This CVPR 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.

PELA: Learning Parameter-Efficient Models with Low-Rank Approximation

Yangyang Guo, Guangzhi Wang, Mohan Kankanhalli
National University of Singapore

Abstract

Applying a pre-trained large model to downstream tasks is prohibitive under resource-constrained conditions. Recent dominant approaches for addressing efficiency issues involve adding a few learnable parameters to the fixed backbone model. This strategy, however, leads to more challenges in loading large models for downstream fine-tuning with limited resources. In this paper, we propose a novel method for increasing the parameter efficiency of pre-trained models by introducing an intermediate pre-training stage. To this end, we first employ low-rank approximation to compress the original large model and then devise a feature distillation module and a weight perturbation regularization module. These modules are specifically designed to enhance the low-rank model. In particular, we update only the low-rank model while freezing the backbone parameters during pre-training. This allows for direct and efficient utilization of the low-rank model for downstream fine-tuning tasks. The proposed method achieves both efficiencies in terms of required parameters and computation time while maintaining comparable results with minimal modifications to the backbone architecture. Specifically, when applied to three vision-only and one vision-language Transformer models, our approach often demonstrates a merely ∼0.6 point decrease in performance while reducing the original parameter size by 1/3 to 2/3. We release our code at link.

1. Introduction

Pre-training a large model and fine-tuning it at hand has become a *de facto* paradigm in diverse research fields [9, 10, 60]. While significant performance has been achieved, building such models often compromises increased memory usage and longer training time. Despite these challenges, recent advances in the appreciation of the scaling law [28] and emergent abilities [62] of language pre-training have further fueled practitioners' interest in developing and utilizing large models.

As it is usually prohibitive to deploy these models for downstream tasks, recent studies have resorted to bypassing the fine-tuning of the entire model. Typical approaches

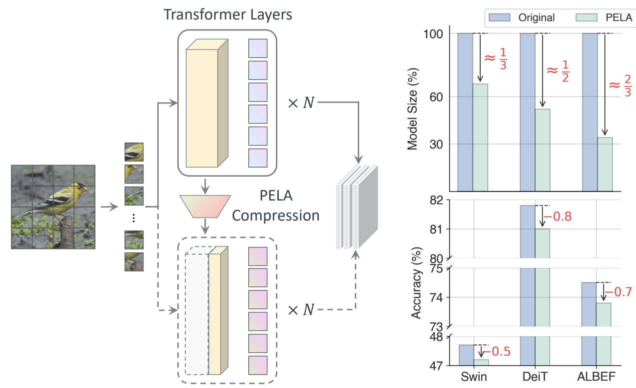

Figure 1. Overview and performance of our proposed PELA method. Left: Using PELA, we compress the trainable weights of a typical ViT model while preserving its overall architecture. Right: Comparison of the Original and our PELA *w.r.t* relative model size and accuracy metric on three pre-trained Transformers.

often introduce a few more learnable parameters to the backbone model while freezing the rest, *e.g.*, adapter [52] and prompt tuning [25] add tunable parameters to the middle and peripheral token positions of Transformers, respectively. However, this approach inevitably leads to the following two disadvantages. First, the potential of pre-trained large models is not fully exploited as the majority of parameters are not tuned with downstream task objectives. Second, loading the pre-trained model becomes even more burdensome for researchers with limited resources. In contrast, conventional methods such as knowledge distillation (KD) [20, 21, 51] and quantization [7, 23] can partially alleviate this issue. Yet, there is currently no established approach for constructing a high-performing student model of KD and non-differentiable operators of quantization usually make it less feasible to perform back-propagation.

This paper targets developing a highly parameter-efficient approach to help downstream task fine-tuning, as illustrated in Fig. 1. By parameter-efficiency, we refer to a compressed model with reduced-size (*e.g.*, 2× smaller), easy-to-implement, computationally-efficient, and minimal-architectural-change merits. Our method offers a pre-trained compressed model that downstream tasks can directly perform fine-tuning on, in contrast to previous approaches such as LoRA [22], adapter [52], and prompt tuning [25]. In particular, this method is specially designed to address the over-parameterization problem [1], where we resort to the low-rank approximation to replace the pre-trained weight matrix in each matrix multiplication operation with two low-rank matrices. By this means, the original model size and fine-tuning time are both fairly reduced. However, using this naive approach to perform fine-tuning yields less satisfactory outcomes (refer to Sec. 4.4). We attribute this fact to two reasons: Directly decomposition with low-rank approximation cannot effectively learn instance-level discriminative representations; and the intermediate feature distribution is perturbed after this operation, resulting in sub-optimal performance.

In order to approach this problem, we propose fully taking advantage of the pre-trained model via two modules. The implementation involves two parallel model branches: one consists of the pre-trained model with fixed parameters during pre-training, while the other is the low-rank model with tunable parameters. Based upon this framework, our first module distills the feature knowledge from the large pre-trained model to our compressed low-rank model in terms of each Transformer layer. The other module helps bind the weight change within a pre-defined perturbation radius. These two modules help the low-rank model mimic the feature distribution of the large pre-trained model, thereby enhancing its discrimination capability. During fine-tuning, we simply use the low-rank model as a replacement for the original large one to achieve parameter and computational efficiency for downstream tasks.

As far as we know, the literature on achieving a desirable efficiency-effectiveness trade-off by using low-rank approximation on pre-trained Transformer weights is quite limited. We apply our method to three vision-only Transformers *i.e.*, DeiT [55], DeiT-III-Large [57] and SwinT [40], with 1/2 to 2/3 of the original model parameters; and one vision-language Transformer – ALBEF [34] where the parameter size is reduced to 1/3 of the original model. We then conduct extensive experiments on a range of downstream tasks, including image classification, semantic segmentation, and object detection for the vision-only Transformers; Visual entailment, visual grounding, cross-modal retrieval, and visual question answering for the vision-language Transformer. Our approach achieves performance that is highly comparable to the backbone model, with differences mostly around 0.6 points, despite using only 1/3 to 2/3 of the original FLOPs. In addition, this parameter-efficiency benefit further enables the model to scale with larger batch sizes, leading to improved performance that sometimes even outperforms backbones.

