VTC-LFC: Vision Transformer Compression with Low-Frequency Components

Anonymous Author(s)
Affiliation
Address
email

Abstract

Although Vision transformers (ViTs) have recently dominated many vision tasks, deploying ViT models on resource-limited devices remains a challenging problem. To address such a challenge, several methods have been proposed to compress ViTs. Most of them borrow experience in convolutional neural networks (CNNs) and mainly focus on the spatial domain. However, the compression only in the spatial domain suffers from a dramatic performance drop without fine-tuning and is not robust to noise, as the noise in the spatial domain can easily confuse the pruning criteria, leading to some parameters/channels being pruned incorrectly. Inspired by recent findings that self-attention is a low-pass filter and low-frequency signals/components are more informative to ViTs, this paper proposes compressing ViTs with low-frequency components. Two metrics named low-frequency sensitivity (LFS) and low-frequency energy (LFE) are proposed for better channel pruning and token pruning. Additionally, a bottom-up cascade pruning scheme is applied to compress different dimensions jointly. Extensive experiments demonstrate that the proposed method could save $40\% \sim 60\%$ of the FLOPs in ViTs, thus significantly increasing the throughput on practical devices with less than 1% performance drop on ImageNet-1K.

1 Introduction

Recently, Vision transformer (ViT) [13] and its variants [45, 28, 56] have outperformed convolutional neural networks (CNNs) in several vision tasks. However, ViT models still face the challenge of high computational cost when deployed to resource-limited devices. Following previous experiences in compressing CNN models, some pruning methods based on sparse learning [61, 55], taylor expansion [54], or automatic searching [8] have been proposed for ViT models to reduce model redundancy via channel pruning. In addition to the redundancy in parameters, recent literature [43, 24, 37] further points out that some noise tokens mainly encoded task-irrelevant information (*e.g.*, background), and some tokens become similar in deeper layers, showing that great redundancy also exists in tokens. The mainstream works [24, 42, 32] filter out the less informative tokens to reduce the FLOPs without changing the model structure.

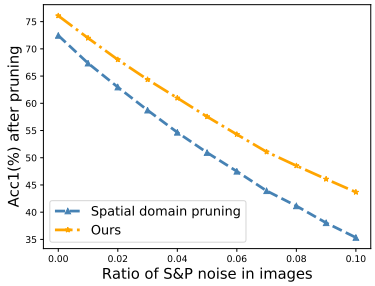


Figure 1: Noise resistance of spatial domain pruning and our pruning. 'S&P' means salt-and-pepper noise, the pruned model is DeiT-Small, and the performance is evaluated on ImageNet-1k.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Although aforementioned methods have made great progress in ViT compression in spatial domain, we find that they generally suffer from the following two problems: (i) different from CNN pruning which maintains the performance well without finetuning [30, 12], dramatic performance drop is observed when the same method is applied in ViT pruning; (ii) conducting ViT pruning only in spatial domain is not robust to noise, and as shown in Figure 1, after adding noise in the images, the accuracy of spatial compression dramatically drops. To make ViT compression more effective and robust, we propose to conduct ViT compress with the help of frequency domain. Recent studies [3, 50, 38, 34, 47] have indicated that self-attention (SA) behaves like a low-pass filter, and low-frequency signals/components are more informative to ViT models. Inspired by such low-frequency characteristics of ViT, we propose a compression framework named Vision Transformer Compression with Low-Frequency Components (VTC-LFC) which solves the problem from a new angle and emphasizes the contributions of low-frequency components during compression. To our best knowledge, this is the first work that compresses vision transformers in the frequency domain. The main contributions of this paper are listed as below:

Channel pruning based on low-frequency sensitivity: Channel pruning is a popular structured pruning strategy that aims to remove redundant parameters in fully connected layers of ViT. The mainstream works use some evaluation metrics (*e.g.* Taylor scores [54], weights norm [55], or sparse factor [61]) to estimate importance scores of parameters. Recent studies [3, 38, 47] find that ViTs are more reliable to the low-frequency components in images, *i.e.* the low-frequency information is more important for ViTs. Therefore, we infer that channels that are less effective in encoding low-frequency components will contribute less to the feature representation for ViT models. Motivated by such a property, we propose a better channel pruning criterion named low-frequency sensitivity (LFS) based on the Taylor scores [54]. Different from the standard Taylor scores which are computed with the original images, LFS filters out high-frequency components from images and uses only low-frequency components to estimate the importance of model parameters. In this way, channels that efficiently encode low-frequency information are more likely to be preserved, and the compressed model tends to be more robust to noise. Experimental results show that LFS can alleviate the performance drop after compression without bells and whistles.

Token pruning based on low-frequency energy: Token compression/sampling aims to select the informative tokens that store more useful information. The popular methods dynamically select those tokens with high correlation to other tokens (*e.g.* the CLS token) as the informative tokens. However, it may be sub-optimal because the selected tokens tend to be similar to each other, and the information included in the token itself has been neglected to some extent. As pointed out by [50, 34], the self-attention module in ViTs behaves like a low-pass filter, *i.e.* the tokens with more low-frequency components can pass more information to the next layers. Inspired by this, we propose improving the attention-based token selection with an extra item, token low-frequency energy (LFE), which quantifies the low-frequency information in tokens. By correcting the attention scores with LFE of tokens, the selector can better distinguish informative tokens from both the long-term dependency in spatial domain and low-frequency contributions in frequency domain.

Bottom-up Cascade Pruning Framework: To jointly compress channels and tokens of vision transformers, we propose a bottom-up cascade pruning framework. The model accuracy is further preserved through automatically balancing compression ratios of channel pruning and token pruning block-by-block.

