Dense Vision Transformer Compression with Few Samples

Hanxiao Zhang*   Yifan Zhou   Guo-Hua Wang

National Key Laboratory for Novel Software Technology, Nanjing University, China

School of Artificial Intelligence, Nanjing University, China

{zhanghx, zhouyifan, wangguohua}@lamda.nju.edu.cn

Abstract

Few-shot model compression aims to compress a large model into a more compact one with only a tiny training set (even without labels). Block-level pruning has recently emerged as a leading technique in achieving high accuracy and low latency in few-shot CNN compression. But, few-shot compression for Vision Transformers (ViT) remains largely unexplored, which presents a new challenge. In particular, the issue of sparse compression exists in traditional CNN few-shot methods, which can only produce very few compressed models of different model sizes. This paper proposes a novel framework for few-shot ViT compression named DC-ViT. Instead of dropping the entire block, DC-ViT selectively eliminates the attention module while retaining and reusing portions of the MLP module. DC-ViT enables dense compression, which outputs numerous compressed models that densely populate the range of model complexity. DC-ViT outperforms state-of-the-art few-shot compression methods by a significant margin of 10 percentage points, along with lower latency in the compression of ViT and its variants.

1. Introduction

Vision Transformers (ViT) have demonstrated outstanding results on various vision tasks by leveraging the Transformer structure initially proposed for natural language processing (NLP). Unlike Convolutional Neural Networks (CNN), the self-attention mechanism allows ViT to learn global representations for images without an obvious inductive bias. However, with millions to billions of parameters, ViTs can only be deployed on high-end devices, even if we only consider the inference stage. To fit large models into small devices, network compression techniques are commonly adopted to reduce computing and memory costs.

Although some methods have been proposed to compress ViT through either token pruning or channel prun-

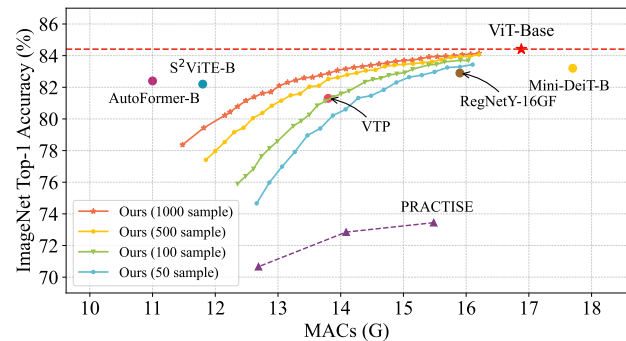*H. Zhang is the corresponding author.

Figure 1. Top-1 accuracy vs. MACs (G) on ImageNet. Our DC-ViT was compressed with few training images (50, 100, 500, or 1000), and PRACTISE was compressed with 500 images. However, other methods were not few-shot, using 1.28 million labeled training images during compression. It is quite a success that our DC-VIT uses only **less than 0.1% of their training images without labels**, but achieves slightly lower or even higher accuracy.

ing [4, 7, 26, 32, 36, 52], they all assume *access to the entire original training set*, which is unfortunately not the case in numerous situations, especially outside of the academia. When working with datasets containing large amounts of sensitive data (*e.g.*, medical or commercial), few-sample compression enables model pruning using few non-sensitive data and safeguards data privacy and security.

Few-shot compression has been studied for years, but existing methods focus solely on CNN compression. As Vision Transformers (ViT) is already a mainstream model, it is critical to design few-shot ViT compression methods. As shown in Sec. 4.2, existing few-shot CNN compression methods do *not* work well in few-shot ViT compression. The proposed DC-ViT method not only outperforms previous state-of-the-art (SOTA) methods in compressing ViTs, but also in few-shot compression of CNNs (*cf*. Fig. 1).

Another essential demand is that the method should be both highly accurate and dense, where *dense* means many models can be compressed in the target compression range, while a *sparse* method can only output few models (DC-ViT
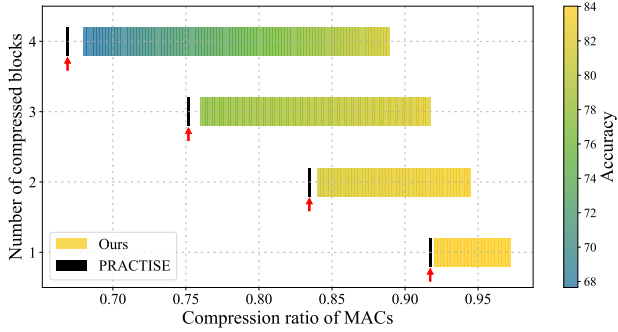
Figure 2. The range of compression options attainable by our DC-ViT and PRACTISE [43] by varying the number of compressed blocks. The black vertical lines represent the specific compression rates achieved by [43], highlighted by the red arrows. DC-ViT can output models densely in a wide range of compression rates for different numbers of blocks. The gradient color bar indicates the accuracy of our method at different compression rates.

vs. PRACTISE in Fig. 1). If a sparse compression method can only emit two compressed models with 80% and 110% of the target MACs the hardware can afford, we are forced to choose the 80% one, and 20% of the available computing power is wasted, which will lead to lower accuracy compared to a model that can use 100% of the compute budget. A *dense* compression method will naturally overcome this drawback.

PRACTISE [43] is a state-of-the-art few-shot CNN compression method, which outperforms other existing methods significantly in accuracy but only provides sparse compression options. As shown in Fig. 2, given the target range for MACs compression, PRACTISE can only produce 4 models by dropping different numbers of blocks. While existing filter-level few-shot CNN pruning methods might attain dense compression, they exhibit substantial accuracy degradation compared to block-level pruning [43].

In this paper, we propose a novel and effective framework for *dense* ViT compression under the few-shot setting. As shown in Fig. 2, our method (denoted by the gradient color bars) can produce compressed models of various sizes that densely cover the range of MACs. We also introduce a new metric to measure the importance of a block, which matches the actual performance of the *finetuned* model after compressing different blocks. To the best of our knowledge, the proposed Dense Compression of Vision Transformers (DC-ViT) is the first work in dense few-shot compression of both ViT and CNN.