2. Related Work

2.1. Parameter-Efficient Learning

Efficiency has long been an engaging problem in a variety of research areas [27, 50]. After stepping into the deep representation learning era, the progressive improvements in our community often trade with a large number of model parameters, latency, and footprints [42, 59]. With this concern, previous efforts have been mainly devoted to three distinctive directions: knowledge distillation (KD), quantization, and pruning. Deemed as a principled model compression algorithm, early KD aims to transfer the knowledge from a cumbersome teacher model to a lightweight student model via class logit alignment [21, 47]. Recent focus has been shifted to feature-based knowledge transfer due to its performance advantage over conventional logit-based ones [20, 26, 46, 70]. For example, [26, 51] distill the knowledge from hidden states and attention matrices, which on the other hand, can also bypass the logit-free training objectives. However, choosing features from which layers to align remains challenging as there is no teacher-student layer match from a theoretical basis. Quantization, from another angle of efficient learning, maps larger bit parameters to smaller ones, *e.g.*, 32-bit floating point to an 8-bit integer [45]. This kind of method is not dependent on the model structures, which makes it flexible in various neural networks [7, 23, 37]. The key downside lies in its performance reduction and possible infeasibility for back-propagation. Different from the above two categories, pruning is leveraged to remove unnecessary or less important components in models [59]. By removing some connections [66] or parameters [30], the original dense network reduces to a sparse one, in which the required capacity for storage as well as the amount of computations will dwindle.

Transformer-based approaches have succeeded in diverse research domains since their introduction [9, 58]. These models often involve billions of parameters, which consequently, motivates some specific methods working on addressing the parameter-efficiency problem [31]. The typical strategy is to add a few learnable parameters while freezing the majority of the Transformer backbone during downstream training. For instance, prompt tuning appends some task-specific parameters into the input space [25]; Adapter models introduce several learnable MLP components into each Transformer layer [52]; and fine-tuning bias only has also been proven effective for maintaining good performance of large language models [73].

2.2. Low-Rank Approximation

Low-rank approximation aims to decompose one matrix into two smaller matrices, subject to the constraint that the resulting matrices have reduced rank [48, 49]. One key merit of this algorithm is data compression, whereby previous work has applied it to principal component analysis [43] and recommendation [13, 19].

Pertaining to Convolutional Neural Networks (CNNs), some approaches apply the low-rank approximation to each feature map via higher-order tensor decomposition [12, 53, 74]. Dynamically decomposing trainable matrices has also attracted much attention [67, 69, 71]. Some more studies explored other aspects of low-rank approximation, such as rank learning [24], constrained optimization [33], and employing it specifically in token embedding matirx [5, 31] or self-attention computation in Transformers [61]. LoRA [22] models the residual of parameters with low-rank approximation, wherein only the newly decomposed matrices are exploited for downstream training and it thus achieves significantly reduced trainable parameters. Despite its benefits, the LoRA approach still has limitations, as it necessitates the storage and reloading of large pre-trained weights in hard disk and GPU memory, respectively. In other words, only the newly introduced trainable parameters that are of a smaller magnitude compared to the full parameters are updated for fine-tuning, making it similar to adapters [15, 52] and prompt tuning [25]. Unlike existing approaches, our method uses low-rank approximation during pre-training to entirely replace the pre-trained weights with reduced low-rank matrices. As a result, we achieve both memory and computational efficiency goals for downstream fine-tuning tasks.

3. Method

Transformers have grown into a fundamental building block of many modern vision models [10, 18]. Take the seminal Vision Transformer (ViT) as an example. ViT first divides an RGB image $I \in \mathbb{R}^{3\times H\times W}$ into $M \times M$ non-overlapping patches. Together with a class token, these image patches are thereafter fed into N layers with self-attention as the basic operation. To this end, a set of query, key, and value matrices are transformed from the patch embedding to token features $\mathbf{X} \in \mathbb{R}^{(M^2+1)\times d}$, where d denotes the embedding size, followed by several feedforward layers and residual connections. At their core lies the fully connected layer, which is often wrapped in the attention score estimation and MLP operations - $\mathbf{W}^T\mathbf{X} + \mathbf{b}$, where $\mathbf{W} \in \mathbb{R}^{d_{in}\times d_{out}}$ is the learnable weight matrix and $\mathbf{b} \in \mathbb{R}^{d_{out}}$ denotes the bias, and $d_{in} = d$ for the first layer.

3.1. Low-rank Approximation

Over-parameterization is a common issue in modern large models [1]. In this work, we aim to address this problem by reducing the number of model parameters. Inspired by the success of low-rank approximation in other domains [12, 53], we propose to apply this technique directly to the matrix multiplication operations in ViT,

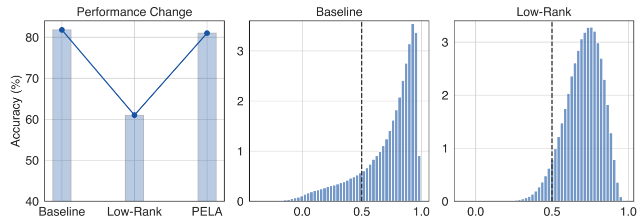

Figure 2. Performance comparison of three models and statistics of the instance-level feature similarity. Left: We use the DeiT model as the baseline and show the performance of its directly low-rank approximation and PELA variants. The middle and right sub-figures illustrate the instance-level feature similarity of DeiT and directly low-rank model variants, respectively.

