Less Is More: Rethinking Parameter-Efficient Fine-Tuning from a Subtractive Perspective

Tianqi Jiang[1], Liu Yang[1*], Xi-Le Zhao[3], Zixuan Qin[2], Qinghua Hu[1]
[1]School of Artificial Intelligence, Tianjin University
[2]School of Computer Science and Technology, Tianjin University
[3]University of Electronic Science and Technology of China, Chengdu, China
jtq_3019@tju.edu.cn, yangliuyl@tju.edu.cn, xlzhao122003@163.com, qinzixuan1958@tju.edu.cn, huqinghua@tju.edu.cn

Abstract

Currently, pretrained models are rapidly scaling in size, which substantially increases the cost of fine-tuning them for downstream tasks. To address this challenge, parameter-efficient fine-tuning (PEFT) methods have been developed to optimize a minimal set of parameters for adaptation. While current PEFT approaches predominantly employ an "additive" strategy, introducing learnable modules into inputs or architectures, neglect the inherent knowledge embedded within pretrained models, which may be redundant or even conflict with downstream tasks. This limitation leads to increased inference latency and suboptimal transfer performance, particularly in scenarios with significant domain gaps. In this paper, we propose a Subtractive Fine-tuning Paradigm(SFP), which converts multiple redundant operations within the original module into a linear transformation to enhance inference speed and model performance. Specifically, we introduce a compact filter block to replace specific module with interference and redundancy in the original structure to reduce model conflicts. By using a pseudo inverse matrix to construct filter block, ensuring that it can inherit the knowledge of the replacement module, and then freezing the rest of the model, only fine-tuning the filter block is performed to eliminate interference and redundant knowledge, thereby enhancing the model's adaptability to downstream tasks. Experimental results demonstrate that our SFP outperforms existing PEFT methods in accuracy while decreasing the overall model parameters by 12%. Compared to full fine-tuning, the accuracy has increased by 8.47%(74.04% vs. 65.57%, VTAB).

Introduction

Pre-training followed by fine-tuning has become a cornerstone framework in artificial intelligence. However, full-parameter fine-tuning methods present several challenges. In particular, fine-tuning high-parameter models demands substantial computational resources (Fu, Zhu, and Wu 2024; Xin et al. 2024c), and performance can degrade when there is a distribution shift between the pre-training dataset and downstream tasks (Xin et al. 2024b). In response to these challenges, PEFT (Sung, Nair, and Raffel 2021) has emerged as a promising alternative. PEFT facilitates effective knowledge transfer from pre-trained models to downstream tasks by freezing most model parameters and selectively updating only minimal adaptation modules. This strategic optimization significantly reduces training costs while maintaining competitive accuracy(Fig. 1).

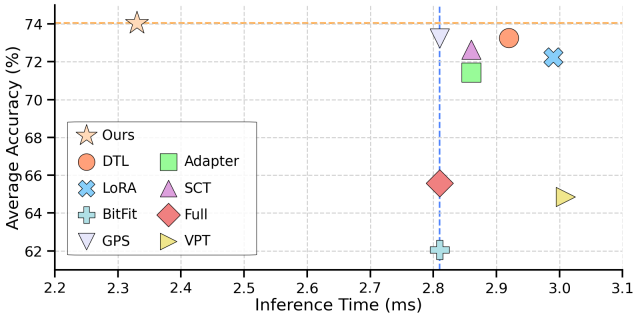

Figure 1: Average accuracy vs. inference time on VTAB-1K (Zhai et al. 2019). Our method outperforms existing PEFT methods in accuracy, speed, and parameter efficiency. Vertical dashed lines indicate the original model's inference time.

As shown in Fig. 2 (a), current methods predominantly adopt an "additive" approach to adapt pre-trained models for downstream tasks by introducing supplementary trainable components: Adapter tuning methods (Houlsby et al. 2019; Xin et al. 2024a) implement task adaptation through inserting adapter within each transformer layer. Prompt tuning methods (Jia et al. 2022; Chen et al. 2023) integrate learnable tokens into the frozen models' input representations, while side tuning (Zhang et al. 2020) utilizes an auxiliary network that operates concurrently with pre-trained models, thereby enhancing performance through parallel integration.

Although these strategies have successfully enabled parameter-efficient fine-tuning, they largely overlook a fundamental issue: pre-trained knowledge repositories inherently contain a considerable amount of redundant and potentially detrimental knowledge relative to the target downstream tasks. Results in suboptimal performance when downstream and pre-training data diverge significantly.The observation raises a critical question: Can we adopt a "subtractive" approach by eliminating redundant and interfering knowledge to achieve improved performance while simultaneously simplifying the model, rather than relying solely on

*Corresponding author.

