
HEAT: HEAD-LEVEL PARAMETER EFFICIENT ADAPTATION OF VISION TRANSFORMERS WITH TAYLOR-EXPANSION IMPORTANCE SCORES *

Yibo Zhong, Yao Zhou

Sichuan University

Chengdu

zhongyibo@stu.scu.edu.cn yaozhou@scu.edu.cn

ABSTRACT

Prior computer vision research extensively explores adapting pre-trained vision transformers (ViT) to downstream tasks. However, the substantial number of parameters requiring adaptation has led to a focus on *Parameter Efficient Transfer Learning* (PETL) as an approach to efficiently adapt large pre-trained models by training only a subset of parameters, achieving both parameter and storage efficiency. Although the significantly reduced parameters have shown promising performance under transfer learning scenarios, the structural redundancy inherent in the model still leaves room for improvement, which warrants further investigation. In this paper, we propose **H**ead-level **E**fficient **A**daptation with **T**aylor-expansion importance score (**HEAT**): a simple method that efficiently fine-tuning ViTs at head levels. In particular, the *first-order Taylor expansion* is employed to calculate each head's *importance score*, termed *Taylor-expansion Importance Score* (TIS), indicating its contribution to specific tasks. Additionally, three strategies for calculating TIS have been employed to maximize the effectiveness of TIS. These strategies calculate TIS from different perspectives, reflecting varying contributions of parameters. Besides ViT, HEAT has also been applied to hierarchical transformers such as Swin Transformer, demonstrating its versatility across different transformer architectures. Through extensive experiments, HEAT has demonstrated superior performance over state-of-the-art PETL methods on the VTAB-1K benchmark.

1 Introduction

Many works are currently focusing on adapting large models such as *Vision Transformer* (ViT) [6], originally pre-trained on large datasets like ImageNet [5], to diverse downstream tasks. Due to the vast number of parameters in these models, adapting them to various tasks requires significant computational cost, and the performance is often unsatisfactory, thus limiting the models' potential for various applications.

To address these issues, numerous studies are focusing on *parameter efficient transfer learning* (PETL) methods like [10] [31] [22]. These techniques efficiently adapt large pre-trained models by tuning only a fraction of the parameters, leaving the remainder frozen. They achieve superior performance on various downstream tasks compared to full fine-tuning, while requiring fewer parameters for adaptation. For instance, adapter-based PETL methods involve inserting trainable lightweight adapters into the model and keeping the model frozen as the backbone [11] [13]. These adapters are basically *fully connected layers* with small hidden dimensions. During the adaption process, only these adapters need to be trained, while the rest of the parameters remain unchanged, thus reducing computational and storage costs.

However, when the model is used to address a significant number of downstream tasks, bottleneck adapters still require a huge number of parameters. To tackle this problem, recent studies are focusing on reducing redundancies in transformers like *precision redundancy* [15], *rank redundancy* [11] [14] or *density redundancy* [8] in the model. These methods exhibit superior performance with much fewer parameters by reducing various redundancies in the

**Citation: Authors. Title. Pages.... DOI:000000/11111.*

model. However, although these methods have explored various forms of redundancies in the model, it remains to be questioned whether other types of redundancy could be further leveraged to improve the performance.

Among all these redundancies in transformer architecture, redundancy in attention heads is a notable phenomenon in self-attention. Various researches have examined the redundancy in attention heads through attention patterns [16], pruning attention heads [34] [4] [25], attention visualization [33] and probing test [4]. It has been proven that many attention heads produce attention matrices that are alike, thus exhibiting similar behaviours, and there is limited set of attention patterns, which indicates overall overparametrization in *Multi-Head Self Attention* (MHSA) [4] [16]. Moreover, in certain cases, disabling some heads unexpectedly leads to an increase in performance [16]. However, while there is existing research and evidence on the redundancy among attention heads in conventional Transformers, the redundancy in the very low-dimensional adapters introduced in PEFT methods still requires further investigation.

Inspired by the aforementioned studies, in this paper, we propose a novel approach termed as **HEAT** (Head-level Efficient Adaptation with Taylor-expansion importance score) for enhancing efficiency during transfer learning by mitigating redundancy among attention heads. Utilizing first-order Taylor expansion, we can approximate a parameter’s contribution to the model’s loss function with minimal computational cost, thus allowing us to assess the importance of each head efficiently. Within the HEAT framework, we demonstrate its superior performance over state-of-the-art PETL methods, showcasing its efficacy in optimizing computational efficiency and reducing storage costs.

Our contributions are as follows:

- Motivated by the persistence of redundant attention heads in Multi-Head Self-Attention (MHSA) models in PETL, we propose HEAT, a novel PETL approach focused on reducing redundancy and enhancing performance at the head level.
- Introducing the first-order Taylor expansion, HEAT selects and masks less important heads in MHSA relative to the specific task at hand, without requiring manual inspection of attention patterns for each downstream task.
- Extensive experiments demonstrate that employing HEAT leads to improved performance over other state-of-the-art PETL methods, underscoring the effectiveness of head-level adaptation.

2 Related Work

2.1 Parameter-Efficient Transfer Learning

Parameter-Efficient Transfer Learning (PETL) methods focus on adapting large pre-trained models like Vision Transformer (ViT) [6] to various downstream tasks by tuning only a small number of parameters in the model, thus achieving both parameter and storage efficiency. Prior work can be roughly divided into methods that involve inserting tokens and adapters. Token-based methods include [12] [21] [42] [32] [38] [41], which essentially insert trainable tokens into the tokens in Multi-Head Self Attention (MHSA) block. Adapter-related methods, on the other hand, insert lightweight adapters into the model, either in the MHSA or in the Feed-Forward Network (FFN) block [11] [3] [10] [14] [13] [30]. In full fine-tuning, whenever adapting to a new downstream task, a completely new backbone is required because full fine-tuning directly updates parameters in the original model. However, in the majority of PETL methods, during training, only the inserted tokens or adapters are updated while the original model remains frozen, allowing the model to serve as the backbone for multiple downstream tasks without additional parameters. Here we review some widely used PETL methods:

2.1.1 AdaptFormer

AdaptFormer [3] involves introducing two adapters parallel to the FFN block. These two adapters are fully connected layers with weight represented as $W_{down} \in \mathbb{R}^{d \times h}$ and $W_{up} \in \mathbb{R}^{h \times d}$, where d is the dimension of query, key or value and h is a small hidden dimension such that $h << d$. The FFN in AdapterFormer can be expressed as:

$$FFN(X) + s \cdot ReLU(LN(X) \cdot W_{down}) \cdot W_{up} + X \rightarrow X'$$

where s is a hyper parameter, X is input to FFN, X' is AdaptFormer’s output after FFN.