$$
\begin{aligned}
\mathbf{W}^T\mathbf{X} &\approx (\mathbf{U}\mathbf{V}^T)^T\mathbf{X} \\
&= \mathbf{V}(\mathbf{U}^T\mathbf{X}),
\end{aligned} \tag{1}
$$

where $\mathbf{U} \in \mathbb{R}^{d_{in}\times d_{lr}}$ and $\mathbf{V} \in \mathbb{R}^{d_{out}\times d_{lr}}$ are low-rank matrices, and d_{lr} represents the desired rank of $\mathbf{W}$. Note that the weight matrices in a deep learning model are often with full-rank, *i.e.*, $rank(\mathbf{W}) = min(d_{in}, d_{out})$. Under such conditions, we seek approximately equal the original matrix and deliberately choose a smaller d_{lr}, *e.g.*, $\frac{1}{4}min(d_{in}, d_{out})$. The second equation constantly holds in neural networks due to the natural associative law. This property allows us to achieve computational efficiency without needing to recover the original weight matrix $\mathbf{W}$ after applying the low-rank approximation. We utilize the well-known SVD approach [32] to perform the low-rank approximation as,

$$
\texttt{SVD}(\mathbf{W}^T) = \mathbf{U}^*\mathbf{\Sigma}\mathbf{V}^*, \tag{2}
$$

where $\mathbf{\Sigma} \in \mathbb{R}^{d_{in}\times d_{out}}$ is a rectangular diagonal matrix with non-negative real numbers on the diagonal, and the singular values are sorted in a monotonously decreasing order; $\mathbf{U}^* \in \mathbb{R}^{d_{in}\times d_{in}}$ and $\mathbf{V}^* \in \mathbb{R}^{d_{out}\times d_{out}}$ are complex unitary matrices. We then formalize the low-rank matrices using the following transformation,

$$
\begin{cases}
\mathbf{U} = \mathbf{U}^*_{[:,:d_{lr}]}\mathbf{\Sigma}^{\frac{1}{2}}_{[:d_{lr},:d_{lr}]}, \\
\mathbf{V} = (\mathbf{\Sigma}^{\frac{1}{2}}_{[:d_{lr},:d_{lr}]}\mathbf{V}^*_{[:d_{lr},:]})^T,
\end{cases} \tag{3}
$$

where $[:, : d_{lr}]$ implies we truncate the given matrix with the top-d_{lr} columns and other truncation operations can also be easily deduced.

Preliminary observation. We apply this low-rank approximation to the fully connected layers of pre-trained models. Unfortunately, this process delivers less desirable results, *e.g.*, the accuracy drops from 81% to 61% as seen in Fig. 2. This indicates that the compressed low-rank model

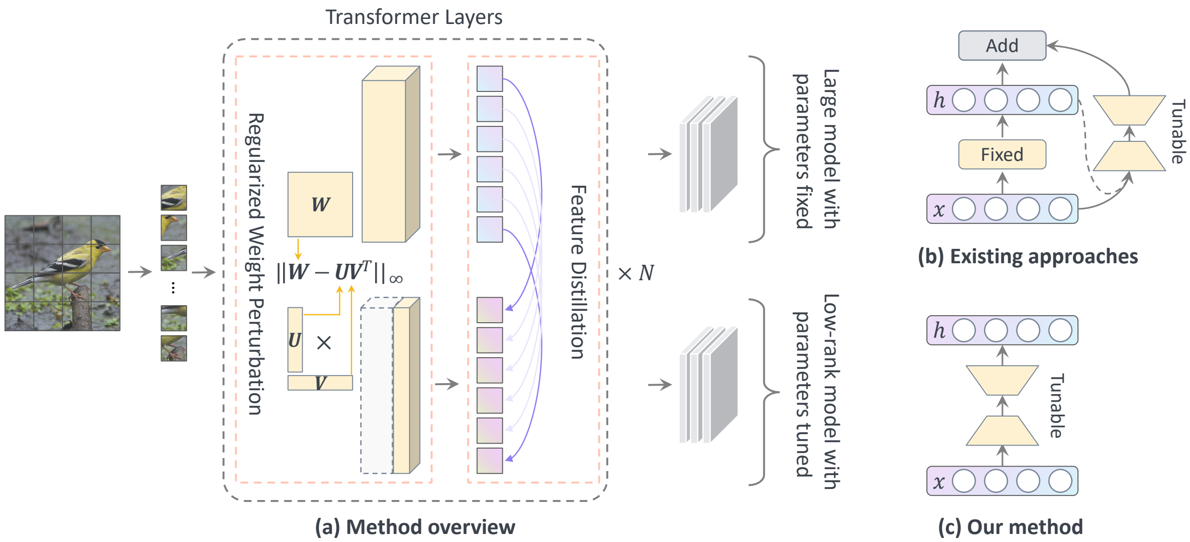

Figure 3. Overview of our proposed PELA and pipeline comparison with existing methods. (a) We leverage a typical ViT model as the base for the illustration of our method. The two involved modules, *i.e*., Feature Distillation aligns the token features in an apple-to-apple fashion of each layer, and Regularized Weight Perturbation bounds the recovered weight matrices. During fine-tuning on downstream tasks, existing approaches use both the fixed pre-trained model weights and the newly added parameters (b). In contrast, our PELA keeps only the low-rank model while excluding the large pre-trained model for efficient computation (c).

does not effectively learn instance-level discriminative representation. Moreover, we found that the learned features after low rank are confined in a narrow feature space. In particular, the right two subfigures in Fig. 2 demonstrate that the feature similarity of each class of the low-rank model is drastically higher than before.

To overcome this, we propose to take full advantage of the large pre-trained model and harness it to guide the training of the low-rank model. Specifically, as shown in Fig. 3, we first perform low-rank approximation on the pre-trained model and retain both models. The parameters of the large pre-trained model are frozen while we train only the low-rank model. Our method further involves two modules: *feature distillation* to align features between these two models, *regularized weight perturbation* to constrain the affinity of the recovered matrix and original matrix. We name this method **PELA**, dubbed ***P**arameter-**E**fficient models for **L**ow-rank **A**pproximation*. To the best of our knowledge, there is limited research on constructing an effective low-rank model based on pre-trained Transformers. Therefore, we aim to address this gap by investigating the potential of low-rank approximation to achieve an optimal efficiency-effectiveness trade-off.

