GOHSP: A Unified Framework of Graph and Optimization-Based Heterogeneous Structured Pruning for Vision Transformer

Miao Yin[1]* **Burak Uzkent**[2], **Yilin Shen**[2], **Hongxia Jin**[2], **Bo Yuan**[1]

[1] Rutgers University
[2] Samsung Research America

Abstract

The recently proposed Vision transformers (ViTs) have shown very impressive empirical performance in various computer vision tasks, and they are viewed as an important type of foundation model. However, ViTs are typically constructed with large-scale sizes, which then severely hinder their potential deployment in many practical resources-constrained applications. To mitigate this challenging problem, structured pruning is a promising solution to compress model size and enable practical efficiency. However, unlike its current popularity for CNNs and RNNs, structured pruning for ViT models is little explored.

In this paper, we propose GOHSP, a unified framework of Graph and Optimization-based Structured Pruning for ViT models. We first develop a graph-based ranking for measuring the importance of attention heads, and the extracted importance information is further integrated to an optimization-based procedure to impose the heterogeneous structured sparsity patterns on the ViT models. Experimental results show that our proposed GOHSP demonstrates excellent compression performance. On CIFAR-10 dataset, our approach can bring 40% parameters reduction with no accuracy loss for ViT-Small model. On ImageNet dataset, with 30% and 35% sparsity ratio for DeiT-Tiny and DeiT-Small models, our approach achieves 1.65% and 0.76% accuracy increase over the existing structured pruning methods, respectively.

Introduction

Recently applying transformer architecture to computer vision has emerged as an important forefront of foundation model design (Dosovitskiy et al. 2020). Thanks to the delicate vision-specific self-attention, inherent minimal inductive biases and high scalability and parallelism, *vision transformers (ViTs)* (Dosovitskiy et al. 2020; Touvron et al. 2021; Zhou et al. 2021) have shown very outstanding and even state-of-the-art performance in many fundamental and downstream image and video processing tasks, such as image classification, object detection, super-resolution, video classification etc.

Motivated by the scaling success of the giant natural language processing (NLP) transformers (e.g., BERT (Devlin et al. 2018) and GPT-3 (Brown et al. 2020)), the existing ViTs are also constructed with large model sizes to adapt for massive data training (Zhai et al. 2021). Consequently, they are suffering from huge memory footprints and extensive computational costs. These limitations, if not being properly addressed, could severely hinder the widespread adoption of ViTs in many practical scenarios, especially on the resource-constrained mobile platforms and Internet-of-things (IoT) devices.

To mitigate this challenging problem, one attractive solution is to perform model compression (Yu et al. 2017; Kim et al. 2016; Pan et al. 2019) to reduce the network costs without affecting task performance. However, unlike the current popularity of compressing convolutional and recurrent neural networks (CNNs and RNNs), ViT-oriented model compression has not been systematically studied yet. In particular, *structured pruning*, as an important hardware-friendly compression strategy that can bring practical efficiency on the off-the-shelf hardware, is little explored for ViT models.

To date, a rich set of structured pruning approaches have been proposed and investigated in the existing literatures, and most of them focus on sparsifying the CNNs at the channel level (He, Zhang, and Sun 2017; Ye et al. 2018). On the other hand, as will be analyzed and elaborated in Section , because of the difference of the underlying architecture, the structured sparse ViT models can exhibit multi-granularity sparsity (i.e., head-level and column-level) in the different component modules (i.e., attention head and multi-layer perception (MLP)). The co-existence of such heterogeneous sparse patterns raises a series of new research challenges and questions when we consider the efficient structured pruning strategy for ViT models. For instance, for each component module what is the corresponding suitable pruning criterion to obtain its specific sparse pattern? Also, how should we perform the entire pruning process across different modules with different levels of granularity sparsity to optimize the overall compression and task performance?

Technical Preview & Contributions. To answer these questions, in this paper we propose GOHSP, a unified framework of Graph and Optimization-based Structure Pruning for vision transformer. To be specific, we first develop a graph-based ranking approach to measure the importance of attention heads. As a soft-pruning guideline, such importance information is then integrated to the overall

*This work was done during Miao Yin's internship at Samsung Research America.

optimization-based procedure to impose the different types of structured sparsity in a joint and global way. Overall, the contributions of this paper are summarized as follows:

- We propose a graph-based ranking algorithm to measure and determine the importance of attention heads. By modeling the inter-head correlation as a converged Markov chain, the head importance can be interpreted and calculated as the stationary distribution, which is further used as a soft guideline for the overall pruning procedure.
- We propose a unified framework to jointly optimize different types of structured sparsity in the different modules. The complicated coordination for different sparse patterns are automatically learned and optimized in a systematic way.
- We evaluate the performance of our structured pruning approach of different ViT models. Particularly, On ImageNet dataset, with 30% and 40% sparsity ratio for DeiT-Tiny and DeiT-Small models, our approach achieves 1.65% and 0.76% accuracy increase than the existing structured pruning methods, respectively.