2.1.2 FacT

FacT [14] tensorizes the weights of each ViT, and decompose their increments into lightweight factors. ΔW is the increment matrix of the frozen weight during fine-tuning. ΔW is decomposed into $U \in \mathbb{R}^{d \times r_1}$, $V \in \mathbb{R}^{d \times r_2}$, $\Sigma \in \mathbb{R}^{12L \times r_1 \times r_2}$, where

$$\Delta W = s \cdot \Sigma \times_2 U^T \times_3 V_T$$

in which \times_i is mode- i product:

$$\Delta W_{i,j,k} = s \cdot \sum_{t_1=1}^{r_1} \sum_{t_2=1}^{r_2} \Sigma_{i,t_1,t_2} U_{j,t_1} V_{k,t_2}$$

$$\forall i \in \{1, 2, \dots, 12L\}, \forall j, k \in \{1, 2, \dots, d\}$$

2.1.3 VPT

VPT [12] inserts trainable tokens into the input space while keeping the model backbone frozen. In VPT-Shallow, the tokens are inserted into the first transformer layer only, while in VPT-Deep, the tokens introduced at every transformer layer’s input space.

2.1.4 LoRA

LoRA [11] inserts two adapters parallel to the Multi-Head Self Attention (MHSA) block, which equals to decomposing the increment matrix of query W_q and value W_v into $A_{q/v} \in \mathbb{R}^{d \times h}$, $B_{q/v} \in \mathbb{R}^{h \times d}$, the MHSA in LoRA can be expressed as:

$$MHSA(X) + s \cdot X A_q B_q \rightarrow Q$$

$$MHSA(X) + s \cdot X A_v B_v \rightarrow V$$

where s is a hyper-parameter, X is the input to MHSA and Q and V are output query and value of MHSA. *Bi-LoRA* [15] is a quantized version of LoRA using low-precision adapters.

2.1.5 NOAH

NOAH [40] utilizes VPT, LoRA and Adapter. It views these PETL methods as prompt modules and learns the optimal combination of these prompt modules through neural architecture search algorithm, without the need for manual design.

Among the aforementioned methods, much emphasis is placed on researching efficient adaptation of the MHSA. Although these methods have significantly improved parameter efficiency and model performance, there is still a need for additional exploration regarding the multi-head design of attention heads in MHSA, which could be leveraged to further enhance parameter efficiency and model effectiveness. Therefore, our motivation stems from addressing this problem and utilize it to optimize the current PETL methods.

2.2 Redundancy in Attention Heads

Research has delved into the attention mechanism and the essential design aspects of multi-heads in transformers, pointing out the problem of overparametrization and redundancy in the Multi-head Self Attention (MHSA). Studies focusing on attention patterns [16], head pruning techniques [34] [4] [25], visualization of attention [33], and probing tests [4] have shown that a considerable number of attention heads generate similar attention matrices, displaying redundant behavior. This suggests that MHSA contains a limited variety of attention patterns, confirming its overall excess in parameters [4] [16]. Additionally, in specific scenarios, deactivating certain heads unexpectedly results in performance improvements [16].

Consequently, several approaches have aimed at head pruning: [25] employ a greedy algorithm for iteratively removing less contributive heads, [34] utilize stochastic gates and an L_0 penalty for pruning, observing that specialized heads are pruned last, and [1] apply a zero-shot strategy for head pruning before fine-tuning without task-specific data.

Although these methods have studied the mechanism of multiheads in MHSA and proposed measures to prune redundant heads, there is still room for further exploration of pruning strategies. Therefore, we propose an effective strategy to reduce this redundancy and apply it to the field of PETL, where redundancy among multiheads has yet to be investigated, to further enhance model performance.

3 Method

We start with an overview of the proposed HEAT (Head-level Efficient Adaptation with Taylor-expansion importance scores) framework and then explore the calculation method for TIS (Taylor-Expansion Importance Scores). The discussion of method concludes with an examination of several different TIS strategies for HEAT.

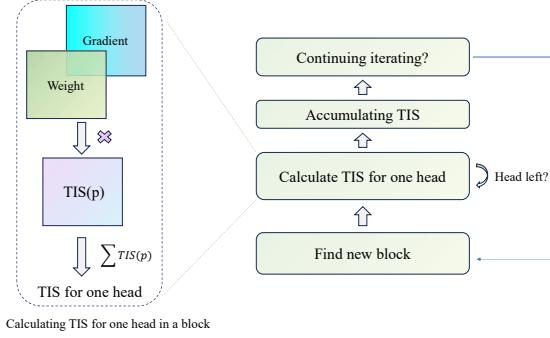


Figure 1: Complete process of calculating and accumulating TIS for each head across all transformer blocks

3.1 Overall Framework

In general, under the HEAT framework, we first compute the cumulative TIS for each head across all transformer blocks to obtain a final TIS value. Then, based on a specified hyper-parameter N_m , which determines the number of candidates to mask, we select N_m heads with the lowest TISs and mask out the channels corresponding to these heads from the input. This process is illustrated in Figure 1. To compute TIS, we first train the original model for a small number of epochs to acquire the components required for our calculations. Then, we sort all heads according to their TISs and select N_m heads as a set of candidate C using the strategies mentioned above. For all heads in C , we mask them during forward propagation and train for the remaining epochs.