3.2. Feature Distillation

As highlighted in the previous sub-section, the low-rank approximation can alter the feature distribution of the pre-trained model. To address this problem, we resort to feature-based knowledge distillation, which has been proven effective in aligning the features between models [46]. Nevertheless, the low-rank compression is performed on each matrix multiplication operation, rather than specific Transformer blocks or layers. Directly distilling knowledge from all the output features of the original model leads to more clutter as some low-rank compression is already wrapped within the self-attention computation. Thanks to the layer-wise residual connection of Transformers, we employ a compromise in this work – simply aligning the token features of each layer. From a general view of typical Transformer models, the feature distillation loss is defined as follows,

$$\begin{aligned}\mathcal{L}_{fd} &= \sum_{i=1}^{N} \mathcal{D}(\mathcal{M}_s(\mathbf{X}_s^i), \mathcal{M}_t(\mathbf{X}_t^i)), \\ &= \frac{1}{2N}\sum_{i=1}^{N} \parallel \mathcal{M}_s(\mathbf{X}_s^i) - \mathcal{M}_t(\mathbf{X}_t^i) \parallel^2,\end{aligned} \tag{4}$$

where $\mathbf{X}_s^i$ and $\mathbf{X}_t^i$ denote the i-th layer token features of the compressed low-rank model and original model, respectively; $\mathcal{M}$ is a transformation that transfers the feature to the target feature space and we employ identity mapping in our implementation. In this way, the output features from each ViT layer of the low-rank model are expected to share a similar distribution with that of the corresponding large pre-trained model.

An alternative view from knowledge distillation. Recent studies have shown that feature-based knowledge distillation significantly outperforms conventional logit-based one [20]. Nonetheless, how to design a student model and transfer the knowledge from the teacher model remains challenging as it is rather difficult to define teacher-student feature matching. Our method offers a neat solution to this problem due to the following two reasons: 1) Unlike previous approaches (*e.g*., [26, 51] manually removing certain layers), the low-rank approximation is effortless and straightforward to compress the cumbersome teacher model to a lightweight student model. 2) There exists a natural correspondence between the teacher model and student model as we have not excessively altered the model architectures.

3.3. Regularized Weight Perturbation

An ideal low-rank approximation is to learn an approximating matrix of the original one subject to a reduced rank constraint. This leads to an efficiency-effectiveness dilemma – A larger rank corresponds to a lower reconstruction error and vice versa. Intuitively, we associate the matrix reconstruction with that of weight perturbation, which is relatively new as opposed to the feature/input perturbation robustness problem [63]. As a result, a smaller rank in our method, from the other angle, can be seen as more weight perturbations. To reduce the negative influence of these perturbed parameters, we use the l_∞-norm for constraining the reconstruction error,

$$\begin{cases} \| \hat{\mathbf{W}}^{(k)} - \mathbf{W}^{(k)} \|_\infty \leq \epsilon, \\ \hat{\mathbf{W}}^{(k)} = \mathbf{U}^{(k)}(\mathbf{V}^{(k)})^T, \forall k \in [K] \end{cases} \tag{5}$$

where ϵ represents the perturbation radius, $\mathbf{W}^{(k)}$ is the original weight matrix, and $[K]$ denotes the weight index set. Given ϵ, preserving the neural network robustness against weight perturbation can be cast as the following optimization problem [63],

$$\mathcal{L}_{rwp} = \sum_{k=1}^{|[K]|} (\| \hat{\mathbf{W}}^{(k)} - \mathbf{W}^{(k)} \|_\infty - \epsilon). \tag{6}$$

3.4. Training

The above two modules enable us to capture the compelling discriminative capability of large pre-trained models. To obtain a compact low-rank model, we comprehensively consider the objectives from both the base pre-training and our proposed two modules,

$$\mathcal{L} = \mathcal{L}_{base} + \alpha \mathcal{L}_{fd} + \beta \mathcal{L}_{rwp}, \tag{7}$$

where α and β are loss weight hyper-parameters and $\mathcal{L}_{base}$ is the loss functions of the original pre-training tasks. It can be the classification loss of a typical ViT, or vision-text matching and masked language modeling losses of a vision-language model. We then optimize our model on the same datasets as the pre-trained model, such as ImageNet [8].

After this intermediate pre-training stage, our low-rank model is smoothly deployed for downstream fine-tuning since the model architecture is rarely altered. In contrast to existing methods such as prompt tuning [25], adapters [52], and LoRA [22], which require both the large pre-trained model and the fine-tuning parameters, we only keep the low-rank model for efficient inference and parameter usage (see Fig. 3 for a visual comparison).

3.5. Complexity Analysis

Before analyzing the complexity of our method, we first let $d_{lr} = \frac{1}{\kappa}\frac{d_{in} \times d_{out}}{d_{in}+d_{out}}$, where κ is a positive number and we name it *compression ratio*. We choose to use a universal compression ratio for all the matrix multiplication operations for simplicity while leaving the exploration of dynamic ratios for different layers as future work.

Let us consider the case where $\kappa = 2$ and a single patch feature $x \in \mathbb{R}^{d_{in}}$ for downstream fine-tuning. Recall Eqn. 1, the original matrix multiplication takes $\mathcal{O}(d_{in} \times d_{out})$ to operate. However, with our PELA method, this time complexity reduces to $\mathcal{O}((d_{in} + d_{out}) \times d_{lr}) = \frac{1}{2}\mathcal{O}(d_{in} \times d_{out})$[1]. Similarly, as the majority operation in existing Transformers is matrix multiplication (excluding some very few layer normalization parameters and bias parameters), the model size thus also roughly halves from its original scale. This is why our method is significantly different from other recent efficient approaches such as LoRA [22], where the overall model size in fact increases.

4. Experiments

4.1. Common Efficient Learning Baselines

We evaluated our PELA against four efficient baselines: **TinyBERT [26]** and **MaskAlign [68]** from the feature-based knowledge distillation group; **ToMe [2]** - a recent strong vision token pruning approach; and **LoRA [22]**, which is a widely used parameter-efficient transfer learning baseline. However, we excluded some experiments due to certain incompatibilities, such as using ToMe for the Swin model and for the visual grounding task.