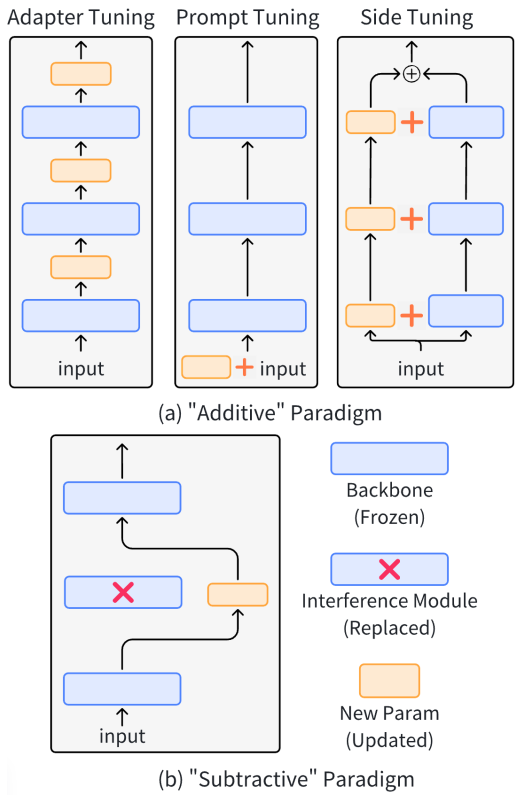

Figure 2: (a) The current mainstream methods based on the "additive" paradigm: they overlook that for a given task, some knowledge within a pre-trained model is beneficial, whereas other knowledge can be redundant or even detrimental. These methods indiscriminately incorporate trainable modules into every component. (b) Our "subtractive" paradigm: To streamline the processing pipeline, we substitute the original module which contains substantial interference knowledge with a compact filter block. This approach preserves essential knowledge while eliminating interfering elements. In addition to outperforming other PEFT methods, it achieves a direct 12% reduction in total parameters.

incremental architectural modifications?

To address this challenge, as shown in Fig. 2 (b). We propose a subtraction paradigm that replaces the original layers with compact, parameter-efficient filter block modules that selectively retain only task-critical knowledge, thereby mitigating extraneous knowledge interference. Through a dedicated construction scheme, the filter block effectively inherits the knowledge from its corresponding layer. Subsequently, we replace the original layer with this filter block while keeping the remaining model components frozen. The training process exclusively optimizes the filter block to systematically eliminate task-irrelevant interference while preserving essential knowledge. As shown in Fig. 1, this methodology not only significantly reduces the total number of model parameters but also enhances task-specific knowledge purity.

In a nutshell, we summarize our contributions as follows:

- We pioneer a subtractive paradigm for PEFT. Specifically, we replace redundant or detrimental modules within the original structure with simple, lightweight filter blocks, and only train these filter blocks while freezing the remaining network, thereby eliminating redundant or interfering knowledge.
- We establish practical guidelines for designing and training SFP, designed its construction strategy and training methods, laying the foundation for future research.
- Extensive experiments on the Visual Task Adaptation Benchmark across all 19 datasets demonstrate that SFP outperforms popular PEFT approaches while decreasing the overall model parameters by 12%.

Related Work

Parameter-efficient Fine-tuning (PEFT)

Current addition-based PEFT methods can be broadly categorized into four classes: adapter tuning, prompt tuning, side tuning, and reparameter tuning. Adapter tuning (Houlsby et al. 2019; Sharma et al. 2023; Yang et al. 2023) adapts pre-trained models by inserting adapter into each transformer layer. ReAdapter (Luo et al. 2023) removes nonlinear components from conventional adapters and proposes RepAdapter, integrating lightweight networks into pre-trained architectures. Prompt tuning (Jia et al. 2022; Huang et al. 2023; Tsai, Mao, and Yang 2023) operates under full model freezing by introducing learnable tokens into input image representations to align features with the model's pre-trained domain. LION (Wang et al. 2024) exemplifies this paradigm, by inserting two equilibrium layers at both ends of frozen backbone networks. Side tuning (Chen et al. 2022; Lin et al. 2023) employs a smaller and separate network that operates in parallel with the pretrained models. DTL (Fu, Zhu, and Wu 2024) utilizes pre-trained models and separate networks to jointly process representations. Reparameter tuning (Hu et al. 2022; Basu et al. 2024) uses low-rank matrices to adjust the overall model. However, existing methods critically overlook a fundamental limitation: the inherent presence of substantial interfering knowledge within pre-trained large-scale models that adversely impacts downstream task performance.

Model Stitching

The objective of model stitching is to connect the lower layers of one network to the upper layers of another through a stitching layer, originally proposed to measure similarity in internal representations of deep neural networks (Hernandez et al. 2023; Balogh and Jelasity 2023). Previous works (Bansal, Nakkiran, and Barak 2021; Csiszárik et al. 2021) demonstrate that differently initialized networks can be effectively stitched without significant performance degradation. Based on this observation, SN-Net (Pan, Cai, and Zhuang 2023) pioneered stitching-layer connections between differently sized models within the same model family to obtain scalable architectures, while SN-Netv2 (Pan et al. 2024) further explored diverse stitching pathways. Throughout these developments, the stitching layer remains

an auxiliary structure enabling cross-architectural integration of neural networks. While (Balogh and Jelasity 2025) argues that model stitching cannot serve as a valid measure of functional similarity, as the stitching layer generates out-of-distribution (OOD) representations compared to the original architecture. This observation also implies that the potential of stitching layers may have been underestimated. Inspired by this, we implement direct substitution of layers with stitching layers to unlock their latent potential.

Method

In this section, we first introduce the preliminary of component substitution, then we describe the details of our proposed subtractive paradigm.