3.1.1 Model structure

There are two types of blocks in ViT: 1) Multi-Head Self Attention (MHSA) and 2) Feed-Forward Network (FFN). In MHSA, we use $W_q, W_k, W_v, W_o \in \mathbb{R}^{d \times d}$ to represent the heads transforming the query, key, value, and output, respectively. All queries, keys, and values undergo transformations through W_q, W_k and W_v before the calculation of self-attention. Because each transformation is multi-head, we can further divide these weights into N heads, where each head is represented as $W_q^{(i)}, W_k^{(i)}, W_v^{(i)}, W_o^{(i)}$, $i \in \{1, 2, \dots, N\}$, the MHSA is then calculated as:

$$MHSA(X) = \sum_{i=1}^N \text{softmax} \left(\frac{XW_q^{(i)}(W_k^{(i)})^T X^T}{\sqrt{d_k}} \right) XW_v^{(i)}(W_o^{(i)})^T \quad (1)$$

where X is the input, and d is the feature dimension of the input. In case of adapter based PETL methods which focusing on MHSA such as LoRA and FacT, the typical structure includes an adapter that reduces the feature dimension (downward adapter) and an adapter that restores the feature dimension (upward adapter). We refer to them as $A_{down} \in \mathbb{R}^{d \times r}$ and $A_{up} \in \mathbb{R}^{r \times d}$, in which r is a small hidden dimension and $r \ll d$, respectively for brevity. Passing the input through these two adapters can be expressed as:

$$X' = MHSA(X) + s \cdot X A_{down} A_{up} \quad (2)$$

where X' is the output after the MHSA. This can be interpreted as equivalent to passing the input through a unified weight matrix $W \in \mathbb{R}^{d \times d}$, which we denote as the adapter weight in Figure 2 and serves the same functionality as the frozen weight in the backbone. Therefore, we can regard these two adapters as a unified whole.

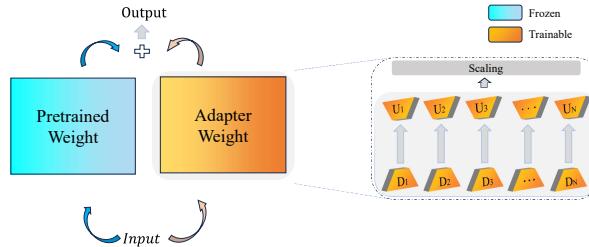


Figure 2: An illustration of the structure of adapter based fine-tuning methods. D_1 to D_n here are downward adapters for all heads while U_1 to U_n are upward adapters.

In HEAT, we also employ a similar approach, utilizing the basic model structure of A_{down} and A_{up} mentioned above in the q and v transformation in MHSA. Moreover, binary quantization method is applied to these adapters, following the approach of [15].

3.1.2 Calculating TISs

To effectively find the least important heads, we propose a Taylor-expansion-based importance score (TIS) criterion for selecting those heads. For each head h , we calculate its TIS by summing up every parameter's TIS in it, and accumulating it across all transformer blocks to represent global importance, which can be expressed as:

$$TIS_h = \sum_{i=0}^{N_b} \sum_{p \in P_h} TIS(p) \quad (3)$$

where N_b is the number of blocks in transformer, and P_h represents the parameters in head h . Once we have obtained all the heads' TISs, for a given N_m indicating number of heads to mask, we mask N_m heads with the lowest TISs.

3.1.3 Head-level Mask

Finally, once we have selected our candidate C by calculating TIS scores, masking those heads is equivalent to:

$$W_m = W \cdot M \quad (4)$$

where W is the weights of multi-head attention and W_m is W after masked. M is a matrix with $\forall i \in C, M^i = 0$ and $\forall i \notin C, M^i = 1$. An example of this is in Figure 3. Since X' is formed by concatenating the outputs of N head transforming X , we can divide X' into N channels, each channel with a token size of $\frac{d}{N}$. Therefore, masking a specific head i can be expressed as:

$$X'^{(i)} = 0, i \in C \quad (5)$$

where $i \in \{1, 2, \dots, N\}$ is the subscript of a certain head in C . This is also equal to:

$$X' = X' \cdot M \quad (6)$$

where M is a mask variable with the same shape as X' with $\forall i \in C, M^i = 0$ and $\forall i \notin C, M^i = 1$. W here can be W_q, W_k, W_v . However, following LoRA, Fact and Bi-LoRA, only W_q and W_v is being used. We use N_m to represent the size of C . Thus, by masking N_m selected heads during forward propagation, their information is discarded during training.

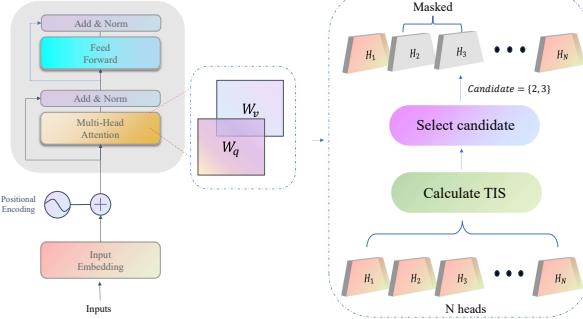


Figure 3: An example of masking head, here we use $N_m = 2$, and selected candidate C is $\{2, 3\}$

3.2 Taylor-expansion Importance Score

For a given number N_m constraining on how many heads to mask, we choose N_m heads that are redundant, which intuitively means they contributes the least to the performance. There has been many ways of pruning in model compression such as greedy algorithm [23] [36], sparse regularization [19][35] and reinforcement learning [9]. The very focus of ours is Taylor expansion criterion [27][26][18]. To effectively find the least important heads, we propose a Taylor-expansion-based importance score criterion for selecting those heads.