As shown in Fig. 1, our DC-ViT offers much *denser* compression than other structured pruning methods, which means that for *any* target compression ratio within a certain range, we can always find one compression setting in DC-ViT that is close enough. It significantly outperforms the SOTA few-shot compression method PRACTISE by more

than *10 percentage points* in top-1 accuracy.

Moreover, Fig. 1 further shows comparison with compressed or neural architecture searched ViT/DeiT [41] and CNN models. Note they all used *1.28 million labeled training images* for compression, while DC-ViT used only $< 0.1\%$ *of their training images, which are even unlabeled.* It is a surprising success that with the same MACs, DC-ViT achieves higher Top-1 accuracy than Mini-DeiT-B [54], RegNetY-16GF [35] and VTP [56], and only slightly (2–4 percentage points) lower than AutoFormer-B [6] and S²ViTE-B [7].

Overall, our contributions are summarized as follows:
• We propose DC-ViT, an effective framework that pioneers few-sample ViT compression.
• We advocate the benefits and necessity of dense compression in practical applications, which is achieved by dropping the attention and adjusting the MLP drop ratio.
• We propose generating synthetic images as a metric dataset to address the potential misalignment between the perceived recovery difficulty (indicated by the training loss) and the actual performance of finetuned models after removing different blocks. We also summarize some suggestions for few-shot ViT compression, which have shown their advantages in DC-ViT and may be useful for future work in the area.

2. Related Works

Building upon its success in natural language processing, researchers have extended the application of Transformers to the computer vision domain, resulting in the ViT models [13]. Many variants of ViT have been proposed for a wide range of other computer vision tasks, including object detection [3, 57], semantic segmentation [50, 55], image super-resolution [49], image generation [5], and video understanding [15, 39]. However, the high computational cost limits its deployment, prompting further research into model compression techniques to address this challenge.

Pruning or compression has been extensively studied to reduce a network's complexity and to make it possible to migrate large networks to less powerful devices (*e.g.*, edge devices). Previous methods of CNN compression can be broadly grouped into two categories. Unstructured pruning [12, 22, 48] typically removes unimportant weights utilizing importance metrics such as the magnitude-based metrics [17] or Hessian-based ones [16, 18, 21]. By setting unimportant weights to zero, these methods achieve highly sparse but unstructured patterns for the network parameters. However, general-purpose GPUs do not support unstructured pruning, and their speed is in fact slow. Structured pruning solves this difficulty by completely removing some groups of parameters, such as filters [23, 47] or blocks [27, 43], where the key is to evaluate the importance of different modules by certain criteria to determine which

groups of parameters should be removed [43, 46].

As for ViT pruning, existing methods can be broadly divided into token pruning and channel pruning. Token pruning focuses on dynamically selecting important tokens for different inputs [26, 32, 36], while channel pruning targets searching optimal sub-networks by resorting to both structured and unstructured approaches [4, 7, 52].

Few-shot knowledge distillation considers knowledge distillation in the context of having only few training samples. Compression methods for deep neural networks typically require finetuning to recover the prediction accuracy when the compression ratio is high. However, conventional finetuning methods are hindered by the need for a large training set. FSKD [24] added a 1×1 convolution layer at the end of each layer of the student network and fit the layer-level output features of the student to the teacher. CD [2] proposed cross distillation to reduce accumulated errors caused by layer-wise knowledge distillation. In MiR [44], the teacher's layers before the penultimate are replaced with well-tuned compact ones. Unlike previous methods that adopt filter-level pruning, PRACTISE [43] was built on block-level pruning and can recover the network's accuracy using only tiny sets of training images.

Since few-shot compression of ViT has not been explored, while the current SOTA methods for ViT compression always falter in the few-shot context (as will be shown by our experiments), in this paper, we mainly compare our DC-ViT with the SOTA few-shot CNN pruning method.

3. Method

Now we propose Dense Compression of Vision Transformers (DC-ViT). The overall framework is illustrated in Fig. 3. Our strategy can be divided into three primary stages: 1) determine the compressed model's network structure based on the target compression ratio; 2) a block-trial procedure to find recoverability scores for different modules, and generate a synthetic metric set to select blocks with highest recoverability; 3) a progressive approach to prune and finetune the model using few-shot and unlabeled training set.

3.1. Determining the Network Structure

For few-shot learning, removing one entire block leads to a change that can be well approximated by a linear layer (which can be absorbed) [43]. Furthermore, block pruning leads to significantly higher throughput than filter pruning, thus it is beneficial to remove blocks completely. Nevertheless, while the removal of a single block can be *approximately* compensated by linear adjustments, the cumulative errors from dropping multiple blocks can escalate to a nonlinear magnitude. Hence, we propose to *replace a dropped block with a small MLP* to remedy such errors.

The structure of ViT makes our remedy particularly suitable. As pointed out by [45], the attention module takes up roughly 46% of a Transformer block's latency. After we remove the attention module, the remaining part of a block is exactly an MLP; by pruning this MLP to a smaller one (as in the left side of Fig. 3), we can not only *achieve dense compression*, but also *reuse parameters* from the original MLP. Our experiments highlighted the significance of parameter reusing in few-shot compression (*cf*. Tab. 6). Therefore, in our *dense* compression process, we need to determine two things based on the target MACs: the number of blocks to remove and the extent to which the MLPs within these blocks should be reduced.

We calculate the minimum number of blocks that should be compressed to meet the target, denoted as k, *i.e.*,

$$k = \left\lceil \frac{\mathrm{MACs}_o - \mathrm{MACs}_p}{\mathrm{MACs}_a + \mathrm{MACs}_m} \right\rceil , \qquad (1)$$

where MACs_o and MACs_p respectively denote MACs of the origin and pruned models ($\mathcal{M}_O$ and $\mathcal{M}_P$), MACs_a denotes MACs of the attention module *including* the corresponding layer norm, while MACs_m refers to MACs of the MLP module *excluding* the corresponding layer norm of one ViT block, respectively. $\lceil \cdot \rceil$ is the ceiling function.