2 Related work

Vision Transformer. Inspired by the success of transformers [46] in NLP, the Vision Transformer (ViT) [13] is proposed to encode an image into a sequence of tokens and feed them into the pure transformer architecture. Several studies have shown that ViT performs better than convolution neural networks (CNN) on image classification benchmarks [13, 17] when sufficient training data is provided. Many follow-up variants of ViT [4, 6, 1, 16, 49, 9, 56, 48, 59] have also been proposed. For example, DeiT [45] introduces a distillation token structure into ViT, and LV-ViT [21] proposes the token

labeling approach for better training of ViT. In addition to image classification, ViT has also achieved great performance in many other computer vision applications, such as semantic segmentation [10, 51, 11], image retrieval [19, 14], object detection [2, 62] and image reconstruction [7, 53]. However, despite of the outstanding performance in a series of tasks, its high computational cost restricts the deployment of ViT, which motivates the study of lightweight ViT models, including pruning [8, 61, 40, 52], block-weights sharing [23], and dynamic prediction architecture [60, 41], among which pruning is a universal approach for almost all model structures.

ViT Pruning. As an efficient compression approach, pruning [31, 20, 25, 26, 58, 29, 27] has been widely applied on various convolutional neural networks (CNNs) in computer vision. Pruning approaches [57, 18, 52, 43] have also been proposed for ViT to reduce its model size and inference time. These methods can be roughly grouped into two categories: **1) Channel pruning**, which reduces the number of weights, channels, heads or blocks in ViT. SViTE [8] jointly optimizes parameters and explores connectivity for both unstructured pruning (zeroing weights) and structured pruning (removing heads and channels). ViT-Slim [5] applies L_1 sparsity on channels and produces compressed ViTs with unstructured heads (the shape of heads is different). UVC [55] drops heads, channels, and blocks in a unified framework to achieve a high compression ratio. VTP [61] transfers the sparse-learning scheme in CNN pruning to compress ViT. NViT [54] generates smaller networks from the DeiT-base with the Taylor-based pruning scheme. **2) Token pruning**, which focuses on dynamically selecting significant tokens for different inputs. Token pruning would significantly reduce the computational cost while maintaining all parameters. TokenLearner [40] adaptively generates a small set of token vectors according to the spatial attention. EViT [24] downsamples tokens every three blocks and selects tokens with high correlation with the CLS token. DynamicViT [37] estimates the importance of tokens with an MLP [46] based predictor. IA-RED^2 [32] introduces a multi-head interpreter to drop uninformative tokens. SP-ViT [22] softly prunes tokens with token selector modules and packages the redundant tokens into one. Different from previous methods, this paper compresses ViTs from a novel prospect, frequency domain, to prune both parameters and tokens in a unified framework.

Frequency domain analysis for ViT and CNN. The recent explorations [3, 50, 38, 34, 47] of ViTs have indicated that ViTs behave in an opposite way to CNNs in frequency domain. [3] finds out that ViTs perform better than CNNs when only low-frequency components of images are fed into the models, and proposes the HAT method to enhance the capability of ViTs in capturing high-frequency information. [50] analyzes ViT features from the Fourier spectrum domain and shows that the self-attention module amounts to a low-pass filter. [34] also demonstrates that multi-head self-attentions exhibit opposite behaviors to convolutions, and take advantage of both mechanisms to design a novel AlterNet. To summarize, all these studies point out that the low-frequency components play an important role in information extraction of ViT.

3 Methodology

3.1 Preliminary

The necessary notations are defined as below. As shown in Figure 2, a transformer block contains a multi-head self-attention (MHSA) module with multiple heads and a feed-forward network (FFN) module with two fully-connection layers. The input images in a mini-batch are denoted as $X \in R^{B\times 3\times H\times W}$, where B, W and H are the batch size, width and height of images, respectively. The inputs of MHSA and FFN in the l-th block are denoted as $X^{l,1} \in R^{B\times N^l\times D}$ and $X^{l,2} \in R^{B\times N^l\times D}$, respectively. N^l is the number of tokens, and D is the dimension of a token. In the l-th block, the linear projection matrices $W_q^{l,h}$, $W_k^{l,h}$, and $W_v^{l,h}$ are used to calculate $Q_{l,h}$ (query), $K_{l,h}$ (key), and $V_{l,h}$ (value) for the h-th attention head. The parameters of the linear projection module in MHSA are denoted as W_{proj}^l, and two linear projection matrices in FFN are W_{fc1}^l and W_{fc2}^l. Our goal is to reduce the channel number of linear projection matrices and the token number N^l.