4.2. Experiments on Vision-Only Models

4.2.1 Baseline Models and Results

We applied our method to the widely used DeiT-Base [55] and Swin-Base [40] models. To ensure comprehensive coverage, we also selected DeiT-III-Large [57] which is larger in model size and requires much longer training time. The compression ratio is 1/2 and 1/3 for the DeiT models and Swin, respectively. After the low-rank approximation, we trained our model on the ImageNet-1k dataset [8] and evaluated it on the corresponding validation set, and report the results in Table 1. As expected, the model parameters and FLOPs *for inference* are significantly reduced according to

[1]We refer this reduced complexity only to the matrix multiplication since we do not optimize other operations such as attention computation.

Table 1. Model performance of image classification on ImageNet-1K [8] with 224x224 resolution. The parameters and FLOPs are estimated during inference.

Method		Params(M)	GFLOPs	Acc(%)
ViT-Base [10]		86.6	35.1	77.9
CrossViT-B [4]		105.0	40.3	82.2
T2T-ViT-24 [72]		64.1	25.5	82.3
RegNetY-16G [44]		83.6	31.9	82.9
DeiT	Base [55]	86.6	33.7	81.8
	TinyBert [26]	44.2	17.3	78.0
	MaskAlign [68]	44.2	17.3	78.2
	ToMe [2]	86.6	16.5	76.4
	PELA	44.1	17.0	81.0
Swin-Base	Base [40]	87.8	30.3	83.5
	TinyBert [26]	58.6	20.6	78.8
	MaskAlign [68]	58.6	20.6	79.1
	PELA	62.2	21.3	82.5
DeiT-III-Large	Base [57]	304.4	119.4	84.9
	TinyBert [26]	156.8	61.5	79.2
	MaskAlign [68]	156.8	61.5	79.5
	PELA	153.2	59.8	83.9

Table 2. Model performance of semantic segmentation on the ADE20K dataset [76] with UperNet [64].

Backbone		Params(M)	GFLOPs	mIoU
ResNet-101 [16]		85.5	689	44.9
PatchConvNet-B60 [56]		141.0	1,258	48.1
MAE ViT-B [18]		163.9	2,343	48.1
DeiT	Base [55]	121.4	320.4	45.0
	LoRA [22]	124.8	331.1	40.6
	TinyBert [26]	79.0	214.5	36.4
	MaskAlign [68]	79.0	214.5	36.8
	PELA	78.9	203.4	43.2
Swin-Base	Base [40]	121.3	798.6	47.7
	LoRA [22]	124.7	822.6	44.2
	TinyBert [26]	92.1	721.4	40.0
	MaskAlign [68]	92.1	721.4	39.6
	PELA	79.3	685.3	47.2
DeiT-III-Large	Base [57]	428.4	1,155	47.0
	LoRA [22]	440.4	1,190	44.7
	TinyBert [26]	280.8	784	38.1
	MaskAlign [68]	280.8	784	38.4
	PELA	277.2	739	45.6

each respective compression ratio. On the flip side, the dropped accuracy of the two base models is 0.8% and 1.0%, respectively. Even for the relatively larger model DeiT-III-Large, our method only trades 1.0% accuracy with half of the parameters and FLOPs. Moreover, our PELA surpasses other efficient learning baselines by a notable margin.

4.2.2 Downstream Tasks and Results

After the backbones are pre-trained on the ImageNet dataset, as per prior studies [16, 18], we further evaluated the model performance on downstream semantic segmentation and object detection tasks.

The results are presented in Table 2 and Table 3, which illustrate the effectiveness of our method in performing semantic segmentation and object detection tasks, respectively. While our approach benefits from the reduced memory and computation requirements, the involvement of downstream frameworks and heads limits the extent to which these benefits can be realized when compared to vanilla classification. For instance, the reduced FLOPs for Swin-Base on object detection in Table 3 are 18% as compared to the previous 30% in Table 1. Nevertheless, our approach still performs comparably with each respective model, demonstrating its effectiveness in balancing the trade-off between efficiency and accuracy. Notably, our PELA significantly outperforms LoRA in terms of both model performance and model size.

Table 3. Model performance of object detection on the MSCOCO dataset [38] with Cascade Mask RCNN [3, 17].

Backbone		Params(M)	GFLOPs	AP^{box}
ResNet-50 [16]		77.3	411.0	46.3
ResNeXt-101-32 [65]		96.0	546.1	48.1
Swin-Base	Base [40]	145.0	1,501	50.1
	LoRA [22]	149.0	1,547	46.1
	TinyBert [26]	115.8	1,302	41.1
	MaskAlign [68]	115.8	1,302	41.1
	PELA	103.0	1,232	49.0

4.3. Experiments on Vision-Language Model

4.3.1 Baseline Model and Downstream VL Tasks

Traditional visual-language pre-training approaches [6, 54] frequently utilized pre-extracted CNN features for image representation, often requiring precise bounding box annotations. In contrast, ALBEF [34] leverages ViT for visual feature extraction during pre-training and has exhibited exceptional performance across a variety of VL tasks. Therefore, we chose ALBEF as our evaluation testbed to assess the effectiveness of our method. Furthermore, the all-in Transformer nature of ALBEF enabled us to effortlessly achieve more compression. In this context, we used 1/3 of the parameters of the original ALBEF model.

We utilized four downstream vision-language tasks in this work, including Image-Text Retrieval, SNLI-VE, VG, and VQA [14]. A detailed introduction to these tasks can be found in the supplementary material. For the experiments, we strictly followed the implementation of ALBEF except for reducing the batch size due to resource constraints.

4.3.2 Overall Results

The results on these downstream tasks are reported in Table 4, 5, and 6. From these tables, we have the following three important observations. 1) The recent approach ALBEF [34] has demonstrated significant performance improvements over conventional methods like LXMERT [54]

Table 4. Performance comparison of text retrieval (TR) and image Retrieval (IR) on the Flickr30K and MSCOCO datasets.