Minimize the number of compressed blocks. As shown in Fig. 2, one target MACs can be obtained by removing different numbers of blocks. We can choose to prune more blocks but retain a larger ratio of their corresponding MLPs. Alternatively, we can also prune fewer blocks but keep a smaller part of the MLPs in these blocks. For example, to keep 85% of the original model's MACs, we can throw away 2, 3, or 4 blocks, as shown in Fig. 2. However, the color bars, which represent the accuracy of the compressed models, clearly indicate that minimizing the number of compressed blocks is preferable when multiple compression configurations are feasible.

To further simplify our method, we allocate the *same* drop ratio into every MLP module of the k compressed blocks. Hence, the drop ratio of MLP nodes r_d will be

$$r_d = \frac{\mathrm{MACs}_o - \mathrm{MACs}_p - k \cdot \mathrm{MACs}_a}{k \cdot \mathrm{MACs}_m} . \qquad (2)$$

The MLP in a ViT block has 2 fully connected (FC) layers, with the dimension of features changing from d to $4d$ and then back to d again. We change the dimensionality of the intermediate result from $4d$ to $4d(1 - r_d)$, which finalizes the compression structure. Our ablation studies show that randomly picking $4d(1 - r_d)$ nodes out of $4d$ works surprisingly well, thus DC-ViT adopts this simple random strategy.

3.2. Block Selection

In order to select which blocks are to be compressed, we adopt a block-wise trial procedure to obtain candidate models. Next, we generate synthetic images as a tiny metric set
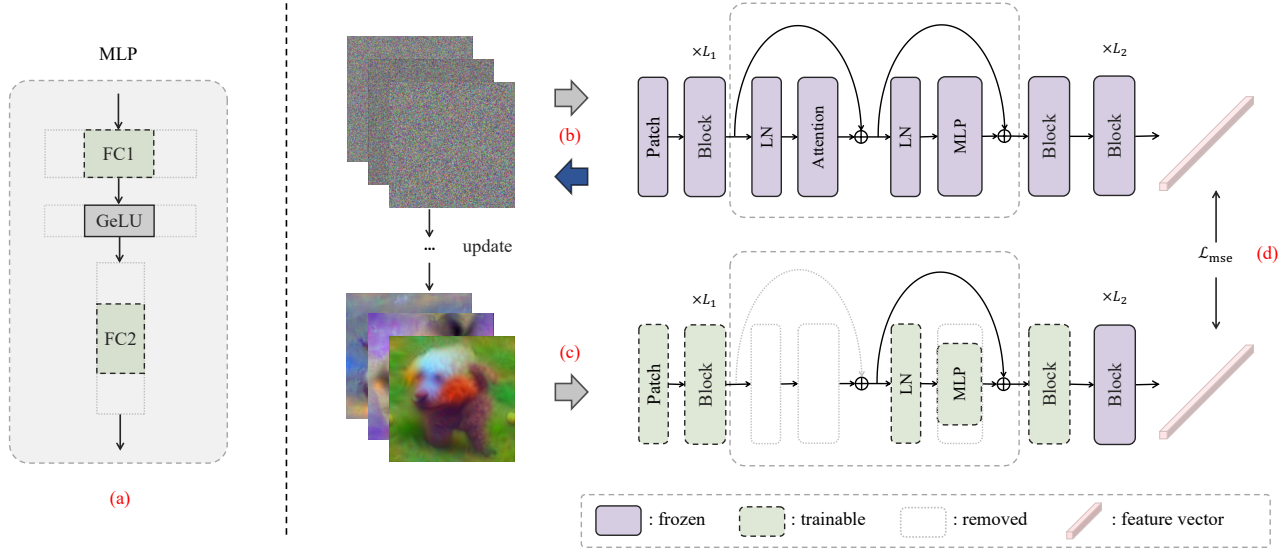
Figure 3. The DC-ViT framework. (a) Determining the network structure to achieve the target MACs reduction. (b) Generating a synthetic metric set from Gaussian noise. (c) Using the synthetic metric set to select the blocks with highest recoverability. (d) Progressive pruning and finetuning. The original model is at the top, and the pruned model is at the bottom. We drop the whole attention module but reuse part of the MLP, and use the MSE loss for feature mimicking, but *only* update the front part of blocks till the next block of the compressed one.

to evaluate the recoverability of different blocks after compression.

3.2.1 Block-wise Trial

Once the structure parameters (k and r_d) are determined, we operate a block-wise trial procedure to obtain candidate models $\mathcal{M}_{P(\mathcal{B}_i)}$ with block $\mathcal{B}_i$ compressed, then use the criterion proposed in the following section to calculate the recoverability scores of different blocks.

For a few-shot task with C classes, let $\mathcal{D}_\mathcal{T}$ denote the tiny subset of the original training set *without* labels. In some applications, not only is the complete training set unavailable, but also labels for the accessible tiny set. Therefore, unsupervised few-shot compression is preferred.

In the block-wise trial step, we obtain candidate models $\mathcal{M}_{P(\mathcal{B}_i)}$ with block $\mathcal{B}_i$ replaced by a small MLP, as described in Sec. 3.1. Following MiR [44], to finetune $\mathcal{M}_{P(\mathcal{B}_i)}$, we use the mean squared error (MSE) loss for feature mimicking to minimize the feature gap between the pruned and original models, *i.e.*,

$$\mathcal{L}_{MSE}\left(\mathcal{M}_O, \mathcal{M}_P, \mathcal{D}_\mathcal{T}\right) = \sum_{x \in \mathcal{D}_\mathcal{T}} \|\mathcal{M}_O\left(x\right) - \mathcal{M}_P\left(x\right)\|_F^2,$$
(3)

where $\mathcal{M}_P\left(x\right)$ and $\mathcal{M}_O\left(x\right)$ denote the *union* of all output tokens of the pruned and the original model, respectively. Even though the CLS token is generally considered sufficient as the representation of an image, conducting feature mimicking using all output tokens works *better* than just focusing on the CLS token, as will be shown in Tab. 6. We

also find that by *updating only the first few blocks up to the last compressed one*, instead of all blocks, not only improves the accuracy of finetuned models but also reduces the time cost of gradient back-propagation. As shown in Fig. 3, only the blocks in the yellow color are trainable. Our ablation studies in Tab. 6 back up this strategy.