The process of finding the least important heads can be denoted as an optimization process in which, for a parameter p_i , loss function is $L(p_i)$ when p_i is non-zero, and $L(p_i = 0)$ when p_i is set to zero. The change to the loss function when setting p_i to zero is:

$$TIS(p_i) = \Delta L(p_i) = |L(p_i = 0) - L(p_i)| \quad (7)$$

where $\Delta L(p_i)$ is our Taylor-expansion Importance Score (TIS), indicating how much change zeroing out a parameter p_i can bring to the model's performance. To approximate $L(p_i = 0)$, we utilize first-order Taylor expansion, the polynomial at point p is expressed as follow:

$$f(x) = \sum_{i=0}^i \frac{f^{(i)}(p)}{i!} (x - p)^i + R_i(x) \quad (8)$$

Therefore, we can approximate $L(p_i = 0)$ as:

$$L(p_i = 0) = L(p_i) - \frac{\partial L}{\partial p_i} p_i + R_1(p_i = 0) \quad (9)$$

Using Lagrange form to calculate the remainder $R_1(p_i = 0)$:

$$R_1(p_i = 0) = \frac{\partial^2 L}{\partial(p_i^2 - \varepsilon)} \frac{p_i^2}{2} \quad (10)$$

where ε is a real number between 0 and p_i . The remainder is ignored following [7] [27] mainly for huge computational cost. Finally, by substitute Eq 9 into Eq 7, we get $\Delta L(p_i)$ as:

$$TIS(p_i) = \Delta L(p_i) = \left| \frac{\partial L}{\partial p_i} p_i \right| \quad (11)$$

where $\frac{\partial L}{\partial p_i}$ can be easily computed as gradient of p_i is already available during backward propagation while p_i being the weight.

	Natural						Specialized				Structured										
	Cifar100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Cheer-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	Average	Size(MB)
<i>Traditional Fine-Tuning</i>																					
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	68.9	327
Linear	64.4	85.0	63.2	97.0	86.3	36.6	51.0	78.5	87.5	68.5	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	57.6	0
<i>PETL methods</i>																					
BitFit	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	65.2	0.39
VPT-Shallow	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	67.8	0.24
VPT-Deep	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	72.0	2.03
AdaptFormer	73.8	92.3	72.7	99.3	91.6	89.1	56.5	87.8	95.5	84.9	75.2	83.3	62.5	52.4	81.7	86.2	55.9	34.4	40.2	76.7	0.56
LORA	72.0	91.2	71.6	99.1	91.3	88.9	56.4	87.2	94.6	83.9	74.9	83.7	64.0	52.3	81.2	84.8	53.3	38.1	43.4	76.4	1.13
NOAH*	69.6	92.7	70.2	99.1	90.4	86.1	53.7	84.4	95.4	83.9	75.8	82.8	68.9	49.9	81.7	81.8	48.3	32.8	44.2	75.5	1.37
Bi-LoRA	72.1	91.7	71.2	99.1	91.4	90.2	55.8	87.0	95.4	85.5	75.5	83.1	64.1	52.2	81.3	86.4	53.5	36.7	44.4	76.7	0.14
Fact-TT	73.4	91.0	72.4	99.2	91.4	90.1	56.6	87.3	94.7	84.5	75.8	83.0	64.9	51.3	81.4	87.4	53.2	33.5	44.3	76.7	0.30
HEAT (Ours)	72.7	92.3	71.4	99.2	91.4	90.2	55.9	88.0	95.8	85.5	75.5	83.7	64.9	52.3	82.3	86.7	53.5	40.0	44.8	77.2	0.14

Table 1: Full results on VTAB-1K benchmark. The average is computed based on the group-wise averages. (The * denotes results obtained by NOAH are from using normalized input)

3.3 Head-level TIS strategies

3.3.1 Criterion value

Empirically, we find that using only the absolute value for TIS leads to less satisfactory results across various datasets. This is likely because relying solely on the absolute value limits the potential for reducing the loss. while in some tasks, aiming to reduce loss can also cause unstable performance. Therefore, we adopt three scoring strategies for calculating TIS for a parameter p_i : 1) the absolute value $\left| \frac{\partial L}{\partial p_i} p_i \right|$ 2) the positive value $\frac{\partial L}{\partial p_i} p_i$ 3) the negative value $-\frac{\partial L}{\partial p_i} p_i$ to maximize performance:

$$TIS(p_i) = \begin{cases} \left| \frac{\partial L}{\partial p_i} p_i \right|, & \text{absolute value} \\ \frac{\partial L}{\partial p_i} p_i, & \text{positive value} \\ -\frac{\partial L}{\partial p_i} p_i, & \text{negative value} \end{cases} \quad (12)$$

3.3.2 Joint or separate calculation

When masking heads, we primarily focus on q and v in MHSA. Therefore there are two strategies for calculating the TIS of a certain head h : 1) calculating TIS for q and v separately 2) calculating TIS for q and v jointly. Substituting into Eq 3, calculating separately can be expressed as:

$$TIS_q = \sum_{i=0}^{N_b} \sum_{p \in P_q} TIS(p) \quad (13)$$

and:

$$TIS_v = \sum_{i=0}^{N_b} \sum_{p \in P_v} TIS(p) \quad (14)$$

where N_b is the number of blocks in ViT, P_v or P_q is parameters from value and query in the corresponding head. In this manner, candidates for q and v are calculated separately, leading to different C_q and C_v . Substituting into Eq 3, calculating jointly can be expressed as:

$$TIS_{qv} = \sum_{i=0}^{N_b} \sum_{p \in P_q} TIS(p) + \sum_{j=0}^{N_b} \sum_{p \in P_v} TIS(p) \quad (15)$$

In this manner, final TIS for a head are calculated by summing up its TIS_q and TIS_v , and we use final TISs of all heads to determine the candidates. Thus in this case, we obtain the same C_q and C_v . The selected heads are the least important across q and v , so TIS in this manner represents a head's global importance across both q and v .