Preliminaries of Component Substitution

Consider a pre-trained model where the i-th layer implements a function f_i. The full model $f\colon \mathcal{X} \to \mathcal{Y}$ maps input $\mathbf{X} \in \mathcal{X}$ to output $\mathcal{Y}$. For a L-layer feed-forward network, this is expressed as function composition:

$$f(\mathbf{X}) = f_L \circ f_{L-1} \circ \cdots \circ f_1(\mathbf{X}), \tag{1}$$

where $\circ$ denotes function composition. The core of component substitution involves substituting the l-th layer in the original model with a lightweight filter block $\mathcal{F}$, which can be formulated as:

$$f_{\text{sub}}(\mathbf{X}) = f_L \circ \cdots \circ f_{l+1} \circ \mathcal{F} \circ f_{l-1} \circ \cdots \circ f_1(\mathbf{X}), \tag{2}$$

where $\mathcal{F}\colon \mathcal{A}_{l-1} \to \mathcal{A}_l$ maintains identical input/output dimensions as the original f_l. Here $\mathcal{A}_l$ denotes the activation space of layer l, with $\mathcal{A}_0 = \mathcal{X}$ and $\mathcal{A}_L \subseteq \mathcal{Y}$.

By setting replacement positions and numbers, it is possible to effectively and flexibly handle modules with redundant or interfering knowledge.

Subtractive Fine-tuning Paradigm

Based on the framework of component substitution mentioned above, we can replace designated modules with compact filter blocks featuring minimal parameters to achieve architectural simplification. These filter blocks clear interference knowledge based on an "inheritance-filtration" paradigm, where they first assimilate the original knowledge and subsequently purge interference knowledge, thereby preserving essential knowledge for enhanced task adaptation.

The subsequent section provides a detailed description of the construction process of filter block and the process of clearing interference knowledge, while also offering strategic guidance for selecting the optimal replacement location.

Construction of Filter Block For the filter block, we aim to design a mini-module characterized by a small number of parameters and limited computational capacity, allowing it to retain only the key knowledge relevant to the current task. Prior research on representation similarity has concentrated on utilizing underpowered alignment modules to evaluate functional correspondence. SN-Net preserves this approach by incorporating a fully connected (FC) layer as a cross-model adapter, which projects representations from model A into the feature space of model B for model stitching. Although such FC-based alignment has demonstrated empirical efficacy, recent findings (Balogh and Jelasity 2025) indicate that it tends to produce out-of-distribution representations. This suggests the expressive power of fully connected layers may have been underestimated.

Consequently we employ FC layers as filter blocks, strategically replacing the original layers at designated positions. This substitution mechanism capitalizes on the FC layer's inherent capacity for linear transformation and feature filtering while maintaining minimal parameter complexity.

In order to ensure the smooth progress of component substitution, the filter block module needs to inherit the original knowledge of the replaced layer. Considering that unlike most works (Bao et al. 2021; Liu et al. 2021) where networks are trained from scratch, our approach is built on pre-trained models. In this case, the model has already learned a good and relatively fixed representation, which allows us to abstract several operations in the original layer into a single overall operation.

Given that we plan to replace the l-th layer of the original model with filter block. For the l-th layer, let $\mathbf{X}_{l-1} \in R^{N\times D}$ denote the input tensor, where N is the sequence length and D is the hidden dimension. The l-th layer processes it to obtain the output $\mathbf{X}_l \in R^{N\times D}$ through $\mathbf{X}_l = f_l(\mathbf{X}_{l-1})$. Given the redundancy and potential interference inherent in many operations within the modules replaced by the filter block, we avoid preserving all underlying processes individually. Thus, we try to approximate this complex functionality with a linear transformation:

$$\min_{\mathbf{W}_l} \|\mathbf{X}_{l-1}\mathbf{W}_l - \mathbf{X}_l\|_F^2, \tag{3}$$

where $\mathbf{W}_l \in R^{D\times D}$ represents the effective transformation matrix. This formulation enables filter block module initialization through pseudoinverse computation:

$$\mathbf{W}_l = \mathbf{X}_{l-1}^{\dagger}\mathbf{X}_l, \tag{4}$$

where $\mathbf{X}_{l-1}^{\dagger}$ denotes the Moore-Penrose pseudoinverse of $\mathbf{X}_{l-1}$. Based on the above calculation, we use $\mathbf{W}_l$ as the initialization for the filter block. This allows the filter block to inherit the knowledge of the original layer. Although this operation is simple, it has been proven to be reliable in the experiments. Considering that many of the operations in the original layer we replaced are either redundant or incorrect, the approximation described above is considered acceptable.

Efficient Training of SFP The design of the filter block has significantly streamlined the internal operations of the replaced layers. However, following the above construction, the current filter block still contains a considerable amount of redundant or interfering knowledge. It is essential to eliminate these extraneous elements in order to retain only the knowledge that is beneficial for the task.

Assuming that the l-th layer of the current model contains a significant amount of redundant and interfering knowledge, we propose directly replacing this layer with the filter block constructed as described above, thereby obtaining:

$$f_{\text{sub}}(\mathbf{X}) = f_L \circ \cdots \circ \mathcal{F}_{\text{init}} \circ \cdots \circ f_1(\mathbf{X}), \tag{5}$$

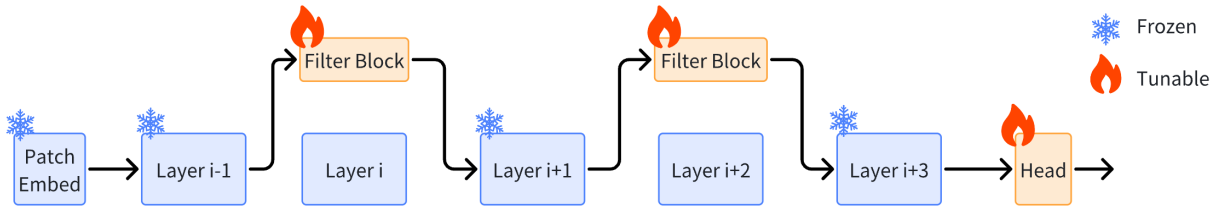

Figure 3: Illustration of the computation paths for dual-layer substitutions, our method demonstrating how the subtractive design directly mitigates the propagation of interference knowledge through architectural simplification.

where input $\mathbf{X} \in \mathcal{X}$. Since $\mathcal{F}_{\text{init}}$ encodes both useful and irrelevant knowledge for the target task, our next essential step is to filter out redundant and interfering components to retain only the most relevant knowledge.