3.2.2 Generate a Synthetic Metric Set

Since the training data is extremely limited in few-shot compression, various few-shot data augmentation methods exist in the few-shot learning domain. Their primary objective is to produce augmented/synthetic images that align with the true data distribution using limited training samples to increase the sample richness. Some rely on hand-crafted rules from experts with specialized knowledge [1, 8, 10, 11, 14, 20], while a surge of probabilistic-based [25] and generative-based [9, 38, 53] approaches have emerged for automatic synthetic image generation with the maturity of meta-learning.

Operating under the assumption that a deep neural network can sufficiently train and retain crucial information from a dataset, we propose to utilize the original pre-trained model to generate synthetic images as the metric set $\mathcal{S}$ for block selection without using external data. As will be illustrated in Sec. 3.2.3, the metric set is used to reveal the performance of the finetuned candidate models.

Following DeepDream [30], we leverage the pre-trained model's intrinsic knowledge to efficiently generate a compact batch of synthetic images, forming a robust metric dataset. Given a batch of Gaussian noise as the initial syn-

Latency (ms)	Block	Attn	FFN	Attn+0.5·FFN
Mean	102.95	107.70	111.12	103.93
Std.	0.44	0.11	0.08	0.12

Table 1. Mean and standard deviation of latency in ViT-Base when different parts are removed. "Attn": the attention module, "FFN": the feed-forward network (MLP), and "Attn+0.5·FFN": the attention module plus half of MLP.

thetic images $\mathcal{S} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_n\}$ and arbitrary target labels $\mathcal{Y} = \{y_1, y_1, \cdots, y_n\}$, the synthetic images are updated by minimizing

$$\mathcal{L}_\mathcal{S} = \min_\mathcal{S} \frac{1}{|\mathcal{S}|} \sum_{i=1}^{|\mathcal{S}|} \left(\mathcal{L}(\hat{x}_i, y_i) + \mathcal{R}(\hat{x}_i)\right), \quad (4)$$

where $\mathcal{L}(\cdot)$ is the cross entropy between the predicted probability and the target label, and $\mathcal{R}(\cdot)$ is the regularization term. We use the image prior regularizer of ℓ_2 and total variation (TV) proposed in [29]:

$$\mathcal{R}(\hat{x}) = \alpha_{\ell_2}\mathcal{R}_{\ell_2}(\hat{x}) + \alpha_{\text{tv}}\mathcal{R}_{\text{TV}}(\hat{x}), \quad (5)$$

where $\mathcal{R}_{\ell_2} = \|\hat{x}\|^2$ is utilized to promote image stability and $\mathcal{R}_{\text{TV}} = \sum_{i,j} \left((\hat{x}_{i,j+1} - \hat{x}_{ij})^2 + (\hat{x}_{i+1,j} - \hat{x}_{ij})^2\right)^{\frac{\beta}{2}}$ can control the sharpness of the synthetic data.

Since the metric loss $\mathcal{L}_{CE}$ in Eq. (6) does not require a class label, the randomly assigned target y in Eq. (4) only acts as a hint for the pre-trained model to generate synthetic data. After a few iterations, the synthetic images are able to capture the distribution of the original dataset (*cf*. the appendix).

3.2.3 Choose the Blocks to Compress

Since all ViT blocks in the same model are structurally identical, there is little latency variance caused by compressing different blocks in DC-ViT (*cf*. Tab. 1). Therefore, the score we propose for block selection is a simple one that evaluates the performance of candidates on the synthetic metric set $\mathcal{S}$ generated as in Sec. 3.2.2,

$$\mathcal{L}_{CE}\left(\mathcal{M}_O, \mathcal{M}_{P(\mathcal{B}_i)}, \mathcal{S}\right) = \\ -\sum_{x \in \mathcal{S}} \sum_{c=1}^{C} p\left(\mathcal{M}_O(x); c\right) \log\left(p\left(\mathcal{M}_{P(\mathcal{B}_i)}(x); c\right)\right), \quad (6)$$

where $p\left(\mathcal{M}_{P(\mathcal{B}_i)}(x); c\right)$ denotes the probability of training sample x is classified into the c-th class by the i-th candidate model, which is calculated using the output CLS token, followed by linear and softmax transformations, akin to $p\left(\mathcal{M}_O(x); c\right)$.

Synthetic images bridge the gap between perceived and actual block recoverability. As illustrated in Fig. 4, the
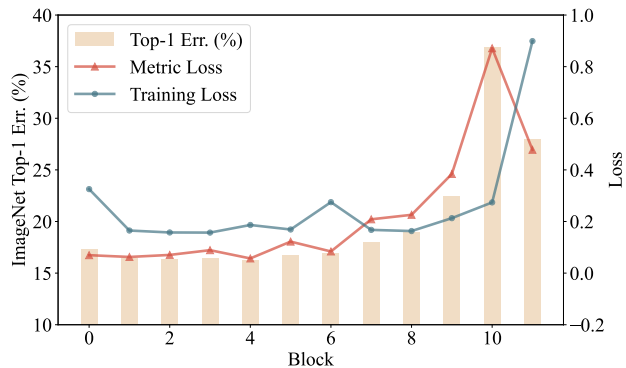

Figure 4. The top-1 error of 12 candidate models (by compressing 12 ViT blocks one at a time) on the test dataset and different criteria for block selection. The top-1 error is calculated using the original test set. The metric loss is calculated using the synthetic metric set $\mathcal{S}$. The training loss is calculated using the tiny training set $\mathcal{D}_\mathcal{T}$. We scaled them to the same range.

top-1 error of candidate models on the test dataset and the metric loss on the synthetic metric set $\mathcal{S}$ are highly correlated. Conversely, the training loss on the tiny training set $\mathcal{D}_\mathcal{T}$ shows no such correlation, even though with different data augmentation from the finetuning phase, and this disparity becomes more evident with fewer training samples. We attribute this discrepancy to overfitting on the limited training set. Hence, the use of synthetic images as a metric dataset can address the potential misalignment between the perceived difficulty in model performance recovery (as indicated by the training loss) and the actual performance of finetuned models after removing different blocks.