One characteristic of HEAT is that, apart from the basic adapter model structure, it operates solely at the input level without making any modifications to the model's architecture. Therefore, the improved accuracy of HEAT does not stem from increasing the number of parameters but rather from reducing the amount of information learned by the model. This seemingly counter-intuitive observation can be further investigated to better understand its implications.

4 Experiments

4.1 Transfer Learning on VTAB-1K Benchmark

4.1.1 Datasets

We first evaluate our method on the VTAB-1K benchmark [39], a suite of vision tasks designed to evaluate general visual representations, thus demonstrating our method's general capability. The VTAB-1K consists of 19 tasks from various domains and with various semantics, divided into three groups: 1) **Natural** 2) **Specialized** 3) **Structured**. For each task, VTAB-1K uses only 1000 examples. All our reported results are top-1 accuracy on the test sets. Following [15] [14] [40] [12], we use unnormalized input for all datasets. Some previous work which normalizes input is re-implemented by [15], so in such case we directly refer to their original data.

4.1.2 Comparison with current PETL methods

we compare our method with **VPT** [12], **NOAH** [40], **AdaptFormer** [3], **BiTFit** [37], **FacT-TT** [14] and **LoRA** [11] and **Bi-LoRA** [15]. The hidden dimension r is set to 8 for AdaptFormer and LoRA. FacT-TT's rank r is searched from {8, 16, 32}. Other methods' settings follow their best recipes in papers. For HEAT, TIS is calculated following the three strategies, and hyper-parameter N_m is roughly searched from {1, 3, 6} in HEAT empirically. Some of the top accuracy is also reported from directly masking front heads for their high performance on certain datasets. Following [15] and [40], we use AdamW as optimizer with a learning rate of 1e-3 and a batchsize of 64 to train for 100 epochs. We first train for 10 epochs to calculate TIS and set the mask candidate C for the remaining 90 epochs. For backbone, Following [15] [14], all methods use ViT-B/16 pre-trained on Image-Net 21K as backbone.

4.1.3 Results

All our results are shown in Table 1, in which HEAT outperform all existing PETL methods on VTAB-1K benchmark. It outperform Bi-LoRA by 0.5% and outperform Bi-AdaptFormer [15], the SOTA by 0.2%. Performance of HEAT also exceeds LoRA by 0.8%, which can be regarded as the baseline for the PETL method on MHSA, which is the focus of our study. To be specific, HEAT achieve state-of-the-art performance on 8 tasks across 19 datasets. from which we can conclude:

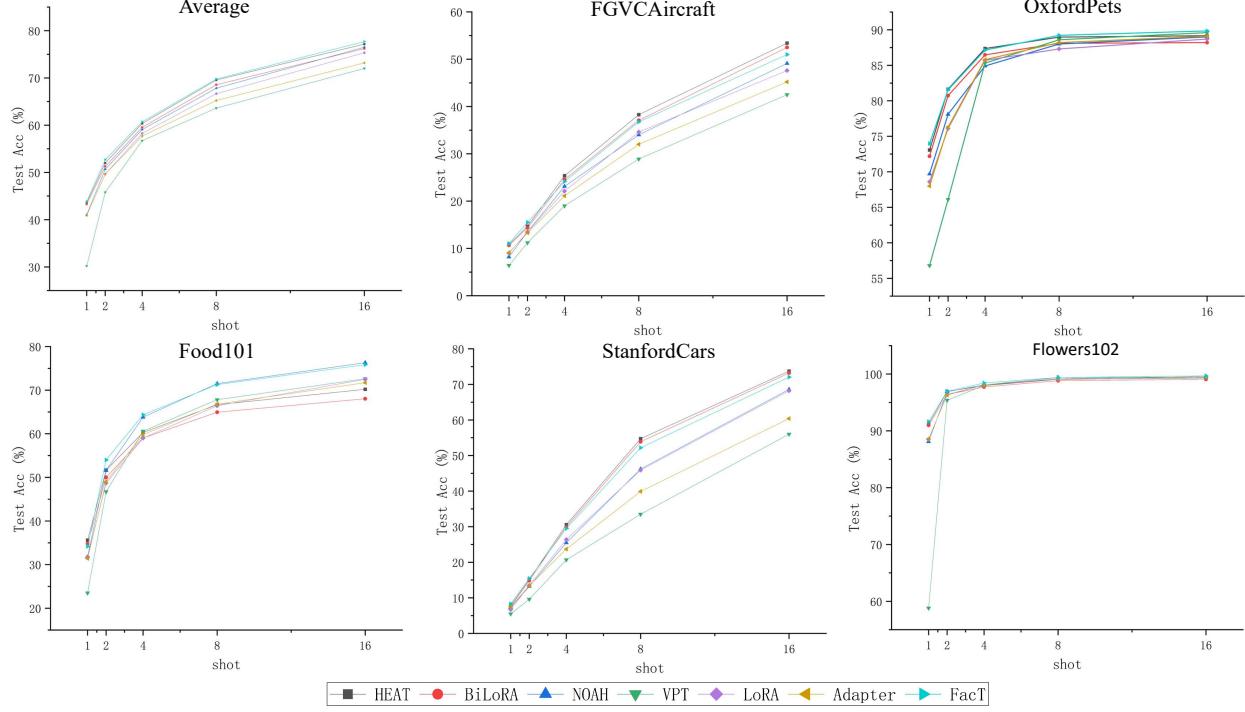


Figure 4: Full results on few shot learning

(1) Reducing the redundancy among attention heads in MHSA can bring performance improvement on various domains. Therefore, any other methods utilizing the MHSA block in transformer could potentially benefit from our method based on an empirical inference: our method can improve performance by reducing redundancy among attention heads. Therefore, it is possible for our method to be combined with other PETL methods that focus on reducing different types of redundancy, thereby achieving even higher performance, since the effect of reducing redundancy on performance might be additive.