Dataset	Model		Params	TFLOPs	TR			IR		
					R@1	R@5	R@10	R@1	R@5	R@10
Flickr30K	UNITER [6]		110	0.37	87.3	98.0	99.2	75.6	94.1	96.8
	VILLA [11]		110	-	87.9	97.5	98.8	76.3	94.2	96.8
	ALBEF	Base [34]	419	7.41	93.4	99.5	99.6	80.6	95.8	98.0
		LoRA [22]	431	7.49	92.1	99.2	99.0	80.2	95.6	97.7
		TinyBERT [26]	230	4.66	57.6	82.8	89.9	40.8	70.6	79.4
		MaskAlign [68]	230	4.66	59.0	84.2	90.9	41.1	70.4	80.5
		ToMe [2]	419	2.61	74.8	92.6	96.4	62.0	86.2	91.5
		PELA	173	2.58	91.6	99.3	99.6	79.7	94.8	97.5
MSCOCO	UNITER [6]		110	0.37	65.7	88.6	93.8	52.9	79.9	88.0
	OSCAR [36]		110	-	70.0	91.1	95.5	54.0	80.8	88.5
	ALBEF	Base [34]	419	7.41	72.6	91.2	95.2	54.9	80.5	88.1
		LoRA [22]	431	7.49	73.2	91.7	95.9	56.5	81.3	88.9
		TinyBERT [26]	230	4.66	33.6	62.1	74.8	22.6	49.8	63.2
		MaskAlign [68]	230	4.66	35.7	64.9	77.3	24.2	52.5	65.5
		ToMe [2]	419	2.61	56.2	82.3	90.1	41.7	71.0	81.3
		PELA	173	2.58	71.6	91.0	95.3	55.1	80.8	88.3

Table 5. Model performance on visual entailment and VQA. Params (M) and TFLOPs are counted based on the VQA model.

Model		Params	TFLOPs	SNLI-VE		VQA	
				val	test	test-dev	test-std
VisualBERT [35]		134	0.37	-	-	70.80	71.00
ViLT [29]		118	1.01	-	-	70.94	-
LXMERT [54]		224	0.41	-	-	72.42	72.54
UNITER [6]		116	0.37	78.59	78.28	72.70	72.91
12-in-1 [41]		-	-	-	76.59	73.15	-
ALBEF	Base	581	7.05	79.29	79.79	74.55	74.89
	LoRA	644	7.14	79.34	79.53	71.07	-
	TinyBERT	392	4.55	73.83	73.31	61.33	-
	MaskAlign	392	4.55	73.74	73.48	63.85	-
	ToMe	581	2.55	77.58	78.02	68.59	-
	PELA	259	2.47	78.55	78.66	73.84	73.87

and UNITER [6]. However, superior performance is achieved at the expense of increased parameters and FLOPs, mainly due to the usage of a cumbersome trainable ViT for image processing. In comparison to the baselines, which use a universal Transformer for both vision and language, such as UNITER [6], ALBEF offers superior visual features but introduces a larger model size and computational complexity. 2) Our PELA method helps alleviate this problem through the low-rank approximation. As can be observed, PELA is able to achieve comparable performance to ALBEF while using only 1/3 of the parameters and FLOPs. This translates to a significant reduction in model size and computation, with most performance degradation limited to just one point. 3) Regarding the comparison with efficient learning baselines, our PELA approach consistently achieves better performance in most cases. The only exception is for retrieval tasks, where PELA exhibits slightly inferior model performance compared to LoRA. However,

Table 6. Model performance on the challenging weakly-supervised visual grounding task.

Model		Val	TestA	TestB
ARN [39]		32.78	34.35	32.13
CCL [75]		34.29	36.91	33.56
ALBEF	Base	57.94	65.07	45.75
	PELA	57.06	65.85	45.10

it is important to note that LoRA requires a larger number of model parameters and FLOPs.

4.4. Ablation Study

Effectiveness of the two modules. We first studied the model performance of direct decomposition of pre-trained weights using low-rank approximation. However, as indicated in Table 7, this approach results in a significant drop in performance, possibly because of the shift in feature distribution. We then added our proposed two modules to the low-rank model and observed performance improvements. By combining the two modules together, our model can often outperform other variants, demonstrating the effectiveness of the proposed method.

Performance variation *w.r.t.* compression ratio. Training large models often involves a trade-off between effectiveness and efficiency. To demonstrate this, we trained our model using different compression ratios with fewer epochs to simplify the process and present the results in Fig. 4. This graph indicates that a smaller compression ratio, *i.e.*, a larger model, typically yields better performance. However, a model that is too small, such as one that is compressed to 1/10 of its original size, may not be capable of achieving satisfactory results.

Table 7. Ablation studies of the proposed method over five tasks. For the downstream tasks of ALBEF, we selected representative evaluation metrics for space concerns.

Model	$\mathcal{L}_{fd}$	$\mathcal{L}_{rwp}$	DeiT		Swin		ALBEF					
			Cls	Seg	Cls	Seg	Retrieval		SNLI-VE		VG	
			Acc	mIoU	Acc	mIoU	TR@1	IR@1	val	test	TestA	TestB
Baseline			81.80	44.99	83.50	47.68	72.64	54.91	79.29	79.79	65.07	45.75
PELA	✗	✗	61.08	24.42	77.60	28.80	65.94	49.45	76.10	76.21	61.36	41.36
	✓	✗	80.90	43.67	82.89	47.28	70.66	54.55	78.35	78.07	65.70	44.90
	✗	✓	80.55	42.94	82.86	47.24	71.44	54.43	78.50	78.39	66.24	44.57
	✓	✓	80.96	43.24	82.54	47.21	71.26	54.75	78.55	78.66	65.86	45.10

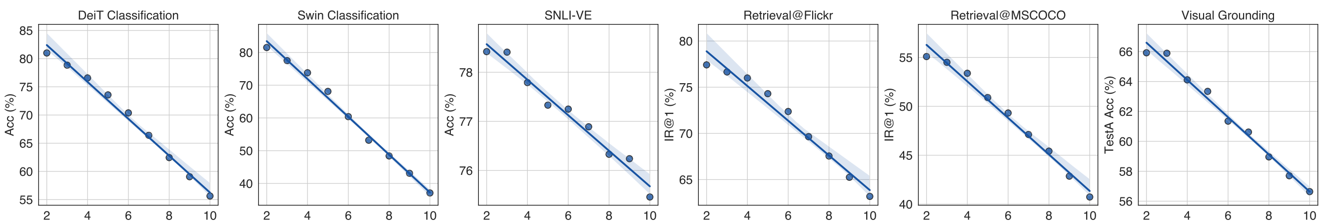

Figure 4. Model performance change *w.r.t.* compression ratios.