Evaluating the recoverability directly using the finetuned candidates has two advantages. The performance of candidates consistent with the final structure reflects the recoverability of removing different blocks more intuitively. Furthermore, these candidate models can be reused in subsequent compression without being repeatedly finetuned.

3.3. Progressive Pruning and Finetuning

Our findings indicate that to compress more than one block, progressive compression—compressing blocks sequentially, one after the other—is more effective. Since few-shot compression is a delicate process, it's beneficial to minimize extensive modifications to the model structure in a single stage. Adopting a gradual approach to compression not only mitigates potential risks but also demonstrates superior performance, as will be shown in Tab. 6.

We provide the pseudo-code for DC-ViT in Algorithm 1. In the beginning, we determine the compressed network's structure. Then, we sequentially trial-removed different blocks in the original model $\mathcal{M}_O$ to create candidates $\mathcal{M}_{P(\mathcal{B}_i)}$ and finetune the candidates with the tiny training

Algorithm 1 DC-ViT

Input: original model $\mathcal{M}_O$, tiny training set $\mathcal{D}_\mathcal{T}$, synthetic metric set $\mathcal{S}$, number of blocks to be compressed k, MLP drop ratio r_d

Output: pruned model $\mathcal{M}_P$

1: $\mathcal{M}_P \leftarrow \mathcal{M}_O$
2: $mList \leftarrow$ empty list
3: **for** each block $\mathcal{B}_i$ of $\mathcal{M}_P$ **do**
4:      $\mathcal{M}_{P(\mathcal{B}_i)} \leftarrow$ drop the attention and (randomly) r_d of MLP nodes from $\mathcal{B}_i$ ▷ trial-compress single block
5:      Minimize $\mathcal{L}_{MSE}\left(\mathcal{M}_O, \mathcal{M}_{P(\mathcal{B}_i)}, \mathcal{D}_\mathcal{T}\right)$ to update $\mathcal{M}_{P(\mathcal{B}_i)}$ ▷ finetune corresponding model
6:      $\mathcal{L}_m \leftarrow \mathcal{L}_{CE}\left(\mathcal{M}_O, \mathcal{M}_{P(\mathcal{B}_i)}, \mathcal{S}\right)$ ▷ metric loss
7:      Add pair $(\mathcal{L}_m, i)$ to $mList$
8: **end for**
9: Sort $mList$ based on the metric loss of each block
10: $C \leftarrow$ first k indices of $mList$ ▷ blocks to compress
11: Progressively compress $\mathcal{B}_{C[:]}$ in $\mathcal{M}_P$ ▷ line 5-6
12: **return** $\mathcal{M}_P$

Batch size	64	Epochs	2000
Warmup epochs	20	Weight decay	$1e-4$
Optimizer	AdamW	Learning rate	$3e-5$
Learning rate decay	cosine	Gradient Clip.	1.0
Drop path rate	0.5	Drop rate	0.0
Rand Augment	9/0.5	Color Jitter	0.3

Table 2. Hyper-parameters of our DC-ViT.

set $\mathcal{D}_\mathcal{T}$. We generate a synthetic metric set to evaluate the rank of blocks' recoverability. Finally, we select the k blocks with the smallest metric values and progressively compress the selected blocks in $\mathcal{M}_O$ to obtain the pruned model $\mathcal{M}_P$.

4. Experiments

In this section, we report results across various architectures, followed by ablation studies to validate the efficacy of our *dense* compression method. Additionally, we offer some suggestions for few-shot ViT compression.

4.1. Experimental Settings

Architectures. We used the vanilla ViT models initially introduced by Dosovitskiy et al. [13], the smaller and the larger ViT variants proposed by Touvron et al. [41], DeiT-B [41] and Swin-B [28]. We also tested our method on CNN architectures, using ResNet-34 and MobileNetV2 as the backbone. Finally, we demonstrate the transferability of the compression models on downstream tasks.

Benchmark data. We randomly sampled tiny training sets from ImageNet-1k [37] and evaluated the top-1 / 5 accuracy of the pruned models on the validation set. When we reported the mean accuracy with the standard deviation, we ran the experiments 5 times with 5 random seeds (2021 $\sim$ 2025); otherwise, we took 2023 as the random seed.

Training procedure. Tab. 2 displays the default hyper-parameter setting we used at training time for all our experiments. Unless otherwise specified, the experiments were conducted at 224^2 resolution.

4.2. Baseline Pruning Methods

To the best of our knowledge, the field of few-shot compression for ViTs is currently unexplored. Previous ViT compression methods always rely on the complete ImageNet-1k dataset and often falter in the few-shot context. For example, S^2PViT [7], EViT [26], are the SOTA ViT compression methods that achieve excellent performance on DeiT. However, when we applied them to a few-shot setting and used 1000 training samples (which is already a large training set in few-shot compression), *neither of them converged.*

Moreover, most methods for few-shot compression of CNNs involve specific operations on convolutional layers, which are not easily transferable to ViT architectures. Therefore, we adopt the SOTA few-shot CNN pruning method, PRACTISE [43], which can be directly extended to ViT as a strong baseline. To further verify the effectiveness of our DC-ViT, we also extend DC-ViT to CNN architectures and benchmark its performance against notable few-shot CNN pruning methods, including FSKD [51], CD [2] and MiR [44]. When comparing with other few-shot pruning methods, we are mainly concerned about the latency-accuracy tradeoff of the pruned model. The latency of the pruned models is measured using a batch size of 64 and recorded the average inference time over 500 runs.

4.3. DC-ViT Performance

We demonstrate our DC-ViT performance on various ViT architectures, including plain ViT [13], DeiT [41] and Swin [28], as well as on CNN architectures.

4.3.1 Results on ViT-Base

We primarily compared our DC-ViT method with the SOTA few-shot CNN pruning method PRACTISE [43] on ViT-B.

Implementation details. We set the MACs of the pruned models in DC-ViT to be equal to the cases where the baseline drops 1, 2, or 3 entire blocks respectively, and the number of training samples was set to 50, 100, 500 and 1000. Moreover, we ran the baseline for $2000 \cdot k$ and 2000 epochs respectively for a fair comparison, because DC-ViT takes a progressive compression strategy and the finetuning step costs 2000 epochs in each iteration.