(2) HEAT's parameter efficiency matches that of LoRA and Bi-LoRA, as it avoids introducing additional structures and parameters. This eliminates the need for methods that solely increase parameter count to boost accuracy, such as scaling up the model. A core mechanism of HEAT is its selective processing of information at the head-level directly on the input, a process that doesn't influence the model structure. Consequently, HEAT's head-level operations have no impact on the model's parameter count. Therefore, after applying this mechanism, HEAT maintains the same model structure as before, remaining highly parameter efficient.

(3) When the head-level mask of HEAT result in performance degradation, we simply opt for a hyper-parameter setting of $N_m = 0$. In this scenario, no information is discarded during the head-level mask, making the HEAT model structure consistent with LoRA. Therefore, LoRA could be viewed as a special case of HEAT when $N_m = 0$. Given that the head-level mask in HEAT only zeros out certain channels of the input without altering the model itself, in the worst-case scenario, HEAT maintains the original performance when no head-level mask is applied without the risk of performance reduction.

However, through extensive experiments, we find that by masking a small number of heads such as $N_m = 1$, we can often achieve the same accuracy without mask and in many cases we can even improve the performance. Most of our best results come from $N_m = 1$ or $N_m = 3$, which makes sense because having too many masked heads can also lead to the loss of essential information for the task, rather than just reducing redundancy. Previous work has proven that many heads play an important and consistent role, pruning them out will have a detrimental effect on the overall performance [16] [34]. The upper limit $N_m = 6$ of our search space is determined through extensive experiments, and exceeding this limit typically results in noticeable performance degradation since more than half of the information is discarded. More discussions are provided in ablation studies.

4.2 Few Shot Learning

Few-shot learning is a situation when acquiring data for downstream tasks is quite challenging, and only a limited number of training samples are available for each task. It assesses the model’s capability of learning from a limited training data and apply that knowledge to predict unseen instances. It can assess a model’s ability to generalize and perform well on tasks with few training samples available.

4.2.1 Datasets

Experiments are conducted on five datasets including FGVC-Aircraft [24], Oxford-Pets [29], Food-101 [2], Stanford Cars [17] and Oxford-Flowers102 [28]. Experimental settings are employed with 1, 2, 4, 8, and 16 shots.

4.2.2 Comparison with current PETL methods

We compare HEAT with Bi-LoRA, LoRA, VPT, NOAH, AdaptFormer, FacT-TT. The hyper-parameter h for LoRA and AdaptFormer are set to 8. For HEAT, the results are reported based on the rough selection using only either positive or negative values in Eq 12 of the TIS strategy. As for hyper-parameter N_m , we only choose the top accuracy in $N_m = 1$ or $N_m = 3$. The average results are obtained from three distinct seeds for each shot.

4.2.3 Results

As shown above in Figure 4, HEAT outperform the vast majority of compared PETL methods, and achieve performance comparable to FacT-TT, the SOTA. HEAT performs on par with FacT-TT on FGVC-Aircraft, Oxford-Pets, Stanford Cars and Oxford-Flowers102. On Food-101, HEAT shows slightly inferior performance compared to FacT-TT. On average, HEAT achieves performance on par with FacT-TT and outperform Bi-LoRA and LoRA. HEAT surpass Bi-LoRA’s performance by roughly 0.8 in average, demonstrating that HEAT can still achieve high performance even when only a limited number of training samples are available.

Method	Average	Natural	Specialized	Structured
Full	74.99	79.2	86.2	59.7
Linear	62.60	73.5	80.8	33.5
VPT-Deep	71.55	76.8	84.5	53.4
Bi-LoRA	76.67	82.0	86.8	61.2
HEAT	76.93	82.3	87.1	61.4

Table 2: Results on VTAB-1K using Swin-B as backbone, reported accuracy from average, natural, specialized and structured

4.3 On Hierarchical Transformers

We also apply our method on Swin Transformer [20], a widely used and representative design of hierarchical structures of transformers aiming to improve ViT.

Given that the number of heads and dimension of token is different across layers, simply setting a N_m hyper-parameter is not enough. Therefore, we also set a hyper-parameter *percent* indicating the percentage of heads to mask in window attention. The calculation of TIS and candidate C for swin transformer is restricted in layers with the same number of attention heads in this case.

We use Swin-B as our backbone following [15]. It consists of four stages, with depths {2, 2, 18, 2} and number of heads {4, 8, 16, 32}. In this case, take *percent* = 0.25 as an example, we set the number of candidate for the four stages separately as {1, 2, 4, 8}. While in many tasks, the deeper the layer is, it learns more task specific information, setting a fixed number of heads also proves to be effective since it discard little information in deeper layer while discarding much information in shallower layer where there is more redundancy.

The results are shown in Table 2, using **swin-B** pre-trained on ImageNet-21K [5] as backbone for HEAT. Following [15] [14], we compare HEAT with several methods which could also be applied on swin transformer: Full, Linear, VPT [12] and Bi-LoRA [15]. From the results, we observe that HEAT achieve high performance in Swin Transformer even with only a few hyperparameter settings, showcasing its versatility across different transformer designs. While its performance may be suboptimal compared to ViT, there is potential for further improvement as our experiments

were conducted in only a limited number of cases. Additionally, our method on Swin also demonstrates that HEAT is structure-agnostic and method-agnostic, indicating its potential to enhance performance across various architectures. Further experimental details are provided in the appendix.

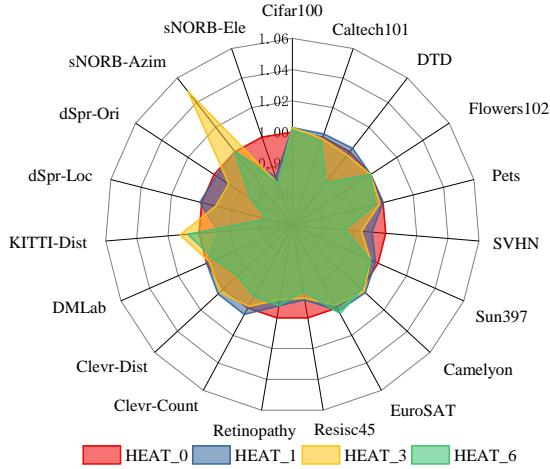


Figure 5: To test the effect of arbitrary mask, arbitrary masks of 1, 3, and 6 heads were applied on HEAT, and comparisons were made with the baseline without masking. HEAT_n means that in the HEAT model, there are n heads being masked arbitrarily with HEAT₀ serving as the baseline. The values in the figure represent the ratio of results after applying the mask to the baseline results.