Table 8. Efficiency comparison of two pre-training strategies.

Method	Batch Size	GPU Mem↓	Latency↓
DeiT-Base	64×4	12.77 GB	76.24 ms/img
DeiT-Base$_{pela}$		11.34 GB	70.06 ms/img

4.5. Pre-training Efficiency & Model Scaling

Pre-training Efficiency. One may be concerned about the efficiency issues during pre-training. To address this problem, we leveraged the DeiT-Base model and evaluated its pre-training efficiency metrics, and show the results in Table 8. In particular, we employed the plain low-rank model because it already delivers promising model performance. Though other models may trigger longer training time, under this context, as shown in the table, our PELA method outperforms the original model in terms of both GPU memory cost and training latency.

Downstream Model Scaling. Our approach spawns a more compact model compared to the original large pre-trained one, resulting in a surplus of memory that enables us to train downstream models with larger batch sizes. To demonstrate the effectiveness of our method, we increased the batch size for both DeiT-Base and Swin-Base on the semantic segmentation task[2], as shown in Table 9. Our experiments show promising results, with a significant improvement in model performance for both models, achieving an absolute mIoU improvement of 0.57% and 0.78%, respectively. Moreover, our proposed method also outperforms the original Swin-Base baseline using PELA+, highlighting another advantage of our proposed approach.

[2]We kept the GPU memory less than the original large baseline model for a fair comparison.

Table 9. Model scaling performance of DeiT-Base and Swin-Base on semantic segmentation. PELA+ denotes the PELA model with a larger batch size while maintaining similar GPU memory.

Method	Batch Size	Baseline	PELA	PELA+
DeiT-Base	16→20	44.99	43.24	$43.81_{+.57}$
Swin-Base	16→20	47.68	47.21	$47.99_{+.78}$

5. Conclusion and Future Work

In this work, we propose a simple yet effective parameter-efficient pre-training approach that employs low-rank approximation as the core. Even with its simplicity, our method achieves competitive performance with baselines while attaining significantly improved parameter and computational efficiencies. These advantages enable model scaling in terms of model depth, width, and training batch size of downstream task fine-tuning. This work highlights the potential benefits of tackling the over-parameterization problem of learnable weights. In addition to this, we believe that the compression of intermediate features is a promising orthogonal direction for reducing model complexity. Therefore, we plan to investigate feature compression techniques, such as vision token pruning, to further build a more lightweight model in future research.

Acknowledgements: This research / project is supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

[1] Yogesh Balaji, Mohammadmahdi Sajedi, Neha Mukund Kalibhat, Mucong Ding, Dominik Stöger, Mahdi Soltanolkotabi, and Soheil Feizi. Understanding over-parameterization in generative adversarial networks. In *ICLR*, 2021. 2, 3

[2] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *ICLR*, 2023. 5, 6, 7

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, pages 6154–6162. IEEE, 2018. 6

[4] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *ICCV*, pages 357–366. IEEE, 2021. 6

[5] Patrick H. Chen, Si Si, Yang Li, Ciprian Chelba, and Cho-Jui Hsieh. Groupreduce: Block-wise low-rank approximation for neural language model shrinking. In *NeurIPS*, pages 11011–11021, 2018. 3

[6] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. UNITER: universal image-text representation learning. In *ECCV*, pages 104–120. Springer, 2020. 6, 7

[7] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *NIPS*, pages 3123–3131, 2015. 1, 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE, 2009. 5, 6

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186. ACL, 2019. 1, 2

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021. 1, 3, 6

[11] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. In *NeurIPS*, 2020. 7

[12] Jianbo Guo, Yuxi Li, Weiyao Lin, Yurong Chen, and Jianguo Li. Network decoupling: From regular to depthwise separable convolutions. In *BMVC*, page 248. BMVA Press, 2018. 3

[13] Yangyang Guo, Zhiyong Cheng, Jiazheng Jing, Yanpeng Lin, Liqiang Nie, and Meng Wang. Enhancing factorization machines with generalized metric learning. *TKDE*, 34 (8):3740–3753, 2022. 3

[14] Yangyang Guo, Liqiang Nie, Zhiyong Cheng, Qi Tian, and Min Zhang. Loss re-scaling VQA: revisiting the language prior problem from a class-imbalance view. *TIP*, 31:227–238, 2022. 6

[15] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *ICLR*, 2022. 3

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE, 2016. 6

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969. IEEE, 2017. 6

[18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 15979–15988. IEEE, 2022. 3, 6

[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, pages 173–182. ACM, 2017. 3

[20] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *ICCV*, pages 1921–1930. IEEE, 2019. 1, 2, 4

[21] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. 1, 2

[22] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*. OpenReview.net, 2022. 2, 3, 5, 6, 7

[23] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *JMLR*, 18:187:1–187:30, 2017. 1, 2

[24] Yerlan Idelbayev and Miguel Á. Carreira-Perpiñán. Low-rank compression of neural nets: Learning the rank of each layer. In *CVPR*, pages 8046–8056. IEEE, 2020. 3

[25] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, pages 709–727. Springer, 2022. 1, 2, 3, 5

[26] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling BERT for natural language understanding. In *Findings of EMNLP*, pages 4163–4174. ACL, 2020. 2, 5, 6, 7

[27] Roberto J. Bayardo Jr. Efficiently mining long patterns from databases. In *SIGMOD*, pages 85–93. ACM, 1998. 2

[28] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *CoRR*, 2020. 1

[29] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*, pages 5583–5594. PMLR, 2021. 7

[30] Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, Minghai Qin, and Yanzhi Wang. Spvit: Enabling faster vision transformers via latency-aware soft token pruning. In *ECCV*, pages 620–640. Springer, 2022. 2

[31] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*. OpenReview.net, 2020. 2, 3