Considering that interference knowledge within the module is not directly observable, and eliminating interfering information serves the same optimization objective as enhancing the alignment between the model's predictions and the target label space. We approximate the above optimization by minimizing the empirical task loss on the labeled dataset $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{Y}_i)\}$. The loss is defined as:

$$\mathcal{L}_{\text{task}} = \mathcal{L}\left(f_{\text{sub}}(\mathbf{X}; \theta_{\mathcal{F}}), \mathbf{Y}\right), \tag{6}$$

where $\mathcal{L}(\cdot, \cdot)$ is a standard classification loss function (e.g., cross-entropy). Crucially, we freeze all other layers of the model and only update the parameters $\theta_{\mathcal{F}}$ of the filter block:

$$\min_{\theta_{\mathcal{F}}} \mathcal{L}_{\text{task}} \quad \text{subject to} \quad \frac{\partial \mathcal{L}_{\text{task}}}{\partial \theta_i} = 0 \quad \forall \theta_i \notin \theta_{\mathcal{F}}, \tag{7}$$

which ensures that only the filter block is adapted, thus focusing the optimization solely on suppressing interference and enhancing task-relevant knowledge in the filter block.

We summarize the above process into the following three steps: (1) Filter block's construction: using a lightweight FC layer initialized with a pseudo-inverse matrix to fully inherit the original knowledge; (2) Component substitution: replacing the target layer with the filter block while freezing all other parameters; and (3) Selective training: finetuning solely the filter block module under strict low-parameterization constraints.

While the structural substitution with the filter block induces systematic shifts in activation distributions, thereby constraining operational efficacy, we address this geometric mismatch by selectively unfreezing the Layer Normalization (LN) parameters across layers. This targeted adaptation facilitates dynamic recalibration of feature statistics, effectively aligning intermediate representations to optimize filter block functionality. Notably, this strategic LN tuning introduces only 0.038M additional trainable parameters while preserving baseline computational complexity.

Filter Block's Optimal Replacing Positions When considering the optimal replacement location for filter block, our objective is to preserve the crucial components of the pre-trained model relevant to the current task while substituting the less essential or interfering parts. This necessitates an accurate evaluation of the importance of the model parameters. SNIP (Lee, Ajanthan, and Torr 2018) quantifies the importance of network parameters by computing the absolute product of the weight gradient and the corresponding parameters during a single backpropagation pass. This approach effectively identifies task-relevant parameters without necessitating full-scale training and incurs minimal computational overhead, which can be formulated as:

$$\mathcal{S}_p(\theta) = \left| \frac{\partial \mathcal{L}}{\partial \theta} \odot \theta \right|, \tag{8}$$

where $\mathcal{L}$ is the loss function of a network with parameters θ, S_p is the per-parameter saliency and $\odot$ is the Hadamard product. When performing replacement, substituting different positions generates multiple candidate models. For each model, we extend these saliency metrics to evaluate an entire network by summing the contributions of all N parameters in the model:

$$\mathcal{S} = \sum_{i}^{N} \mathcal{S}_p(\theta)_i. \tag{9}$$

During training, all candidate configurations receive equal resource allocation, and at each epoch, the lowest scoring options based on moving average SNIP evaluations are pruned. When the n-th option is sampled, its importance score $\mathcal{S}_n$ is obtained through Eq (9). To obtain robust importance scores for sampling, we accumulate scores with moving average during training:

$$q_n^t = \alpha q_n^{t-1} + (1 - \alpha)\mathcal{S}_n, \tag{10}$$

where q_n^t, q_n^{t-1} are the importance score at the t-th and $t-1$-th iteration respectively, and $\alpha \in [0, 1)$ is the momentum coefficient. In this way, we can get stable importance scores. This approach efficiently navigates the architectural search space while ensuring robust performance estimations.

Here we introduce two architectural substitution schemes as shown in Fig. 2 : (1) Single-layer substitution, wherein a preselected layer is replaced with a filter block module while preserving all other components, and (2) Dual-layer substitution, in which two distinct layers are strategically replaced with two filter blocks. For models comprising L layers, we employ the above method to evaluate the replacement potential of each layer. When replacing multiple layers, we directly leverage pre-computed single-layer prediction scores to determine replacement positions—a computationally zero-cost approach. Thus, the proposed prediction method remains highly efficient.