4.4 Ablation Study

We begin by conducting a study on the role of arbitrarily masking head itself in reducing attention heads redundancy. Then we study the impact of quantization on the performance of the HEAT model. Additionally, since our method involves a three-fold strategy selection process, with each strategy further fine-tuned using hyper-parameters N_m chosen from the set {1, 3, 6}, we here delve into the contribution of each strategy and examine how the choice of N_m affects the overall performance.

4.4.1 Arbitrary mask

To prove that the same phenomenon of many attention heads exhibiting similar behaviours can also be observed in ViT, we test the effect on the performance by arbitrarily discarding information from some heads through masking them. Our observation reveals that not only does this not decrease the performance too much, but in many tasks, it improves the performance notably as shown in Figure 5, thus demonstrating that same redundancy also exists in ViT in current PETL methods. While there is improvement, we did not utilize this property to extreme as we were arbitrarily selecting heads to mask. One option to maximize the performance is to study the attention patterns for each downstream task. However, manually studying the patterns is not feasible when dealing with a significant amount of downstream tasks for its huge computational cost. This leads us to seek methods capable of efficiently identifying mask candidates in the study of PETL without the need for manual investigation and it brings us to our proposed HEAT framework.

	Natural	Specialized	Structured	Average
HEAT	81.87	86.20	63.53	77.20
HEAT (Full Precision)	81.71	85.58	62.84	76.71

Table 3: Comparison between HEAT and HEAT using FP32 precision adapters on VTAB-1K, in which the results of three groups in VTAB-1K are reported.

4.4.2 Quantization

For the adapters used in the model structure of HEAT, we employed a quantization method to reduce computational and storage costs. Here, we analyze the impact of using FP32 precision adapters (i.e., without quantization) on the results of HEAT. From the Table 3, it can be observed that using FP32 precision adapters results in slight degradation in HEAT’s

	Natural	Specialized	Structured	Average
<i>Negative</i>				
HEAT_1	81.59	85.68	63.04	76.77
HEAT_3	81.67	85.38	62.70	76.58
<i>Positive</i>				
HEAT_1	81.51	85.28	63.08	76.62
HEAT_3	81.66	85.10	62.91	76.56

Table 4: Effect of hyper-parameter N_m on overall performance on VTAB-1K. Reported results are obtained when $N_m = 1$ or $N_m = 3$ from two TIS strategies (positive and negative). $N_m = n$ is denoted as HEAT_n here.

performance, but it remains comparable to some competitive methods such as Fact and AdaptFormer. These results indicate that quantization is an integral part of the HEAT approach, and leveraging quantization is essential to fully exploit the advantages of the HEAT method and achieve superior performance.

4.4.3 TIS strategy

Regarding the three strategies, we individually analyzed the performance of employing each strategy alone, and the results are presented in Figure 6, from which we can observe that using positive value $\frac{\partial l}{\partial p} p$ leads to best accuracy on majority of the datasets, while on certain datasets using absolute value $| - \frac{\partial l}{\partial p} p |$ or negative value $- \frac{\partial l}{\partial p} p$ demonstrate better results. The reason probably lies in that using positive value enables us to mask heads that will increase the loss function when pruned, which intuitively means improving the performance. Note that using absolute value only account for top accuracy on one datasets and surprisingly using negative value has accounted for better accuracy on a few datasets, this is probably due to aiming to reduce the loss by using positive value leads to unstable performance while using absolute value or negative value to maintain the loss in such case could potentially yields a better results.

4.4.4 Jointly or separately

There's two options when setting the candidates C for q and v : setting C separately, which means the calculation of TIS is restricted within q or v or setting C jointly, where the calculation is performed jointly between q and v . Results from calculating jointly is significantly better than calculating separately. This is probably due to that calculating TISs jointly represents global importance across both q and v , and keeping C the same between q and v enable stable and consistent performance.

4.4.5 Number of heads

The results of different N_m using negative or positive TIS strategy are shown in Table 4. For hyper-parameter N_m , we determine the search space for it through empirical evidence. When N_m exceeds 6, the performance experiences significant degradation. This is mostly due to the loss of essential information. When $N_m = 1$ or $N_m = 3$, we

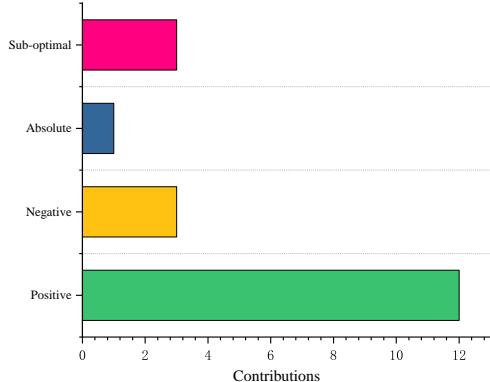


Figure 6: Contribution of each strategy, we report the number of datasets on which each strategy achieving the best results. Sub-optimal means none of the strategies has shown improvement over baseline method

mostly mask heads with redundant information, while $N_m = 6$ or above, we could potentially mask heads that are not redundant or even crucial for a specific task. Therefore, we can conclude that redundant heads constitute a small fraction of the total heads and majority of the heads learn features that are useful to a specific task.

5 Conclusion

In this paper, we explore the redundancy among attention heads in Multi-Head Self Attention and mask attention heads to reduce this redundancy. We further introduce HEAT, a parameter-efficient approach based on Taylor expansion Importance Scores, which allows us to identify and mask heads that are least relevant to a specific task. Through extensive experiments, our method demonstrates superior performance compared to all existing PETL methods, achieving state-of-the-art results on the VTAB-1K benchmark. Our findings underscore the presence of redundancy in multi-head design within parameter-efficient transfer learning, and suggest that effectively selecting heads to mask can lead to improved performance.