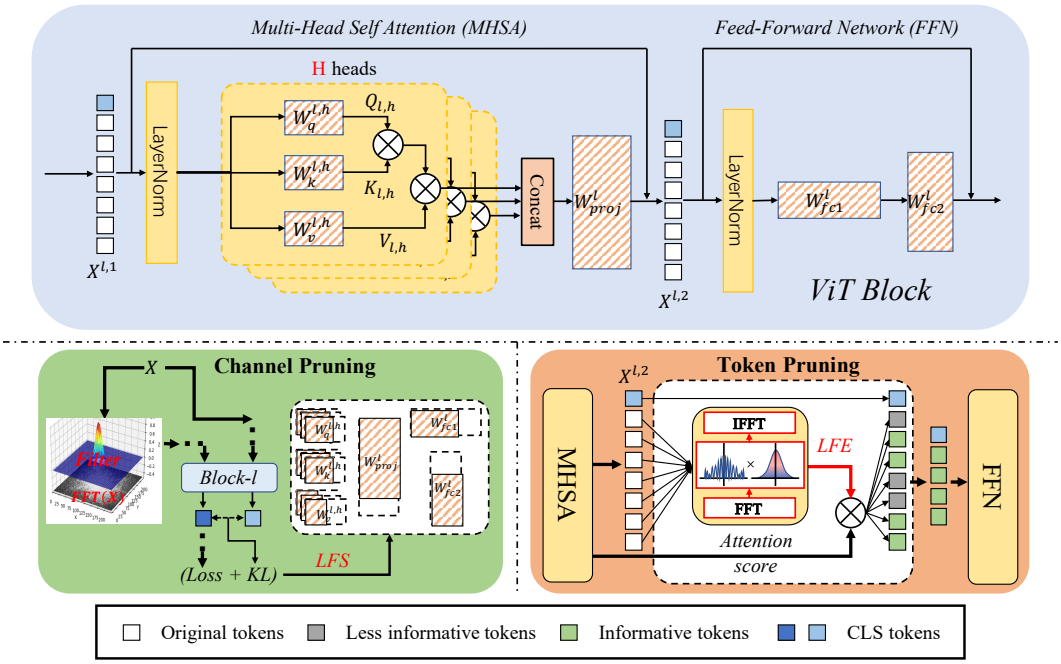


Figure 2: Pruning of channels and tokens in one block. '*LFE*' is the low-frequency energy extracted from tokens according to Equation 6. '*LFS*' denotes the low-frequency sensitivity used to evaluate the importance of channels.

3.2 Channel Pruning based on Low-Frequency Sensitivity

As previously introduced, low-frequency components in images are more valuable for the feature representation in ViT models, *i.e.* channels less effective in encoding low-frequency components will contribute less to the feature representation. Therefore, the key goal is to estimate the sensitivity of a channel to low-frequency components in images. To achieve this, we propose an evaluation policy named low-frequency sensitivity (LFS) that estimates the importance scores of model parameters by taking more low-frequency components in images into account.

Assume redundant channels have less influence on model outputs, removing redundant channels should hardly change the value of loss when feeding a set of training images into the model for loss computation. Thus, the importance of a channel can be quantified by the difference in loss induced by removing this channel. Given a number of images $X \in R^{B\times 3\times H\times W}$ randomly sampled from the training dataset $\mathcal{D}$, the importance score $\mathcal{I}_j$ of a weight w_j is formulated as:

$$\mathcal{I}_j = \left(\mathcal{L}\left(\mathcal{M}\left(X, \mathbf{W}\right), Y \mid w_j = 0\right) - \mathcal{L}\left(\mathcal{M}\left(X, \mathbf{W}\right), Y\right)\right)^2, \tag{1}$$

where $Y \in R^{B\times 1}$ is the label set of data X, $\mathcal{L}\left(\cdot\right)$ denotes the loss function (cross-entropy loss in this paper), $\mathcal{M}\left(X, \mathbf{W}\right)$ is the model output, and $\mathbf{W}$ indicates all model weights.

However, the score in Equation 1 can only reflect the importance on the original whole images. To separate low-frequency components from images, low-pass filtering is applied on images in the Fourier spectrum domain before feeding them into ViTs. The low-frequency components in images $\tilde{X}$ are formulated as:

$$\tilde{X} = \mathcal{F}^{-1}\left(\mathcal{G}\left(\sigma_c\right) \odot \mathcal{F}\left(X\right)\right), \tag{2}$$

where $\mathcal{F}\left(\cdot\right)$ and $\mathcal{F}^{-1}\left(\cdot\right)$ denote the fast Fourier transformation (FFT) [36] and the inverse fast Fourier transform (IFFT), respectively, $\odot$ is the Hadamard product, $\mathcal{G}\left(\cdot\right)$ is the low-pass filter, and $\sigma_c \in (0, 1)$ determines the cutoff frequency of the low-pass filter. Considering that a binary filter will cause the Ringing effect when the image is transformed back to the spatial domain, Gaussian filter is chosen for $\mathcal{G}\left(\cdot\right)$.

In addition to the task-specific loss, the pruned model shall also provide robust feature representation as the original model. In other words, the feature representation of the low-frequency images $\tilde{X}$ shall be as close to that of the original images X as possible. Hence, apart from the cross-entropy loss $\mathcal{L}$ for the classification task, a knowledge-distillation loss is also taken into account. Kullback–Leibler (KL) divergence loss $\mathcal{KL}(\cdot)$ is used to measure the error between the CLS tokens corresponding to the low-frequency image and nature image, respectively. Denote the two CLS tokens as $\tilde{T}$ (from low-frequency images) and T (from nature images), and simplify the cross-entropy loss $\mathcal{L}\left(\mathcal{M}\left(X, \mathbf{W}\right), Y\right)$ to $\mathcal{L}\left(X\right)$. Then, the final importance score s_j of weight w_j, named **Low-Frequency Sensitivity** (LFS), is formulated as:

$$s_j = \lambda \cdot \left(\mathcal{L}(\tilde{X} \mid w_j = 0) - \mathcal{L}(\tilde{X})\right)^2 + (1-\lambda) \cdot \left(\mathcal{KL}(\tilde{T}, T \mid w_j = 0) - \mathcal{KL}(\tilde{T}, T)\right)^2, \tag{3}$$

where λ is the hyper-parameter for the balance of two loss functions.

Calculating the LFS for each parameter with Equation 3 is infeasible for models with millions of parameters. Fortunately, the score can be approximated with the first-order Taylor expansion [30, 54]. Therefore, the approximated version of LFS is represented as below:

$$\hat{s}_j = \lambda \cdot \left(\frac{\partial \mathcal{L}(\tilde{X})}{\partial w_j} \cdot w_j\right)^2 + (1-\lambda) \cdot \left(\frac{\partial \mathcal{KL}(\tilde{T}, T)}{\partial w_j} \cdot w_j\right)^2, \tag{4}$$

where the gradient terms can be easily obtained in the backward procedure of the model. The channel importance score can then be approximated by summing over LFS scores of all parameters in the channel, *i.e.* the LFS of a channel is computed by the sum of $\hat{s}_j$:

$$\hat{\mathcal{S}}_{\mathcal{J}} = \sum_{j \in \mathcal{J}} \hat{s}_j, \tag{5}$$

where $\mathcal{J}$ means the index set of weights in a channel.