Method	MEAN	Param.	Natural							Specialized				Structured							
			CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	PCam	EuroSAT	Resisc45	Retinopathy	Clevr/Count	Clevr/Dist	DMLab	KITTI/dist	dSprites/loc	dSprites/ori	SmallNORB/azi	SmallNORB/ele
Traditional:																					
Full	65.57	0	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
Linear	52.94	0	63.4	85.0	63.2	97.0	86.3	36.6	51.0	78.5	87.5	68.6	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2
PEFT:																					
Adapter	71.44	↑0.19%	69.2	90.1	68.0	98.8	89.9	82.8	54.3	84.0	94.9	81.9	75.5	80.9	65.3	48.6	78.3	74.8	48.5	29.9	41.6
VPT-Shallow	64.85	↑0.09%	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
VPT-Deep	69.43	↑0.65%	**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
BitFit	62.05	0	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
LoRA	72.25	↑0.34%	67.1	91.4	69.4	98.8	90.4	85.3	54.0	84.9	95.3	84.4	73.6	**82.9**	**69.2**	49.8	78.5	75.7	47.1	31.0	44.0
SCT	72.64	↑0.13%	74.0	92.7	71.2	**99.2**	90.7	91.2	53.9	83.1	96.1	85.0	75.8	80.1	65.3	49.6	79.2	71.2	48.2	32.8	40.9
GPS	73.24	0	74.0	92.0	70.9	99.1	**91.7**	85.5	**56.6**	86.6	95.4	86.8	75.2	79.1	62.9	51.5	78.1	83.6	52.2	29.7	40.7
DTL	73.26	↑0.05%	67.7	**94.4**	71.1	99.1	91.5	82.2	55.4	84.0	**96.4**	85.3	73.6	82.8	62.2	48.6	79.3	**88.7**	**52.8**	31.5	**45.3**
Ours	**74.04**	**↓12.0%**	70.5	91.6	**73.5**	**99.2**	91.2	**91.3**	51.9	**86.8**	96.3	**88.3**	**76.3**	80.9	68.5	**53.4**	**83.0**	82.3	49.8	31.7	40.2

Table 1: Performance comparisons on VTAB-1k with ViT-B/16 models pre-trained on ImageNet-21K.

Experiments

Experimental Setting

Datasets Following VPT (Jia et al. 2022), we evaluate our method on the VTAB-1k(Zhai et al. 2019):Visual Task Adaptation Benchmark comprises 19 diverse visual classification datasets, which are organized into three domains: (1) Natural - datasets that contain natural images captured with standard cameras; (2) Specialized - datasets that contain images captured via specialized equipment, such as medical, and satellite images; (3) Structured - datasets that require geometric comprehension such as object counting.

Method	ACC(%)	Param(M)	FLOPs(G)	Time(ms)
Traditional:				
Full	65.57	85.68	16.86	2.81
Linear	52.94	85.68	16.86	2.81
PEFT:				
Adapter	71.44	85.84 (↑0.19%)	16.89 (↑0.2%)	2.86 (↑1.8%)
VPT	64.85	85.76 (↑0.09%)	17.71 (↑5.0%)	3.01 (↑7.1%)
BitFit	62.05	85.68 (↑0.0%)	16.86 (↑0.0%)	2.81 (↑0.0%)
LoRA	72.25	85.98 (↑0.35%)	16.92 (↑0.4%)	2.99 (↑6.4%)
SCT	72.64	85.79 (↑0.13%)	16.88 (↑0.1%)	2.86 (↑1.8%)
GPS	73.24	85.68 (↑0.0%)	16.86 (↑0.0%)	2.81 (↑0.0%)
DTL	73.26	85.72 (↑0.05%)	16.92 (↑0.4%)	2.92 (↑3.9%)
Ours	**74.04**	**75.43 (↓12.0%)**	**14.84 (↓12.0%)**	**2.33 (↓17.1%)**

Table 2: Comparison with state-of-the-art methods in terms of accuracy, inference time per image, FLOPs, parameters on VTAB-1k. In terms of efficiency, we present both the absolute values and the relative gap in comparison to the full finetuning method.

Baselines First, two traditional fine-tuning methods are incorporated in all experiments: "Full," which fine-tunes the entire pre-trained model, and "Linear," which exclusively fine-tunes the task-specific classification head. Second, we select Adapter (Houlsby et al. 2019), VPT (Jia et al. 2022), BitFit (Zaken, Ravfogel, and Goldberg 2021), LoRA (Hu et al. 2022), SCT (Zhao et al. 2024), GPS (Zhang et al. 2024), and DTL (Fu, Zhu, and Wu 2024) as our PEFT baselines.

Implementation details and Metrics We adhere to the experimental settings described in (Xin et al. 2024b). To ensure methodological consistency across all comparative evaluations, a uniform training protocol was implemented: Optimization utilized a CosineLRScheduler coupled with the AdamW optimizer, maintaining identical hyperparameters throughout 100 training epochs. We set $\alpha = 0.3$ in Eq (10). The computation of the initialized pseudo inverse matrix is based on the 64 images in the training set.

We assess the performance of our approach using several metrics: accuracy, parameters, inference time and FLOPs. Inference time and FLOPs are measured during testing with batch size 256 and the inference time metric represents the time taken to process a single image. All experiments are conducted on the NVIDIA A6000GPU.

Main Results of SFP

Parameter Reduction with Performance Gain As illustrated in Table 1, our method demonstrates strong performance in the datasets with large domain gaps relative to pretraining data. Compared with the current SOTA method DTL, SFP achieves significant accuracy improvements on specialized and structured datasets(notably, a 2.1% gain on specialized datasets). This enhancement is primarily attributed to our knowledge and structure simplification strategy, which effectively prunes detrimental pretrained knowl-