As shown in Tab. 3, under the same MACs settings, compared with removing complete blocks, our *dense* com-

MACs	Methods	Latency (ms)	50 sample	100 sample	500 sample	1000 sample
15.5×10^9 ($k=1$, 8.3% ↓)	PRACTISE*	103.2 (6.5% ↓)	$71.28_{\pm0.24}$	$75.69_{\pm6.77}$	$73.32_{\pm0.14}$	$74.71_{\pm0.13}$
	DC-ViT-B	**101.9 (8.3% ↓)**	$\mathbf{82.88_{\pm0.10}}$	$\mathbf{83.30_{\pm0.17}}$	$\mathbf{83.73_{\pm0.14}}$	$\mathbf{83.78_{\pm0.04}}$
14.1×10^9 ($k=2$, 16.6% ↓)	PRACTISE*	94.7 (15.4% ↓)	$69.70_{\pm0.32}$	$73.75_{\pm5.91}$	$72.54_{\pm0.31}$	$74.52_{\pm0.22}$
	PRACTISE†	94.7 (15.4% ↓)	$69.87_{\pm0.48}$	$74.11_{\pm6.01}$	$72.74_{\pm0.12}$	$74.69_{\pm0.10}$
	DC-ViT-B	**93.5 (16.7% ↓)**	$\mathbf{80.59_{\pm0.17}}$	$\mathbf{81.58_{\pm0.09}}$	$\mathbf{82.74_{\pm0.06}}$	$\mathbf{83.15_{\pm0.08}}$
12.7×10^9 ($k=3$, 24.8% ↓)	PRACTISE*	85.9 (23.5% ↓)	$63.75_{\pm0.44}$	$67.58_{\pm4.42}$	$70.03_{\pm0.56}$	$72.91_{\pm0.04}$
	PRACTISE†	85.9 (23.5% ↓)	$65.07_{\pm0.49}$	$69.60_{\pm4.96}$	$70.54_{\pm0.12}$	$73.28_{\pm0.03}$
	DC-ViT-B	**84.9 (24.5% ↓)**	$\mathbf{74.70_{\pm0.81}}$	$\mathbf{77.03_{\pm0.26}}$	$\mathbf{80.18_{\pm0.37}}$	$\mathbf{81.60_{\pm0.10}}$

Table 3. Top-1 validation accuracy (%) on ImageNet-1k for pruning ViT-Base with 50, 100, 500, 1000 samples. * means the method is implemented by us, † means the epoch was set to $2000 \cdot k$ (k is the number of compressed blocks) for a fair comparison. ↓ means the percentage of reduction. The accuracy and latency of the original ViT-Base are 84.41% and 111.1 ms, respectively.

Methods	Latency (ms)	Top-1 / 5 acc (%)
ResNet-34	34.4	73.3/91.4
FSKD [24]	29.0(15.8% ↓)	45.3/71.5
CD [2]	29.0(15.8% ↓)	56.2/80.8
MiR [44]	29.0(15.8% ↓)	64.1/86.3
PRACTISE [43]	29.0(15.8% ↓)	70.3/89.6
Ours	**28.8(16.5% ↓)**	**72.2/90.7**
MobileNetV2	23.2	71.9/90.3
FSKD [24]	19.4(16.2% ↓)	48.4/73.9
MiR [44]	19.4(16.2% ↓)	67.6/87.9
PRACTISE [43]	**18.8(19.1% ↓)**	69.3/88.9
Ours	**18.8(19.1% ↓)**	**69.8/89.2**

Table 4. Top-1 validation accuracy (%) on ImageNet-1k for pruning Resnet-34 and MobileNetV2. Previous few-shot pruning methods and our DC-ViT are used to prune Resnet34 and MobileNetV2 with 50 training samples.

Model	Methods	Latency (ms)	Top-1 / 5 acc (%)
ViT-T	Original	20.5	75.33 / 92.81
	PRACTISE†	17.4 (15.1% ↓)	54.88 / 82.43
	DC-ViT-T	**17.2 (16.5% ↓)**	**64.42 / 86.65**
ViT-S	Original	42.9	81.38 / 96.07
	PRACTISE†	36.5 (14.9% ↓)	69.18 / 90.42
	DC-ViT-S	**36.1 (16.7% ↓)**	**78.64 / 94.79**
ViT-L	Original	341.9	85.68 / 97.74
	PRACTISE†	317.0 (7.3% ↓)	83.99 / 97.12
	DC-ViT-L	**314.3 (8.2% ↓)**	**85.49 / 97.63**
DeiT-B	Original	110.2	81.74 / 95.58
	PRACTISE†	94.2 (14.5% ↓)	79.30 / 94.31
	DC-DeiT-B	**93.6 (16.7% ↓)**	**81.26 / 95.35**
Swin-B	Original	155.9	84.47 / 97.46
	PRACTISE†	135.6 (13.0% ↓)	82.88 / 96.76
	DC-Swin-B	135.6 (13.0% ↓)	**83.82 / 97.09**

Table 5. Top-1 / 5 validation accuracy (%) on ImageNet-1k for pruning various ViT architectures with 500 samples. † means the method is implemented by us. ↓ means percentage of reduction.

pression method by dropping the attention module and part of the MLP module always gains lower latency, especially when the number of compressed blocks and training samples increases, and our DC-ViT approach outperforms the baseline by a significant margin, achieving roughly 10 percentage points higher top-1 accuracy.

4.3.2 Results on CNN Architectures

Few-shot compression on ViT remains under-explored in existing literature, thus we sought to further validate the efficacy of our approach by extending it to CNN architectures and benchmark its performance against few-shot CNN pruning methods FSKD [51], CD [2], MiR [44]. We used ResNet-34 and MobileNetV2 as the backbone and use 50 training samples. Since the structure of CNN doesn't facilitate the reuse of MLP nodes for *dense* compression like DC-ViT, we tested the cases where our method eliminates entire blocks. As shown in Tab. 4, DC-ViT still outperforms previous methods by a significant margin.

