)
2. Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Alt- man, Simran Arora, et al. On the opportunities and risks of foundation models, 2021. [1](#_bookmark1)
3. Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. [6](#_bookmark12)
4. Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Sub- biah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakan- tan, Pranav Shyam, Girish Sastry, Amanda Askell, Sand- hini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Rad- ford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. [1](#_bookmark1), [2](#_bookmark4)
5. Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *In- ternational Conference on Learning Representations*, 2019. [2](#_bookmark4), [3](#_bookmark5)
6. Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. Tinytl: Reduce memory, not parameters for efficient on-device learning. In *Advances in Neural Information Processing Sys- tems*, pages 11285–11297. Curran Associates, Inc., 2020. [1](#_bookmark1),

[4](#_bookmark8), [5](#_bookmark10), [7](#_bookmark15)

1. Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tian- heng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint* *arXiv:1906.07155*, 2019. [7](#_bookmark15)
2. Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for seman- tic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. [3](#_bookmark5)
3. Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapt- ing vision transformers for scalable visual recognition, 2022. [2](#_bookmark4)
4. Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv* *preprint arXiv:2104.02057*, 2021. [6](#_bookmark12)
5. MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. [https : / / github . com / open -](https://github.com/open-mmlab/mmsegmentation) [mmlab/mmsegmentation](https://github.com/open-mmlab/mmsegmentation), 2020. [7](#_bookmark15)
6. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina

Toutanova. Bert: Pre-training of deep bidirectional trans- formers for language understanding, 2018. [1](#_bookmark1), [2](#_bookmark4)

1. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Syl- vain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. [1](#_bookmark1)
2. Gamaleldin F. Elsayed, Ian Goodfellow, and Jascha Sohl- Dickstein. Adversarial reprogramming of neural networks, 2018. [2](#_bookmark4)
3. William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with sim- ple and efficient sparsity, 2021. [1](#_bookmark1), [2](#_bookmark4)
4. Yunhui Guo, Yandong Li, Liqiang Wang, and Tajana Rosing. Depthwise convolution is all you need for learning multiple visual domains. *Proceedings of the AAAI Conference on Ar-* *tificial Intelligence*, 33:8368–8375, 2019. [3](#_bookmark5), [4](#_bookmark8)
5. Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg- Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Con-* *ference on Learning Representations*, 2022. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5), [4](#_bookmark8), [6](#_bookmark12)
6. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition* *(CVPR)*, 2016. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5), [4](#_bookmark8), [5](#_bookmark10)
7. Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual rep- resentation learning. *2020 IEEE/CVF Conference on Com-* *puter Vision and Pattern Recognition (CVPR)*, 2020. [6](#_bookmark12), [2](#_bookmark4)
8. 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 *Proceedings of the 36th International* *Conference on Machine Learning*, 2019. [2](#_bookmark4), [3](#_bookmark5)
9. 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 *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. [1](#_bookmark1)
10. Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco An- dreetto, and Hartwig Adam. Mobilenets: Efficient convolu- tional neural networks for mobile vision applications, 2017. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5), [4](#_bookmark8)
11. Edward Hu, Yelong Shen, Phil Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5)
12. Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Vi- sual prompt tuning. In *European Conference on Computer* *Vision (ECCV)*, 2022. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5), [5](#_bookmark10), [6](#_bookmark12)
13. 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*, Colorado Springs, CO, 2011. [5](#_bookmark10)
14. Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning, 2019. [2](#_bookmark4), [5](#_bookmark10), [6](#_bookmark12), [3](#_bookmark5), [4](#_bookmark8)
15. Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network represen- tations revisited. In *International Conference on Machine* *Learning*, pages 3519–3529. PMLR, 2019. [8](#_bookmark19), [3](#_bookmark5)
16. Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and* *Recognition (3dRR-13)*, Sydney, Australia, 2013. [5](#_bookmark10), [6](#_bookmark12)
17. Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceed- ings of the 2021 Conference on Empirical Methods in Nat- ural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5)
18. Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation, 2021. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5), [6](#_bookmark12)
19. Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dolla´r. Mi- crosoft coco: Common objects in context, 2014. cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list. [7](#_bookmark15)
20. Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hi- roaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natu- ral language processing, 2021. [2](#_bookmark4)
21. 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 (ICCV)*, 2021. [1](#_bookmark1), [4](#_bookmark8), [6](#_bookmark12), [2](#_bookmark4)
22. Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution. In *International Conference on* *Computer Vision and Pattern Recognition (CVPR)*, 2022.
23. Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feicht- enhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Com- puter Vision and Pattern Recognition (CVPR)*, 2022. [1](#_bookmark1), [2](#_bookmark4), [4](#_bookmark8),