Method	Natural							Specialized				Structured								Average	# Param(M)
	CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	PCam	EuroSAT	Resisc45	Retinopathy	Clevr/Count	Clevr/Distance	DMLab	KITTI/distance	dSprites/loc	dSprites/ori	SmallNORB/azi	SmallNORB/ele		
Traditional:																					
Full	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	85.8
Linear	63.4	85.0	63.2	97.0	86.3	36.6	51.0	78.5	87.5	68.6	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	52.94	0
One Layer:																					
Layer Index	11	9	8	11	10	8	11	6	7	8	6	4	2	8	2	4	9	4	7		
Layer-specific	50.4	87.8	67.8	98.3	88.3	78.0	49.0	84.1	95.1	86.7	74.8	73.8	66.8	47.5	73.6	73.6	43.0	24.3	35.7	68.35	7.09
Ours	**70.5**	**91.6**	**73.5**	**99.2**	**91.2**	90.8	**51.9**	**86.8**	96.2	87.9	75.7	**80.9**	68.2	52.9	81.6	81.6	49.3	31.1	38.5	**73.65**	0.63
Two Layers:																					
Layer Index	3;11	7;10	8;10	10;11	2;11	6;8	3;11	7;10	6;7	6;9	10;11	4;6	2;11	3;8	2;8	3;7	9;11	6;8	2;5		
Layer-specific	59.4	88.2	69.0	98.3	90.0	81.7	50.4	84.0	94.5	86.9	74.8	75.7	66.0	51.2	76.9	78.4	46.2	26.7	38.2	70.33	14.2
Ours	61.4	90.5	72.5	98.5	89.8	**91.3**	46.3	86.7	**96.3**	**88.3**	**76.3**	80.7	**68.5**	**53.4**	**83.0**	**82.3**	**49.8**	**31.7**	**40.2**	73.03	1.22

Table 3: The optimal position for single-/dual-layer substitutions and layer-specific tuning results. "# Param" denotes trainable parameters (M),"Layer-specific" means to freeze the rest and only train the specific layer.

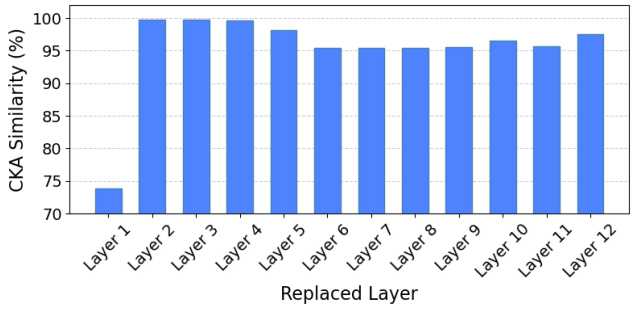

Figure 4: After replacing the specified layer with filter block, the similarity of the image processing results, evaluated using CKA. "Similarity" measures the output resemblance between the modified model (untrained) and the original model on identical image batches.

edge to optimize downstream task performance. However, our method underperforms on CIFAR100 and SUN397, where downstream tasks closely resemble the pretraining data or require indispensable pretrained knowledge, diminishing the necessity of the simplification mechanism. This observation highlights the importance of our specialized initialization procedures.

More importantly, as shown in Fig. 1 and Table 2, unlike existing addition-based paradigms, our approach simplifies the model architecture by fine-tuning the compact filter block. Specifically, it tunes only 0.97M parameters (1.13% of total model parameters), achieving a 12% reduction in both total parameters and FLOPs required for image processing. This efficiency increases processing speed by 17.1%, which is unmatched by existing PEFT methods.

Reliability of Filter Block Substitution As evidenced by Fig. 4, filter block produces representations nearly identical to the original model's outputs when initialized with a single data batch and without any training(similarity evaluation based on CKA(Alvarez 2022)).Given the 92%(1.4GFLOPs vs. 0.11GFLOPs) reduction in processing procedures, we consider this result to be highly commendable. This high fidelity in knowledge preservation establishes a robust reservoir for subsequent training phases, during which task-specific optimization selectively removes non-essential parameters while retaining critical information.

Best Replacement Location Our core principle involves the targeted substitution of interference modules, that means the knowledge encoded in that layer becomes obsolete for the target task. For fine-grained classification tasks (e.g., Pets, Flowers102), optimal performance is achieved by substituting the later layers, as evidenced in Table 3. This finding suggests that although the early visual processing layers of pretrained models retain universal utility, the higher-level representations that encode domain-specific details require task-driven recalibration—a phenomenon consistent with (Raghu et al. 2021). Moreover, for data exhibiting significant semantic differences from the pretraining dataset, replacing the underlying model yields superior results. This means that there was a deviation in the early processing of the image by the pre-trained model.

Layer-specific Tuning Our experiments reveal that certain layers in the pretrained model significantly interfere with downstream tasks, suggesting that specifically training these layers may yield excellent results. Experimental results in Table 3 demonstrate that fine-tuning specific layers containing interference knowledge, while freezing others achieves competitive performance that outperforms full fine-tuning (68.35% vs. 65.57% for single-layer). As a processing for interference information, our approach further refines this paradigm through component substitution. Specifically, the filter block with its compact 0.59M parameters, enables