3.3 Token Pruning based on Low-Frequency Energy

Token redundancy is another major issue in the ViT compression, and several methods [42, 24, 22] sample informative tokens via analyzing the relationship or attention scores between tokens. Such a solution is sub-optimal because the selected tokens tend to be similar, and the information included in the token itself has been neglected to some extent. To address this problem, the **Low-Frequency Energy** (LFE) is proposed to make use of the low-frequency preference of ViT for token pruning. Following other works [42, 24, 22], as shown in Figure 2, the selector is located between the multi-head self-attention module and the feed-forward network module. Inspired by [50], we evaluate the low-frequency ratio of the token after transforming tokens $X^{l,2}$ into the frequency domain by applying FFT on each channel of tokens, denoted as $\mathcal{X}^{l,2}_{b,:,j} = \mathcal{F}\left(X^{l,2}_{b,:,j}\right)$. We then quantify the low-frequency information contained in a token by calculating the ratio of its remaining energy to the total energy after low-pass filtering. Given filter $\mathcal{G}$ with cutoff factor σ_t, the LFE is formulated as:

$$\eta_{l,i} = \frac{\left\|\mathcal{LC}\left[\mathcal{X}^{l,2}\right]\right\|_2}{\left\|\mathcal{DC}\left[\mathcal{X}^{l,2}\right]\right\|_2} = \frac{\left\|\mathcal{F}^{-1}(\mathcal{G}\left(\sigma_t\right) \odot \mathcal{X}^{l,2}_{b,i,:})\right\|_2}{\left\|\mathcal{F}^{-1}\left(\mathcal{X}^{l,2}_{b,:,:}\right)\right\|_2} = \frac{\left\|\tilde{X}^{l,2}_{b,i,:}\right\|_2}{\left\|X^{l,2}_{b,:,:}\right\|_2}, \tag{6}$$

where $\mathcal{DC}\left[\cdot\right]$ and $\mathcal{LC}\left[\cdot\right]$ denote the direct-current component and the low-frequency component, respectively. Intuitively, a token with more low-frequency components will achieve a larger $\eta_{l,i}$.

Similar to EViT [24], the attention scores in the spatial domain is also included to evaluate the final importance scores of tokens. For the h-th head in ViT, the attention value is calculated as:

$$\mathcal{A}^{l,h} = softmax\left(\frac{Q_{l,h}K^T_{l,h}}{\sqrt{d_{l,h}}}\right), \tag{7}$$

where $\mathcal{A}^{l,h}$ is the attention score matrix and $d_{l,h}$ is the output dimension. The CLS token plays a more significant role than other tokens because it is the final output feature which collects information

from all tokens. Moreover, the head with denser and larger attention values is more important, *i.e.*, with a larger proportion. Thus, our proposed modified attention score is formulated as:

$$\hat{\mathcal{T}}_{l,i} = \frac{1}{H}\sum_{h=0}^{H-1}\left(\theta_{h,0}\cdot\mathcal{A}_{i,0}^{l,h} + \theta_{h,1}\cdot\frac{1}{N^l}\sum_{j=1}^{N^l-1}\mathcal{A}_{i,j}^{l,h}\right), \tag{8}$$

where $\theta_{h,0} = \sum_{j=1}^{N-1}\mathcal{A}_{0,j}^{l,h}$ and $\theta_{h,1} = \mathcal{A}_{0,0}^{l,h}$ are the head-weights of the CLS attention value $\mathcal{A}_{i,0}^{l,h}$ and the other attention value $\mathcal{A}_{i,j}^{l,h}$, respectively.

To estimate the importance score of tokens from multiple and diverse aspects, we consider to combine the LFE $\eta_{l,i}$ and attention score $\hat{\mathcal{T}}_{l,i}$ to get the final importance score of a token as:

$$\tilde{\mathcal{T}}_{l,i} = \hat{\mathcal{T}}_{l,i}\cdot\eta_{l,i}, \tag{9}$$

Note: the CLS token is the final output of the ViT model, and is not involved in the token pruning.

3.4 Bottom-up Cascade Pruning

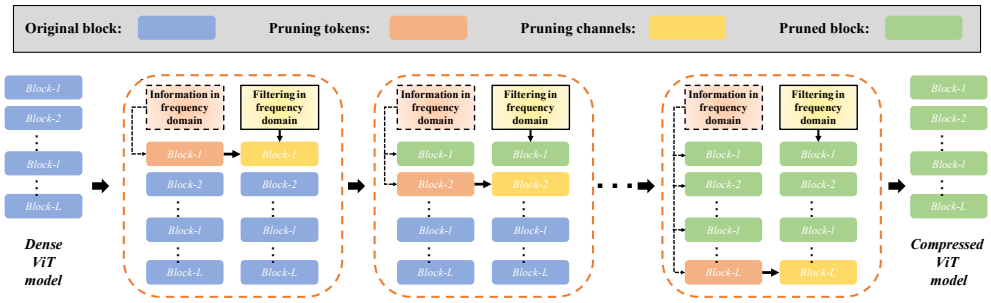


Figure 3: Pipeline of Bottom-up Cascade Pruning.

In our method, the value of LFS and LFE in each block is related to the outputs from previous blocks. Hence, the compression in one block will influence the LFS and LFE in its subsequent blocks. It is sub-optimal to independently determine the pruning ratios and indices of channels and tokens for all blocks at once and ignore their inter-relationship. Therefore, we design a **Bottom-up Cascade Pruning** (BCP) process (Figure 3), which promotes pruning from the first block to the last block. A hyper-parameter, named global allowable drop ε, is set to control the final performance drop after pruning. During compression of each block, the number of tokens is gradually reduced until the performance drop reaches $\varepsilon_t = \rho\cdot\varepsilon/L$, where ρ is a hyper-parameter to control the accuracy drop caused by token pruning. Once token pruning is done, channels will be compressed with a similar procedure. When the performance drop caused by pruning reaches ε/L, the compression for a block is considered as completed. More details are described in Algorithm 1 in Appendix A.1.