4.3.3 Results on Various ViT Variants

We have further adapted our DC-ViT framework to encompass variants of ViT models, including DeiT-Base and Swin-Base. To ensure a fair and meaningful comparison with PRACTISE [43], we uniformly set the MACs of the pruned models in DC-ViT to match those in scenarios where PRACTISE drops 2 entire blocks. However, due to the complex structure of Swin, we directly select and eliminate two pairs of successive Swin blocks, *i.e.*, a total of 4 complete blocks. Tab. 5 demonstrates that DC-ViT consistently outperforms other methods across all these ViT architectures.

4.4. Ablation for Few-shot ViT Compression

We designed ablation studies to provide further insights into our framework. The results are shown in Tab. 6, based on

MACs	15% ↓ ($k=2$)	30% ↓ ($k=4$)
DC-DeiT-B	**83.25**	**79.28**
top-k (2000 epochs)	82.84	67.95
top-k ($2000 \cdot k$ epochs)	83.05	78.50
finetune the entire ViT	82.66	78.17
compress $(k+1)$ blocks	82.84	75.07
re-initialize MLP weights	82.60	77.29
GeLU-based weights reuse	83.07	79.25
metric with MSE loss	74.02	69.22
mimicking on class token	82.73	78.33

Table 6. Ablation of alternative settings in DC-ViT. We pruned ViT-Base with 500 samples to achieve 15% and 30% MACs reduction respectively.

Model	CIFAR-10	CIFAR-100	Flowers	Pets	CUB-200	Indoor67
EfficientNet-B7 [40]	98.9	91.7	98.8	-	-	-
ViT-B [13]	97.4	88.0	98.5	92.9	84.1	83.1
DC-ViT-B	98.0	87.5	99.3	92.2	83.5	82.7

Table 7. Transfer learning tasks' performance of DC-ViT-B with 24.8% latency reduction after few-shot compression. We report the top-1 accuracy of finetuning the original ViT-B model and our DC-ViT-B on downstream classification datasets.

which we provide some suggestions for compressing ViT when sample availability is limited.

Use progressive pruning procedure. As the top-k compression strategy, we finetuned for 2000 and $2000 \cdot k$ epochs respectively for a fair comparison. Either way, the progressive compression strategy achieves higher accuracy, especially when more blocks are compressed.

Update the front part of blocks. When finetuning the entire ViT, the accuracy decreased in all compression settings, especially when more blocks are compressed, which means our partial finetuning approach is not only efficient but also effective in few-shot contexts.

Reuse MLP weights. If the small MLP's weights are not reused but randomly re-initialized for finetuning, we can find that the more blocks are compressed, the more accuracy drops. In addition, we also tried some other ways of selecting the MLP nodes to be reused. However, they did not show any obvious advantage over random selection.

Compress as few blocks as possible. Given a specific compression ratio, if various options are available, the accuracy tends to decrease more drastically when more blocks are compressed. Therefore, we believe that it is better to compress as few blocks as possible so long as the target compression ratio is satisfied, which is also evidenced by Fig. 2.

We also tried some other alternative settings, such as using MSE loss to choose candidate models and conducting feature mimicking on the CLS token, but none of them achieved better performance than our default settings.

4.5. Transfer Learning to Downstream Tasks

Finally, we evaluate the generalizability of our DC-ViT.
Implementation details. We finetuned our DC-ViT-B model compressed in Sec. 4.3.1 on a collection of commonly used recognition datasets: CIFAR-10 and CIFAR-100 [19], Flowers [31], Oxford-IIIT Pets [33], CUB-200 [42] and Indoor67 [34]. During the finetuning phase, an AdamW optimizer was adopted with batch size 256, learning rate $5 \times 10^{-4} \times \frac{\text{batchsize}}{512}$.

Tab. 7 shows the results of the Top-1 accuracy of the original ViT-B and the pruned DC-ViT-B on downstream datasets. Compared to the SOTA ConvNets and transformer-based models, the pruned model achieved comparable or even better performance on all downstream classification tasks, which shows that the efficiency demonstrated on ImageNet can be preserved on downstream tasks by our few-shot compression method.

5. Conclusions

In this paper, we proposed a novel framework, Dense Compression of Vision Transformers (DC-ViT), for few-sample compression on Vision Transformers, which achieves not only much denser compression options in terms of MACs by dropping the attention module and adjusting the MLP drop ratio but also a better performance of the pruned models with higher accuracy and lower latency.

Limitations and Future Works. One common limitation of DC-ViT and other few-shot compression methods is that the ratio of compressed MACs cannot be too large (*e.g.*, mostly ≤ 30%), otherwise, the compressed model's accuracy will drop dramatically. Since DC-ViT introduces small MLPs into pruned blocks, it has the potential to partially solve this difficulty in the future. Another limitation is that few-shot compression is mainly restricted to recognition. However, tasks like detection and segmentation are more intriguing. We plan to test the effect of DC-ViT on ViT-based detector compression in future works.

Acknowledgements

References

[1] Amit Alfassy, Leonid Karlinsky, Amit Aides, Joseph Shtok, Sivan Harary, Rogério Schmidt Feris, Raja Giryes, and Alexander M. Bronstein. Laso: Label-set operations networks for multi-label few-shot learning. In *CVPR*, pages 6548–6557, 2019. 4

[2] Haoli Bai, Jiaxiang Wu, Irwin King, and Michael R. Lyu. Few shot network compression via cross distillation. In *AAAI*, pages 3203–3210, 2020. 3, 6, 7

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 2

[4] Arnav Chavan, Zhiqiang Shen, Zhuang Liu, Zechun Liu, Kwang-Ting Cheng, and Eric P. Xing. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *CVPR*, pages 4921–4931, 2022. 1, 3

[5] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, pages 12299–12310, 2021. 2

[6] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. In *ICCV*, pages 12250–12260, 2021. 2

[7] Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. In *NeurIPS*, pages 19974–19988, 2021. 1, 2, 3, 6

[8] Zitian Chen, Yanwei Fu, Yu-Xiong Wang, Lin Ma, Wei Liu, and Martial Hebert. Image deformation meta-networks for one-shot learning. In *CVPR*, pages 8680–8689, 2019. 4