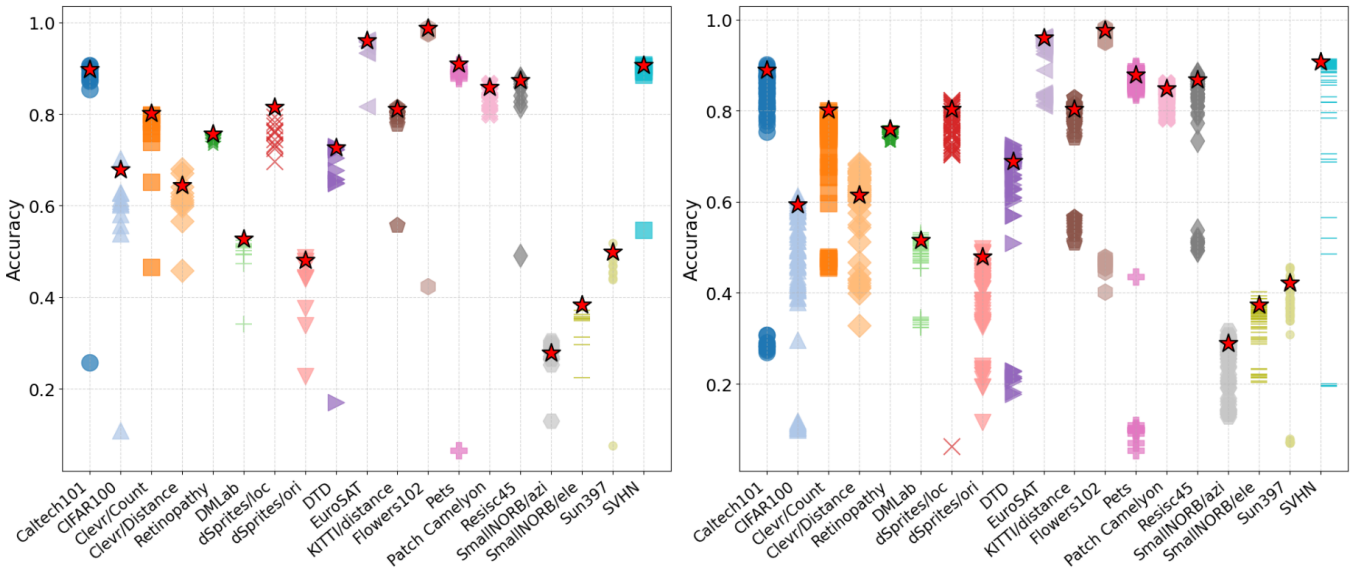

Figure 5: **Left:** Accuracy prediction results for single-layer substitution. Each column displays 12 marks plots corresponding to replacement positions (Layer 1-12), where red stars indicate our method's predicted optimal positions. **Right:** Accuracy prediction results for dual-layer substitution with zero computational overhead.

	Natural	Specialized	Structed
SFP	81.24	86.65	60.51
W/o. LayerNorm	76.90	86.44	59.59
W/o. Initialization	80.01	85.85	59.49

Table 4: Single-layer substitution performance: Custom initialization vs. default Kaiming initialization and LayerNorm tuning vs. freezing.

superior noise filtration compared to layer-specific tuning, which involves 7.09M parameters. This strategic replacement results in a parameter reduction of 6.5M while delivering a 5.3% increase in accuracy.

Prediction of the Optimal Replacement Location As shown in Fig. 5, our predictive positioning framework demonstrates a pronounced ability to identify optimal substitution locations for single-layer replacements. When applied to the 12-layer search space of the ViT-B/16 architecture, the method significantly reduces search complexity, achieving result within 11 iterations and performs well on the vast majority of datasets. In the case of double-layer replacement, two composite effects emerge: (1) interference within the replacement layer, and (2) interactions among filter block modules. Though our approximation strategy ignores the latter effect, it still achieves near-optimal results on most datasets without extra computational cost. As a zero-cost predictor, the results remain acceptable.

Ablation Study

Effect of LayerNorm Incorporating LN layers into training introduces a negligible parameter overhead of 0.0384M. Nonetheless, as evidenced in Table 4, this inclusion confers significant empirical benefits. Specifically, this strategic modification mitigates the representation shifts induced by the knowledge filtration process. By fundamentally altering the model's feature geometry through controlled knowledge adaptation, retraining the LayerNorm parameters becomes imperative to ensure geometric consistency across the transformed activation spaces.

Effect of Filter Block's Initialization The core of filter block construction is to abstract the complex processing within the layer into a linear transformation, and use the linear transformation as its initialization. This operation is designed to preserve the essential knowledge embedded in the pre-trained model. As shown in Table 4, employing the default Kaiming initialization leads to a decline in accuracy across all three task types, thereby confirming that the proposed initialization method effectively retains the key knowledge from the pre-trained model.

Conclusion

In this paper, we propose a novel PEFT paradigm that departs from conventional addition-based methods by adopting a subtractive strategy on existing structures and knowledge. We replace redundant or interfering modules with filter blocks, effectively removing undesired interference during information processing. Empirically, our method improves TOP-1 accuracy on VTAB-1k by 8.47% (74.04% vs. 65.57%) over full fine-tuning while training only 0.97M parameters. Moreover, it reduces the overall parameter count by 12%, lowering deployment costs, and achieves state-of-the-art performance compared with other PEFT methods. In future work, SFP can be extended to tasks such as natural language processing and image generation.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62476194, U23B2049).

References

Alvarez, S. A. 2022. Gaussian RBF centered kernel alignment (CKA) in the large-bandwidth limit. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5): 6587–6593.

Balogh, A.; and Jelasity, M. 2023. On the functional similarity of robust and non-robust neural representations. In *International Conference on Machine Learning*, 1614–1635. PMLR.

Balogh, A.; and Jelasity, M. 2025. How not to Stitch Representations to Measure Similarity: Task Loss Matching versus Direct Matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 15472–15480.

Bansal, Y.; Nakkiran, P.; and Barak, B. 2021. Revisiting model stitching to compare neural representations. *Advances in neural information processing systems*, 34: 225–236.

Bao, H.; Dong, L.; Piao, S.; and Wei, F. 2021. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*.