4 Experiments

In this section, the proposed method is evaluated on the benchmark ImageNet (ILSVRC2012) [39], which is a large dataset containing 1.2M training images and 50k validation images of 1000 classes. All the experiments are deployed with Pytorch [35] on NVIDIA V100 GPUs. The code is modified based on the previous study DeiT[1]. The float operations (FLOPs) of models are evaluated by fvcore[2].

[1] https://github.com/facebookresearch/deit
[2] https://github.com/facebookresearch/fvcore

4.1 Experiments on ImageNet

Implementation details. The proposed method is applied to popular ViT models of three different sizes, DeiT-Tiny, DeiT-Small, and DeiT-Base. The latest state-of-the-art (SOTA) methods are compared, including SCOP [44], CP-ViT [42], PoWER [15], HVT [33], IA-RED^2 [32], S^{2}ViTE [8], EViT [24], and SPViT [18]. In the pruning procedure, the number of training samples used for evaluating the performance drop in BCP is 5000 (randomly sampling 5 training samples from each category), the number of training samples for calculating LFS is 2000, and the cutoff factors σ_c and σ_t are 0.1 and 0.85. For three models, DeiT-Tiny, DeiT-Small, and DeiT-Base, the global allowable drop ε are 9.5, 14, and 14, and the ratio ρ for the allowable drop is 0.55, 0.35, and 0.3 respectively. The removed channels involve the columns (output channels) of $W_q^{l,h}$, $W_q^{l,h}$, $W_q^{l,h}$, and W_{fc1}^l and rows (input channels) of W_{proj}^l, and W_{fc2}^l. After pruning, the compressed models are fine-tuned with hard distillation [45] of their corresponding original models. The base learning rate is set to 0.0001, and most of the other hyper-parameters follow the settings in [8]. We fine-tune the pruned DeiT-Tiny/DeiT-Small/DeiT-Base models for 300/150/75 epochs.

Results and analysis. The comparison with state-of-the-art methods is shown in Table 1, in which the top-1 accuracy and the reduction ratios of FLOPs are reported. In all three models, the proposed method achieves the highest reduction ratio in FLOPs than previous methods with less than 1% performance drop. Compared to the pure token pruning methods like SCOP [44], CP-ViT [42], PoWER [15], HVT [33], and IA-RED^2 [32], our method achieves not only better performance and more reduction in FLOPs, but also less parameters, which demonstrates the superiority of pruning with frequency domain. Although EViT [24], SPViT [18], and S^{2}ViTE [8] achieves better accuracy on DeiT-Base, their reduction in FLOPs (EViT 34.1%, SPViT 33.1%, S^{2}ViTE 33.1% vs Our 57.6%) is much lower than ours. To evaluate the acceleration on inference speed of our pruning technique, the throughput is assessed on a single V100 GPU with batch size 256 in Table 2. The DeiT-Tiny/Small/Base model achieves 69.7%/97.0%/107.1% speed up after pruning, which demonstrates the practicability of the proposed compression approach.

Table 1: Comparison with state-of-the-art methods on ImageNet-1k. 'FLOPs ↓' denotes the reduction ratio of FLOPs. We report two versions with different parameter sizes for our method.

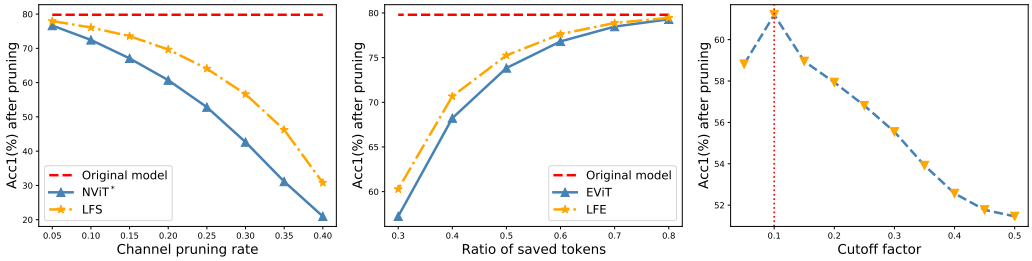
Method	DeiT-Tiny			DeiT-Small			DeiT-Base		
	Acc1(%)	**FLOPs ↓**	**Params**	**Acc1(%)**	**FLOPs ↓**	**Params**	**Acc1(%)**	**FLOPs ↓**	**Params**
Baseline	72.2	–	5.7M	79.8	–	22.1M	81.8	–	86.4M
SCOP [44]	68.9	38.4%	5.7M	77.5	43.6%	22.1M	79.7	42.0%	86.4M
PoWER [15]	69.4	38.4%	5.7M	78.3	41.3%	22.1M	80.1	39.2%	86.4M
CP-ViT [42]	71.2	43.3%	5.7M	79.1	42.2%	22.1M	81.1	41.6%	86.4M
EViT [24]	–	–	–	78.5	50.0%	22.1M	81.3	34.1%	86.4M
HVT [33]	69.7	46.2%	5.7M	78.0	47.8%	22.1M	–	–	–
IA-RED^2 [32]	–	–	–	79.1	31.5%	22.1M	80.3	33.0%	86.4M
S^{2}ViTE [8]	70.1	23.7%	4.2M	79.2	31.6%	14.6M	82.2	33.1%	56.8M
SPViT [18]	70.7	23.1%	4.9M	78.3	28.3%	16.4M	81.6	33.1%	62.3M
VTC-LFC (Ours)	**71.6**	**46.7%**	**5.1**M	**79.4**	**54.4%**	**17.7**M	**81.2**	**57.6%**	**63.5**M
VTC-LFC (Ours)	**71.0**	**41.7%**	**4.2**M	**79.6**	**47.1%**	**15.3**M	**81.6**	**54.4%**	**56.8**M

Table 2: Throughput of baselines and compressed models. 'Speed up' means the improvement in throughput. 'base' denotes the baseline model, and 'pruned' is the compressed model.