References

- [1] Yuchen Bian et al. “On attention redundancy: A comprehensive study”. In: *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: human language technologies*. 2021, pp. 930–945.
- [2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. “Food-101-mining discriminative components with random forests”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part VI 13*. Springer. 2014, pp. 446–461.
- [3] Shoufa Chen et al. “Adaptformer: Adapting vision transformers for scalable visual recognition”. In: *Advances in Neural Information Processing Systems 35* (2022), pp. 16664–16678.
- [4] Kevin Clark et al. “What does bert look at? an analysis of bert’s attention”. In: *arXiv preprint arXiv:1906.04341* (2019).
- [5] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
- [6] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).
- [7] Akash Sunil Gaikwad and Mohamed El-Sharkawy. “Pruning convolution neural network (squeezenet) using taylor expansion-based criterion”. In: *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*. IEEE. 2018, pp. 1–5.
- [8] Shwai He et al. “Sparseadapter: An easy approach for improving the parameter-efficiency of adapters”. In: *arXiv preprint arXiv:2210.04284* (2022).
- [9] Yihui He and Song Han. “Adc: Automated deep compression and acceleration with reinforcement learning”. In: *arXiv preprint arXiv:1802.03494* 2 (2018).
- [10] Neil Houlsby et al. “Parameter-efficient transfer learning for NLP”. In: *International conference on machine learning*. PMLR. 2019, pp. 2790–2799.
- [11] Edward J Hu et al. “Lora: Low-rank adaptation of large language models”. In: *arXiv preprint arXiv:2106.09685* (2021).
- [12] Menglin Jia et al. “Visual prompt tuning”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 709–727.
- [13] Shibo Jie and Zhi-Hong Deng. “Convolutional bypasses are better vision transformer adapters”. In: *arXiv preprint arXiv:2207.07039* (2022).
- [14] Shibo Jie and Zhi-Hong Deng. “Fact: Factor-tuning for lightweight adaptation on vision transformer”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 1. 2023, pp. 1060–1068.
- [15] Shibo Jie, Haoqing Wang, and Zhi-Hong Deng. “Revisiting the parameter efficiency of adapters from the perspective of precision redundancy”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 17217–17226.
- [16] Olga Kovaleva et al. “Revealing the dark secrets of BERT”. In: *arXiv preprint arXiv:1908.08593* (2019).
- [17] Jonathan Krause et al. “3d object representations for fine-grained categorization”. In: *Proceedings of the IEEE international conference on computer vision workshops*. 2013, pp. 554–561.
- [18] Woosuk Kwon et al. “A fast post-training pruning framework for transformers”. In: *Advances in Neural Information Processing Systems 35* (2022), pp. 24101–24116.

- [19] Vadim Lebedev and Victor Lempitsky. “Fast convnets using group-wise brain damage”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2554–2564.
- [20] Ze Liu et al. “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 10012–10022.
- [21] Yuning Lu et al. “Prompt distribution learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 5206–5215.
- [22] Gen Luo et al. “Towards efficient visual adaption via structural re-parameterization”. In: *arXiv preprint arXiv:2302.08106* (2023).
- [23] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. “Thinet: A filter level pruning method for deep neural network compression”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5058–5066.
- [24] Subhransu Maji et al. “Fine-grained visual classification of aircraft”. In: *arXiv preprint arXiv:1306.5151* (2013).
- [25] Paul Michel, Omer Levy, and Graham Neubig. “Are sixteen heads really better than one?” In: *Advances in neural information processing systems* 32 (2019).
- [26] Pavlo Molchanov et al. “Importance estimation for neural network pruning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 11264–11272.
- [27] Pavlo Molchanov et al. “Pruning convolutional neural networks for resource efficient inference”. In: *arXiv preprint arXiv:1611.06440* (2016).
- [28] M-E Nilsback and Andrew Zisserman. “A visual vocabulary for flower classification”. In: *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*. Vol. 2. IEEE. 2006, pp. 1447–1454.
- [29] Omkar M Parkhi et al. “Cats and dogs”. In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3498–3505.
- [30] Jonas Pfeiffer et al. “Adapterfusion: Non-destructive task composition for transfer learning”. In: *arXiv preprint arXiv:2005.00247* (2020).
- [31] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. “Learning multiple visual domains with residual adapters”. In: *Advances in neural information processing systems* 30 (2017).
- [32] Sheng Shen et al. “Multitask vision-language prompt tuning”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024, pp. 5656–5667.
- [33] Jesse Vig and Yonatan Belinkov. “Analyzing the structure of attention in a transformer language model”. In: *arXiv preprint arXiv:1906.04284* (2019).
- [34] Elena Voita et al. “Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned”. In: *arXiv preprint arXiv:1905.09418* (2019).
- [35] Jianbo Ye et al. “Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers”. In: *arXiv preprint arXiv:1802.00124* (2018).
- [36] Ruichi Yu et al. “Nisp: Pruning networks using neuron importance score propagation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 9194–9203.
- [37] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. “Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models”. In: *arXiv preprint arXiv:2106.10199* (2021).
- [38] Yuhang Zang et al. “Unified vision and language prompt learning”. In: *arXiv preprint arXiv:2210.07225* (2022).
- [39] Xiaohua Zhai et al. “A large-scale study of representation learning with the visual task adaptation benchmark”. In: *arXiv preprint arXiv:1910.04867* (2019).
- [40] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. “Neural prompt search”. In: *arXiv preprint arXiv:2206.04673* (2022).
- [41] Kaiyang Zhou et al. “Conditional prompt learning for vision-language models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16816–16825.
- [42] Kaiyang Zhou et al. “Learning to prompt for vision-language models”. In: *International Journal of Computer Vision* 130.9 (2022), pp. 2337–2348.