[6](#_bookmark12), [3](#_bookmark5)

1. Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017. [2](#_bookmark4)
2. S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical re- port, 2013. [6](#_bookmark12)
3. Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggy- back: Adapting a single network to multiple tasks by learn- ing to mask weights, 2018. [2](#_bookmark4), [4](#_bookmark8)
4. Pramod Kaushik Mudrakarta, Mark Sandler, Andrey Zh- moginov, and Andrew Howard. K for the price of 1: Parameter-efficient multi-task and transfer learning, 2018. [2](#_bookmark4)
5. M-E Nilsback and Andrew Zisserman. A visual vocabu- lary for flower classification. In *2006 IEEE Computer Soci- ety Conference on Computer Vision and Pattern Recognition* *(CVPR’06)*, pages 1447–1454. IEEE, 2006. [6](#_bookmark12)
6. Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and

C. V. Jawahar. Cats and dogs. In *IEEE Conference on Com-* *puter Vision and Pattern Recognition*, 2012. [6](#_bookmark12)

1. Jonas Pfeiffer, Andreas Ru¨ckle´, Clifton Poth, Aishwarya Ka- math, Ivan Vulic´, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online, 2020. Association for Computational Linguistics. [2](#_bookmark4)
2. Jonas Pfeiffer, Aishwarya Kamath, Andreas Ru¨ckle´, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non- destructive task composition for transfer learning. *Proceed- ings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021. [2](#_bookmark4)
3. 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, 2021. [1](#_bookmark1), [2](#_bookmark4), [6](#_bookmark12)
4. Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollar. Designing network design spaces. *2020 IEEE/CVF Conference on Computer Vision and* *Pattern Recognition (CVPR)*, 2020. [1](#_bookmark1)
5. Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision trans- formers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128, 2021. [8](#_bookmark19), [3](#_bookmark5)
6. Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural net- works. In *CVPR*, 2018. [1](#_bookmark1), [2](#_bookmark4), [3](#_bookmark5), [4](#_bookmark8), [7](#_bookmark15)
7. Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):640–651, 2017. [3](#_bookmark5)
8. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision* *and Pattern Recognition (CVPR)*, 2015. [5](#_bookmark10), [1](#_bookmark1)
9. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. [4](#_bookmark8)
10. Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2019. [1](#_bookmark1), [4](#_bookmark8)
11. Mingxing Tan and Quoc V. Le. Efficientnetv2: Smaller mod- els and faster training, 2021. [4](#_bookmark8)
12. Sebastian Thrun. Lifelong learning algorithms. In *Learning* *to learn*, pages 181–209. Springer, 1998. [1](#_bookmark1)
13. Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Be- longie. 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](#_bookmark10)
14. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszko- reit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. [2](#_bookmark4)
15. C. Wah, S. Branson, P. Welinder, P. Perona, and S. Be- longie. The caltech-ucsd birds200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technol- ogy, 2011. [5](#_bookmark10)
16. Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understand- ing. In *Proceedings of the European Conference on Com-* *puter Vision (ECCV)*, pages 418–434, 2018. [7](#_bookmark15)
17. Li Yang, Adnan Siraj Rakin, and Deliang Fan. Rep-net: Effi- cient on-device learning via feature reprogramming. In *Pro- ceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12277–12286, 2022. [3](#_bookmark5), [4](#_bookmark8)
18. Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djo- longa, Andre Susano Pinto, Maxim Neumann, Alexey Doso- vitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark, 2019. [2](#_bookmark4), [5](#_bookmark10), [1](#_bookmark1)
19. Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller,

R. Manmatha, Mu Li, and Alexander Smola. Resnest: Split- attention networks, 2020. [4](#_bookmark8)

1. Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. Differen- tiable prompt makes pre-trained language models better few- shot learners, 2021. [1](#_bookmark1)
2. Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search, 2022. [1](#_bookmark1), [5](#_bookmark10), [6](#_bookmark12)
3. Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [7](#_bookmark15)