[9] Zitian Chen, Yanwei Fu, Yinda Zhang, Yu-Gang Jiang, Xiangyang Xue, and Leonid Sigal. Multi-level semantic feature augmentation for one-shot learning. *IEEE Trans. Image Process.*, 28(9):4594–4605, 2019. 4

[10] Wen-Hsuan Chu, Yu-Jhe Li, Jing-Cheng Chang, and Yu-Chiang Frank Wang. Spot and learn: A maximum-entropy patch sampler for few-shot image classification. In *CVPR*, pages 6251–6260, 2019. 4

[11] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 4

[12] Xin Dong, Shangyu Chen, and Sinno Jialin Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *NeurIPS*, pages 4857–4867, 2017. 2

[13] 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*, 2021. 2, 6, 8

[14] Hang Gao, Zheng Shou, Alireza Zareian, Hanwang Zhang, and Shih-Fu Chang. Low-shot learning via covariance-preserving adversarial augmentation networks. In *NeurIPS*, pages 983–993, 2018. 4

[15] Rohit Girdhar, João Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *CVPR*, pages 244–253, 2019. 2

[16] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural network. In *NeurIPS*, pages 1135–1143, 2015. 2

[17] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *ICLR*, 2016. 2

[18] Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *NeurIPS*, pages 164–171, 1992. 2

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 8

[20] Roland Kwitt, Sebastian Hegenbart, and Marc Niethammer. One-shot learning of scene locations via feature trajectory transfer. In *CVPR*, pages 78–86, 2016. 4

[21] Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *NeurIPS*, pages 598–605, 1989. 2

[22] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Snip: single-shot network pruning based on connection sensitivity. In *ICLR*, 2019. 2

[23] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017. 2

[24] Tianhong Li, Jianguo Li, Zhuang Liu, and Changshui Zhang. Few sample knowledge distillation for efficient network compression. In *CVPR*, pages 14627–14635, 2020. 3, 7

[25] Yonggang Li, Guosheng Hu, Yongtao Wang, Timothy Hospedales, Neil M Robertson, and Yongxin Yang. Dada: Differentiable automatic data augmentation. *arXiv preprint arXiv:2003.03780*, 2020. 4

[26] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Evit: Expediting vision transformers via token reorganizations. In *ICLR*, 2022. 1, 3, 6

[27] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David S. Doermann. Towards optimal structured CNN pruning via generative adversarial learning. In *CVPR*, pages 2790–2799, 2019. 2

[28] 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, 2021. 6

[29] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, pages 5188–5196, 2015. 5

[30] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks, 2015. 4

[31] Maria-Elena Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *CVPR*, pages 1447–1454, 2006. 8

[32] Bowen Pan, Rameswar Panda, Yifan Jiang, Zhangyang Wang, Rogério Feris, and Aude Oliva. Ia-red^2: Interpretability-aware redundancy reduction for vision transformers. In *NeurIPS*, pages 24898–24911, 2021. 1, 3

[33] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, pages 3498–3505, 2012. 8

[34] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, pages 413–420, 2009. 8

[35] Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, pages 10425–10433, 2020. 2

[36] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, pages 13937–13949, 2021. 1, 3

[37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015. 6

[38] Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogério Schmidt Feris, Raja Giryes, and Alexander M. Bronstein. Delta-encoder: an effective sample synthesis method for few-shot object recognition. In *NeurIPS*, pages 2850–2860, 2018. 4

[39] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *ICCV*, pages 7463–7472, 2019. 2

[40] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114, 2019. 8

[41] 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, 2021. 2, 6

[42] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011. 8

[43] Guo-Hua Wang and Jianxin Wu. Practical network acceleration with tiny sets. In *CVPR*, pages 20331–20340, 2023. 2, 3, 6, 7

[44] Huanyu Wang, Junjie Liu, Xin Ma, Yang Yong, Zhenhua Chai, and Jianxin Wu. Compressing models with few samples: Mimicking then replacing. In *CVPR*, pages 691–700, 2022. 3, 4, 6, 7

[45] Jiahao Wang, Songyang Zhang, Yong Liu, Taiqiang Wu, Yujiu Yang, Xihui Liu, Kai Chen, Ping Luo, and Dahua Lin. Riformer: Keep your vision backbone effective but removing token mixer. In *CVPR*, pages 14443–14452, 2023. 3

[46] Wenxiao Wang, Shuai Zhao, Minghao Chen, Jinming Hu, Deng Cai, and Haifeng Liu. Dbp: Discrimination based block-level pruning for deep model acceleration. *arXiv preprint arXiv:1912.10178*, 2019. 3

[47] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *NeurIPS*, pages 2074–2082, 2016. 2

[48] Xia Xiao, Zigeng Wang, and Sanguthevar Rajasekaran. Autoprune: Automatic network pruning by regularizing auxiliary parameters. In *NeurIPS*, pages 13681–13691, 2019. 2

[49] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *CVPR*, pages 5790–5799, 2020. 2

[50] Linwei Ye, Mrigank Rochan, Zhi Liu, and Yang Wang. Cross-modal self-attention network for referring image segmentation. In *CVPR*, pages 10502–10511, 2019. 2

[51] Hongxu Yin, Pavlo Molchanov, José M. Álvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K. Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deep-inversion. In *CVPR*, pages 8712–8721, 2020. 6, 7

[52] Shixing Yu, Tianlong Chen, Jiayi Shen, Huan Yuan, Jianchao Tan, Sen Yang, Ji Liu, and Zhangyang Wang. Unified visual transformer compression. In *ICLR*, 2022. 1, 3

[53] Hongguang Zhang, Jing Zhang, and Piotr Koniusz. Few-shot learning via saliency-guided hallucination of samples. In *CVPR*, pages 2770–2779, 2019. 4

[54] Jinnian Zhang, Houwen Peng, Kan Wu, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. Minivit: Compressing vision transformers with weight multiplexing. In *CVPR*, pages 12135–12144, 2022. 2

[55] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H. S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, pages 6881–6890, 2021. 2

[56] Mingjian Zhu, Yehui Tang, and Kai Han. Vision transformer pruning. *arXiv preprint arXiv:2104.08500*, 2021. 2

[57] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2