Basu, S.; Hu, S.; Massiceti, D.; and Feizi, S. 2024. Strong baselines for parameter-efficient few-shot fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 11024–11031.

Chen, A.; Yao, Y.; Chen, P.-Y.; Zhang, Y.; and Liu, S. 2023. Understanding and improving visual prompting: A label-mapping perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19133–19143.

Chen, Z.; Duan, Y.; Wang, W.; He, J.; Lu, T.; Dai, J.; and Qiao, Y. 2022. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*.

Csiszárik, A.; Kőrösi-Szabó, P.; Matszangosz, A.; Papp, G.; and Varga, D. 2021. Similarity and matching of neural network representations. *Advances in Neural Information Processing Systems*, 34: 5656–5668.

Fu, M.; Zhu, K.; and Wu, J. 2024. Dtl: Disentangled transfer learning for visual recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 12082–12090.

Hernandez, A.; Dangovski, R.; Lu, P. Y.; and Soljacic, M. 2023. Model stitching: Looking for functional similarity between representations. *arXiv preprint arXiv:2303.11277*.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-efficient transfer learning for NLP. In *International conference on machine learning*, 2790–2799. PMLR.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Huang, Q.; Dong, X.; Chen, D.; Zhang, W.; Wang, F.; Hua, G.; and Yu, N. 2023. Diversity-aware meta visual prompting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10878–10887.

Jia, M.; Tang, L.; Chen, B.-C.; Cardie, C.; Belongie, S.; Hariharan, B.; and Lim, S.-N. 2022. Visual prompt tuning. In *European conference on computer vision*, 709–727. Springer.

Lee, N.; Ajanthan, T.; and Torr, P. H. 2018. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*.

Lin, W.; Wu, Z.; Yang, W.; Huang, M.; Huang, J.; and Jin, L. 2023. Hierarchical side-tuning for vision transformers. *arXiv preprint arXiv:2310.05393*.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, 10012–10022.

Luo, G.; Huang, M.; Zhou, Y.; Sun, X.; Jiang, G.; Wang, Z.; and Ji, R. 2023. Towards efficient visual adaption via structural re-parameterization. *arXiv preprint arXiv:2302.08106*.

Pan, Z.; Cai, J.; and Zhuang, B. 2023. Stitchable neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16102–16112.

Pan, Z.; Liu, J.; He, H.; Cai, J.; and Zhuang, B. 2024. Stitched ViTs are Flexible Vision Backbones. In *European Conference on Computer Vision*, 258–274. Springer.

Raghu, M.; Unterthiner, T.; Kornblith, S.; Zhang, C.; and Dosovitskiy, A. 2021. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34: 12116–12128.

Sharma, M.; Fantacci, C.; Zhou, Y.; Koppula, S.; Heess, N.; Scholz, J.; and Aytar, Y. 2023. Lossless adaptation of pretrained vision models for robotic manipulation. *arXiv preprint arXiv:2304.06600*.

Sung, Y.-L.; Nair, V.; and Raffel, C. A. 2021. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34: 24193–24205.

Tsai, Y.-Y.; Mao, C.; and Yang, J. 2023. Convolutional visual prompt for robust visual perception. *Advances in Neural Information Processing Systems*, 36: 27897–27921.

Wang, H.; Chang, J.; Zhai, Y.; Luo, X.; Sun, J.; Lin, Z.; and Tian, Q. 2024. Lion: Implicit vision prompt tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 5372–5380.

Xin, Y.; Du, J.; Wang, Q.; Lin, Z.; and Yan, K. 2024a. Vmt-adapter: Parameter-efficient transfer learning for multi-task dense scene understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 16085–16093.

Xin, Y.; Luo, S.; Liu, X.; Zhou, H.; Cheng, X.; Lee, C. E.; Du, J.; Wang, H.; Chen, M.; Liu, T.; et al. 2024b. V-petl bench: A unified visual parameter-efficient transfer learning benchmark. *Advances in Neural Information Processing Systems*, 37: 80522–80535.

Xin, Y.; Yang, J.; Luo, S.; Zhou, H.; Du, J.; Liu, X.; Fan, Y.; Li, Q.; and Du, Y. 2024c. Parameter-efficient fine-tuning for pre-trained vision models: A survey. *arXiv preprint arXiv:2402.02242*.

Yang, T.; Zhu, Y.; Xie, Y.; Zhang, A.; Chen, C.; and Li, M. 2023. Aim: Adapting image models for efficient video action recognition. *arXiv preprint arXiv:2302.03024*.

Zaken, E. B.; Ravfogel, S.; and Goldberg, Y. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*.

Zhai, X.; Puigcerver, J.; Kolesnikov, A.; Ruyssen, P.; Riquelme, C.; Lucic, M.; Djolonga, J.; Pinto, A. S.; Neumann, M.; Dosovitskiy, A.; et al. 2019. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*.

Zhang, J. O.; Sax, A.; Zamir, A.; Guibas, L.; and Malik, J. 2020. Side-tuning: a baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, 698–714. Springer.

Zhang, Z.; Zhang, Q.; Gao, Z.; Zhang, R.; Shutova, E.; Zhou, S.; and Zhang, S. 2024. Gradient-based parameter selection for efficient fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 28566–28577.

Zhao, H. H.; Wang, P.; Zhao, Y.; Luo, H.; Wang, F.; and Shou, M. Z. 2024. Sct: A simple baseline for parameter-efficient fine-tuning via salient channels. *International Journal of Computer Vision*, 132(3): 731–749.