[32] Sangho Lee, Youngjae Yu, Gunhee Kim, Thomas M. Breuel, Jan Kautz, and Yale Song. Parameter efficient multimodal transformers for video representation learning. In *ICLR*, 2021. 3

[33] Chong Li and C.-J. Richard Shi. Constrained optimization based low-rank approximation of deep neural networks. In *ECCV*, pages 746–761. Springer, 2018. 3

[34] Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Gotmare, Shafiq R. Joty, Caiming Xiong, and Steven Chu-Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *NeurIPS*, pages 9694–9705, 2021. 2, 6, 7

[35] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557, 2019. 7

[36] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*, pages 121–137. Springer, 2020. 7

[37] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: pushing the limit of post-training quantization by block reconstruction. In *ICLR*. OpenReview.net, 2021. 2

[38] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 6

[39] Xuejing Liu, Liang Li, Shuhui Wang, Zheng-Jun Zha, Dechao Meng, and Qingming Huang. Adaptive reconstruction network for weakly supervised referring expression grounding. In *ICCV*, pages 2611–2620. IEEE, 2019. 7

[40] 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 *ICCV*, pages 9992–10002. IEEE, 2021. 2, 5, 6

[41] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *CVPR*, pages 10434–10443. IEEE, 2020. 7

[42] Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *CoRR*, abs/2106.08962, 2021. 2

[43] Dimitris S. Papailiopoulos, Alexandros G. Dimakis, and Stavros Korokythakis. Sparse PCA through low-rank approximations. In *ICML*, pages 747–755. JMLR.org, 2013. 3

[44] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, pages 10428–10436. IEEE, 2020. 6

[45] Babak Rokh, Ali Azarpeyvand, and Alireza Khanteymoori. A comprehensive survey on model quantization for deep neural networks. *CoRR*, 2022. 2

[46] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015. 2, 4

[47] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. 2

[48] M. Schuermans, Philippe Lemmerling, and Sabine Van Huffel. Structured weighted low rank approximation. *Numerical Linear Algebra with Applications*, 11(5-6):609–618, 2004. 2

[49] Nathan Srebro and Tommi S. Jaakkola. Weighted low-rank approximations. In *ICML*, pages 720–727. AAAI Press, 2003. 2

[50] Trevor Strohman and W. Bruce Croft. Efficient document retrieval in main memory. In *SIGIR*, pages 175–182. ACM, 2007. 2

[51] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for BERT model compression. In *EMNLP*, pages 4322–4331. ACL, 2019. 1, 2, 5

[52] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. VL-ADAPTER: parameter-efficient transfer learning for vision-and-language tasks. In *CVPR*, pages 5217–5227. IEEE, 2022. 1, 2, 3, 5

[53] Cheng Tai, Tong Xiao, Xiaogang Wang, and Weinan E. Convolutional neural networks with low-rank regularization. In *ICLR*, 2016. 3

[54] Hao Tan and Mohit Bansal. LXMERT: learning cross-modality encoder representations from transformers. In *EMNLP*, pages 5099–5110. ACL, 2019. 6, 7

[55] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357. PMLR, 2021. 2, 5, 6

[56] Hugo Touvron, Matthieu Cord, Alaaeldin El-Nouby, Piotr Bojanowski, Armand Joulin, Gabriel Synnaeve, and Hervé Jégou. Augmenting convolutional networks with attention-based aggregation. *arXiv preprint arXiv:2112.13692*, 2021. 6

[57] Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *ECCV*, pages 516–533. Springer, 2022. 2, 5, 6

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 2

[59] Huan Wang, Can Qin, Yue Bai, Yulun Zhang, and Yun Fu. Recent advances on neural network pruning at initialization. In *IJCAI*, pages 5638–5645. ijcai.org, 2022. 2

[60] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. OFA: unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *ICML*, pages 23318–23340. PMLR, 2022. 1

[61] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768, 2020. 3

[62] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *CoRR*, 2022. 1

[63] Tsui-Wei Weng, Pu Zhao, Sijia Liu, Pin-Yu Chen, Xue Lin, and Luca Daniel. Towards certificated model robustness against weight perturbations. In *AAAI*, pages 6356–6363. AAAI Press, 2020. 5

[64] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, pages 418–434. Springer, 2018. 6

[65] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 1492–1500. IEEE, 2017. 6

[66] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *EMNLP*, pages 9514–9528. ACL, 2021. 2

[67] Yuhui Xu, Yuxi Li, Shuai Zhang, Wei Wen, Botao Wang, Yingyong Qi, Yiran Chen, Weiyao Lin, and Hongkai Xiong. TRP: trained rank pruning for efficient deep neural networks. In *IJCAI*, pages 977–983. ijcai.org, 2020. 3

[68] Hongwei Xue, Peng Gao, Hongyang Li, Yu Qiao, Hao Sun, Houqiang Li, and Jiebo Luo. Stare at what you see: Masked image modeling without reconstruction. In *CVPR*, pages 22732–22741. IEEE, 2023. 5, 6, 7

[69] Huanrui Yang, Minxue Tang, Wei Wen, Feng Yan, Daniel Hu, Ang Li, Hai Li, and Yiran Chen. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. In *CVPR Workshops*, pages 2899–2908. IEEE, 2020. 3

[70] Zhendong Yang, Zhe Li, Ailing Zeng, Zexian Li, Chun Yuan, and Yu Li. Vitkd: Practical guidelines for vit feature knowledge distillation. *CoRR*, abs/2209.02432, 2022. 2

[71] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *CVPR*, pages 67–76. IEEE, 2017. 3

[72] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, pages 558–567. IEEE, 2021. 6

[73] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL*, pages 1–9. ACL, 2022. 2

[74] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun. Accelerating very deep convolutional networks for classification and detection. *TPAMI*, 38(10):1943–1955, 2016. 3

[75] Zhu Zhang, Zhou Zhao, Zhijie Lin, Jieming Zhu, and Xiuqiang He. Counterfactual contrastive learning for weakly-supervised vision-language grounding. In *NeurIPS*, 2020. 7

[76] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *IJCV*, 127: 302–321, 2019. 6