Model	**Acc1 (base/pruned)**	**Throughput (base/pruned)**	**Speed up**
DeiT-Tiny	72.2%/71.3%	2648.7/4496.2	69.7%
DeiT-Small	79.8%/79.4%	987.9/1946.3	97.0%
DeiT-Base	81.8%/81.2%	314.7/651.9	107.1%

Table 3: Results of channel pruning and token pruning with different criteria on DeiT-Small. In the column of 'Acc1 (%)', 'FT' means fine-tuning. 'FLOPs ↓' denotes the reduction ratio of FLOPs. It is noted that the original NViT compresses a large-scale model to the target size (*e.g.* ViT-Small) and uses extra CNN teacher models, so we implement NViT* to compress ViT-Small to the pruned ViT-Small under the standard pruning setting for fair comparison here. For a clear comparison, BCP is not applied in any experiments here.

Channel Pruning		**Token Pruning**		**Acc1 (%)**		**FLOPs ↓**
NViT*[54] (baseline)	LFS (ours)	EViT[24] (baseline)	LFE (ours)	before FT	after FT	
×	×	×	×	79.8	–	0.0%
✓	×	×	×	47.5	78.9	32.8%
×	✓	×	×	61.2	79.4	32.8%
×	×	✓	×	76.8	79.6	43.3%
×	×	×	✓	77.6	80.1	43.3%
✓	×	✓	×	40.5	78.0	55.0%
×	✓	×	✓	57.9	78.7	55.0%



(a) Channel pruning (b) Token pruning (c) Cutoff factor σ_c

Figure 4: (a) results of channel pruning with two different criteria 'NViT*' and 'LFS'. (b) results after token pruning with criteria 'EViT' and 'LFE'. (c) results with different cutoff factors to determine the cutoff frequency of low-pass filters.

4.2 Ablation Study

Effectiveness of LFS and LFE. For fair comparison to analyze the effectiveness of the proposed LFS (for channel pruning) and LFE (for token pruning), we conduct experiments without using BCP framework in Table 3. Two state-of-the-art methods NViT [54] and EViT [24] are selected as baselines for channel pruning and token pruning, respectively. NViT identifies channels based on the Taylor score and EViT selects tokens according to the attention score. The channels are pruned globally using the manual pruning rate as NViT, and the number of tokens is determined following the same ratio as EViT. The main difference between other methods and ours is whether to leverage the characteristics of ViT on low-frequency components. Table 3 shows the results of only compressing channels, tokens, and both. For channel pruning, it can be found that LFS outperforms NViT* with the same FLOPs reduction. For token pruning, LFE achieves an even higher accuracy than the original DeiT-Small. Both comparison results demonstrate the superiority of pruning based on information in the frequency domain over using only information in the space domain. The results of different pruning ratios for channels and tokens are displayed in Figure 4(a) and 4(b), in which the proposed method consistently outperform the other methods under all ratios.

The influence of the cutoff factor, which determines the ratio of saved low-frequency components to all frequency, is also analyzed. As shown in Figure 4(c), $\sigma_c = 0.1$ is the sweet spot, which further proves the preference of ViT on low-frequency information. For σ_t, 0.85 is the best choice.

Factor analysis for BCP. The pruning ratios for channels and tokens are determined by two hyper-parameters ε and ρ. As shown in Table 4, when maintaining $\rho = 0.35$ and increasing ε from 10 to

Table 4: Automatically searching pruning ratios with BCP.'ε' is the hyper-parameter that controls the performance drop caused by pruning, and 'ρ' is a hyper-parameter that balances channel pruning and token pruning. All results are achieved by DeiT-Small.

ε/ρ	$10/0.35$	$13/0.35$	$14/0.15$	$14/0.35$	$14/0.55$	$15/0.35$	$18/0.35$
FLOPs $\downarrow$	48.6%	54.0%	46.0%	54.4%	59.2%	55.2%	58.5%
Params	18.7M	17.8M	17.3M	17.7M	17.9M	17.4M	16.8M
Acc1 w/o FT	71.6%	69.2%	68.8%	68.3%	69.0%	66.9%	64.6%
Acc1 w/ FT	79.8%	79.3%	79.9%	79.4%	78.7%	79.1%	78.9%

18, the model parameter size will be reduced from 18.7M to 16.8M, and the performance ranges from 79.8% to 78.9%. Similarly, when maintaining $\varepsilon = 14$ and increasing ρ from 0.15 to 0.55, both the model size and the FLOPS reduction ratio are increased, while the performance is reduced from 79.9% to 78.7%. Compared with the result (78.7% accuracy and 55.0% Flops reduction ratio) without BCP in the last row of Table 3, we can observe that BCP improves the accuracy or compression ratio of the model, which verifies the effectiveness of the proposed BCP strategy. After comprehensively considering the model parameters size, the FLOPS reduction ratio and model accuracy, we set the $\varepsilon = 14$ and $\rho = 0.35$ in this paper. These two parameters can be adjusted according to specific requirements in real-world applications.

Influence of each module. To further study how each component affects the proposed method, LFS, LFE, and BCP are respectively removed from our scheme. The modified versions are then applied on the same model. Since BCP will automatically adjust pruning ratios of each block, it is necessary to introduce additional variables for controlled experiments. For a fair comparison, we keep the same pruning ratios (determined by our VTC-LFC) of each block in all experiments. As shown in Table 5, the proposed LFS/LFE/BCP improves the performance by 1.3%/0.6%/0.9% before fine-tuning and 0.1%/0.5%/0.2% after fine-tuning, respectively. The experimental results show the effectiveness of the proposed modules.

Table 5: Effects of LFS, LFE, and BCP on the proposed compression method. 'Original' is the original model without pruning. 'Ours' is the result including LFS, LFE, and BCP. 'NvEv-P' means setting pruning rates for channels and tokens following the strategies in NViT and EViT, respectively.

Model	**Method**	**Acc1 (before FT)**	**Acc1 (after FT)**	**FLOPs** $\downarrow$
DeiT-Small	Original	79.8%	–	0.0%
	VTC-LFC (Ours)	68.3%	79.4%	54.4%
	w/o LFS (LFS$\rightarrow$NViT*)	67.0%	79.3%	54.4%
	w/o LFE (LFE$\rightarrow$EViT)	67.7%	78.9%	54.4%
	w/o BCP (BCP$\rightarrow$NvEv-P)	67.4%	79.2%	54.4%

5 Conclusion and Discussion

This paper reveals the disadvantages of pruning ViTs only in the spatial domain and takes advantage of the preference of ViTs for low-frequency information to conduct compression. Two metrics, low-frequency sensitivity and low-frequency energy are proposed to leverage knowledge in the frequency domain for better channel pruning and token pruning. The comparison with the spatial domain pruning approaches proves that the proposed method can identify the informative channels and tokens more precisely, thus better maintaining the model accuracy. Additionally, with the proposed bottom-up cascade pruning strategy, both channels and tokens are automatically compressed in a unified framework. Extensive experiments of different models on ImageNet demonstrate that more than half of computational costs are saved from ViTs, with significant improvements in inference efficiency. The comparison with the latest compression methods shows the superiority (better performance and more FLOPs reduction) of the proposed approach. As a preliminary study about pruning ViTs with the frequency domain, more efficient ways and deeper studies are expected to be explored based on it.

References

[1] A. Ali, H. Touvron, M. Caron, P. Bojanowski, M. Douze, A. Joulin, I. Laptev, N. Neverova, G. Synnaeve, J. Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34, 2021.

[2] A. Amini, A. S. Periyasamy, and S. Behnke. T6d-direct: Transformers for multi-object 6d pose direct regression. In *DAGM German Conference on Pattern Recognition*, pages 530–544. Springer, 2021.

[3] J. Bai, L. Yuan, S.-T. Xia, S. Yan, Z. Li, and W. Liu. Improving vision transformers by revisiting high-frequency components. *arXiv preprint arXiv:2204.00993*, 2022.

[4] H. Bao, L. Dong, and F. Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[5] A. Chavan, Z. Shen, Z. Liu, Z. Liu, K.-T. Cheng, and E. Xing. Vision transformer slimming: Multi-dimension searching in continuous optimization space. *arXiv preprint arXiv:2201.00814*, 2022.

[6] C.-F. R. Chen, Q. Fan, and R. Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 357–366, 2021.

[7] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021.

[8] T. Chen, Y. Cheng, Z. Gan, L. Yuan, L. Zhang, and Z. Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34, 2021.

[9] X. Chen, C.-J. Hsieh, and B. Gong. When vision transformers outperform resnets without pre-training or strong data augmentations. *arXiv preprint arXiv:2106.01548*, 2021.

[10] B. Cheng, A. Schwing, and A. Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems*, 34, 2021.

[11] L. Ding, D. Lin, S. Lin, J. Zhang, X. Cui, Y. Wang, H. Tang, and L. Bruzzone. Looking outside the window: Wide-context transformer for the semantic segmentation of high-resolution remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 2022.

[12] X. Ding, X. Zhou, Y. Guo, J. Han, J. Liu, et al. Global sparse momentum sgd for pruning very deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[14] A. El-Nouby, N. Neverova, I. Laptev, and H. Jégou. Training vision transformers for image retrieval. *arXiv preprint arXiv:2102.05644*, 2021.

[15] S. Goyal, A. R. Choudhury, S. Raje, V. Chakaravarthy, Y. Sabharwal, and A. Verma. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pages 3690–3699. PMLR, 2020.

[16] B. Graham, A. El-Nouby, H. Touvron, P. Stock, A. Joulin, H. Jégou, and M. Douze. Levit: a vision transformer in convnet's clothing for faster inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12259–12269, 2021.

[17] J. Guo, K. Han, H. Wu, C. Xu, Y. Tang, C. Xu, and Y. Wang. Cmt: Convolutional neural networks meet vision transformers. *arXiv preprint arXiv:2107.06263*, 2021.

[18] H. He, J. Liu, Z. Pan, J. Cai, J. Zhang, D. Tao, and B. Zhuang. Pruning self-attentions into convolutional layers in single path. *arXiv preprint arXiv:2111.11802*, 2021.

[19] S. He, H. Luo, P. Wang, F. Wang, H. Li, and W. Jiang. Transreid: Transformer-based object re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15013–15022, 2021.

[20] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[21] Z.-H. Jiang, Q. Hou, L. Yuan, D. Zhou, Y. Shi, X. Jin, A. Wang, and J. Feng. All tokens matter: Token labeling for training better vision transformers. *Advances in Neural Information Processing Systems*, 34, 2021.

[22] Z. Kong, P. Dong, X. Ma, X. Meng, W. Niu, M. Sun, B. Ren, M. Qin, H. Tang, and Y. Wang. Spvit: Enabling faster vision transformers via soft token pruning. *arXiv preprint arXiv:2112.13890*, 2021.

[23] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2019.

[24] Y. Liang, G. Chongjian, Z. Tong, Y. Song, J. Wang, and P. Xie. Evit: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2021.

[25] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1529–1538, 2020.

[26] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. Doermann. Towards optimal structured cnn pruning via generative adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[27] N. Liu, X. Ma, Z. Xu, Y. Wang, J. Tang, and J. Ye. Autocompress: An automatic dnn structured pruning framework for ultra-high compression rates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4876–4883, 2020.

[28] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[29] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.-T. Cheng, and J. Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3296–3305, 2019.

[30] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11264–11272, 2019.

[31] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz. Importance estimation for neural network pruning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[32] B. Pan, R. Panda, Y. Jiang, Z. Wang, R. Feris, and A. Oliva. Ia-red^2: Interpretability-aware redundancy reduction for vision transformers. *Advances in Neural Information Processing Systems*, 34, 2021.

[33] Z. Pan, B. Zhuang, J. Liu, H. He, and J. Cai. Scalable vision transformers with hierarchical pooling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 377–386, 2021.

[34] N. Park and S. Kim. How do vision transformers work? In *International Conference on Learning Representations*, 2021.

[35] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[36] C. Rader and N. Brenner. A new principle for fast fourier transformation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(3):264–266, 1976.

[37] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34, 2021.

[38] Y. Rao, W. Zhao, Z. Zhu, J. Lu, and J. Zhou. Global filter networks for image classification. *Advances in Neural Information Processing Systems*, 34, 2021.

[39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[40] M. S. Ryoo, A. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova. Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv preprint arXiv:2106.11297*, 2021.

[41] R. Schwartz, G. Stanovsky, S. Swayamdipta, J. Dodge, and N. A. Smith. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6640–6651, 2020.

[42] Z. Song, Y. Xu, Z. He, L. Jiang, N. Jing, and X. Liang. Cp-vit: Cascade vision transformer pruning via progressive sparsity prediction. *arXiv preprint arXiv:2203.04570*, 2022.

[43] Y. Tang, K. Han, Y. Wang, C. Xu, J. Guo, C. Xu, and D. Tao. Patch slimming for efficient vision transformers. *arXiv preprint arXiv:2106.02852*, 2021.

[44] Y. Tang, Y. Wang, Y. Xu, D. Tao, C. Xu, C. Xu, and C. Xu. Scop: Scientific control for reliable neural network pruning. *Advances in Neural Information Processing Systems*, 33:10936–10947, 2020.

[45] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[47] H. Wang, X. Wu, Z. Huang, and E. P. Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8684–8694, 2020.

[48] P. Wang, X. Wang, H. Luo, J. Zhou, Z. Zhou, F. Wang, H. Li, and R. Jin. Scaled relu matters for training vision transformers. *arXiv preprint arXiv:2109.03810*, 2021.

[49] P. Wang, X. Wang, F. Wang, M. Lin, S. Chang, W. Xie, H. Li, and R. Jin. Kvt: k-nn attention for boosting vision transformers. *arXiv preprint arXiv:2106.00515*, 2021.

[50] P. Wang, W. Zheng, T. Chen, and Z. Wang. Anti-oversmoothing in deep vision transformers via the fourier domain analysis: From theory to practice. In *International Conference on Learning Representations*, 2021.

[51] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia. End-to-end video instance segmentation with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8741–8750, 2021.

[52] Y. Xu, Z. Zhang, M. Zhang, K. Sheng, K. Li, W. Dong, L. Zhang, C. Xu, and X. Sun. Evo-vit: Slow-fast token evolution for dynamic vision transformer. *arXiv preprint arXiv:2108.01390*, 2021.

[53] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5791–5800, 2020.

[54] H. Yang, H. Yin, P. Molchanov, H. Li, and J. Kautz. Nvit: Vision transformer compression and parameter redistribution. *arXiv preprint arXiv:2110.04869*, 2021.

[55] S. Yu, T. Chen, J. Shen, H. Yuan, J. Tan, S. Yang, J. Liu, and Z. Wang. Unified visual transformer compression. In *International Conference on Learning Representations*, 2021.

[56] L. Yuan, Q. Hou, Z. Jiang, J. Feng, and S. Yan. Volo: Vision outlooker for visual recognition. *arXiv preprint arXiv:2106.13112*, 2021.

[57] C. Yun, Y.-W. Chang, S. Bhojanapalli, A. S. Rawat, S. Reddi, and S. Kumar. $o(n)$ connections are expressive enough: Universal approximability of sparse transformers. *Advances in Neural Information Processing Systems*, 33:13783–13794, 2020.

[58] C. Zhao, B. Ni, J. Zhang, Q. Zhao, W. Zhang, and Q. Tian. Variational convolutional neural network pruning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[59] J. Zhou, P. Wang, F. Wang, Q. Liu, H. Li, and R. Jin. Elsa: Enhanced local self-attention for vision transformer. *arXiv preprint arXiv:2112.12786*, 2021.

[60] W. Zhou, C. Xu, T. Ge, J. McAuley, K. Xu, and F. Wei. Bert loses patience: Fast and robust inference with early exit. *Advances in Neural Information Processing Systems*, 33:18330–18341, 2020.

[61] M. Zhu, K. Han, Y. Tang, and Y. Wang. Visual transformer pruning. *arXiv e-prints*, pages arXiv–2104, 2021.

[62] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [No] This work is scientific in nature, and we do not believe it has immediate negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] Our work does not contain theoretical results.
   (b) Did you include complete proofs of all theoretical results? [N/A] Our work does not contain theoretical results.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We will release the code in the future. We believe the work can be reproduced from the information in the paper and appendix.
   (b) Did you specify all the training details (e.g., data splits, hyper-parameters, how they were chosen)? [Yes] Please see Section 4 and appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We did not report the error bars since training vision transformer on ImageNet is stable. the error bars can be neglected, and the results can be reproduced with the hyper-parameters introduced in the paper.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We describe the details of computation resources in Section 4
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We used publicly available data, i.e., ImageNet, in our experiments. We cited the corresponding papers published by the creators in Section 4.
   (b) Did you mention the license of the assets? [N/A] The license of ImageNet is included in the paper that we have cited.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] The ImageNet we used are publicly available.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We did not collect/curate new data.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] All ImageNet datasets are already publicly available and broadly adopted. We do not think there are any issues of personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]