Related Work

Vision Transformer. Inspired by the grand success of transformer architecture in NLP domains, deep learning researchers have actively explored the efficient transformer-based neural networks for computer vision. Most recently, several vision transformers (ViTs) and their variants have already shown very impressive performance in several image and video processing tasks (Dosovitskiy et al. 2020; Touvron et al. 2021; Zhou et al. 2021). However, in order to achieve competitive performance with the state-of-the-art CNNs, ViTs typically have to scale up their model sizes and therefore they suffer from costly computation and storage.

Dynamic Inference with ViTs. To reduce the deployment costs of ViTs, several works (Wang et al. 2021; Bakhtiarnia, Zhang, and Iosifidis 2021; Rao et al. 2021; Meng et al. 2022; Xu et al. 2022; Uzkent, Yeh, and Ermon 2020; Uzkent and Ermon 2020) have been proposed to improve the processing speed via dynamically pruning the tokens/patches or skipping transformer components adaptively. Essentially as dynamic inference approaches, this set of works do not pursue to reduce the model sizes but focus on input-aware inference to obtain practical speedup. Our structured pruning-based solution is orthogonal to them, and these two different strategies can be potentially combined together to achieve higher speed and smaller memory footprint.

Structured Pruning. Model compression is a promising strategy to reduce the deployment costs of neural networks. Among various model compression techniques, structured pruning is a very popular choice because its hardware-friendly nature can bring practical efficiency on the real-world devices. Based on different pruning criterias, various structured pruning approaches have been extensively studied in the existing literature (Yu et al. 2018; Zhuang et al. 2018; Liu et al. 2019; He et al. 2019; Lin et al. 2020; Tiwari et al. 2021; Lou et al. 2022), and most of them focus on pruning

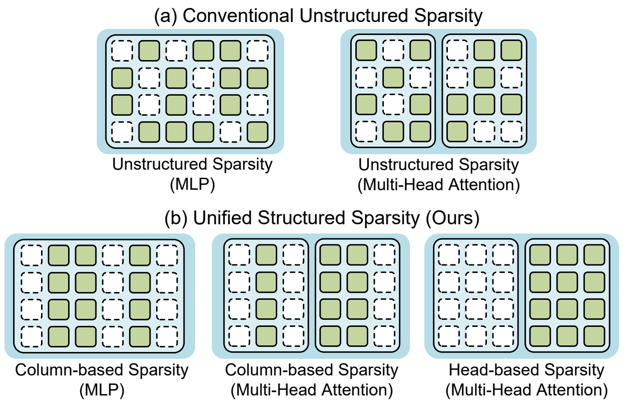
(a) Conventional Unstructured Sparsity
Unstructured Sparsity (MLP)
Unstructured Sparsity (Multi-Head Attention)
(b) Unified Structured Sparsity (Ours)
Column-based Sparsity (MLP)
Column-based Sparsity (Multi-Head Attention)
Head-based Sparsity (Multi-Head Attention)

Figure 1: (a) Sparsity pattern of ViT models after unstructured pruning. Only part of the Multi-Head Attention and MLP columns are pruned which are not hardware-friendly.(b) Heterogeneous sparsity patterns of ViT models after structured pruning. Certain MLP and Multi-Head Attention columns are removed which is hardware-friendly.

CNN models; while the efficient structured pruning of ViTs is little explored. One of these studies, (Chen et al. 2021), prunes the vision transformers using structured pruning. (Yu et al. 2022), on the other hand, focuses on FLOPs reduction with the vision transformers using pruning, layer skipping, and knowledge distillation whereas in our study we focus on structured pruning to mainly reduce the number of parameters for building hardware-friendly compressed models. For this reason, we compare our method to (Chen et al. 2021).

Structured Pruning of ViTs: Analysis

Notation. Considering an L-block vision transformer, $\boldsymbol{W}_{\text{attn}}^{(l)} = \{\boldsymbol{W}_{\text{qkv}}^{(l)}, \boldsymbol{W}_{\text{proj}}^{(l)}\}$ and $\boldsymbol{W}_{\text{mlp}}^{(l)} = \{\boldsymbol{W}_{\text{fc1}}^{(l)}, \boldsymbol{W}_{\text{fc2}}^{(l)}\}$ represent the weights of the attention layer and the MLP layer at l-th block, respectively. For each attention layer, there are H self-attention heads, namely $\boldsymbol{W}_{\text{qkv}}^{(l)} = \{\boldsymbol{W}_{\text{qkv}}^{(l,h)}\}_{h=1}^{H}$ and $\boldsymbol{W}_{\text{proj}}^{(l)} = \{\boldsymbol{W}_{\text{proj}}^{(l,h)}\}_{h=1}^{H}$. To simplify the notation, in the following content we take one block as the example and omit the superscript (layer index).

Heterogeneity of structured sparsity. Because of the difference of the network architecture, the meaning of 'structured sparsity' varies with different model types. As described and performed in (Wen et al. 2016; Anwar, Hwang, and Sung 2017; Liu et al. 2018, 2020), the structured pruning of CNN and RNN typically indicates the removal of the entire channels of the weight tensors and the entire columns of the weight matrices, respectively. Notice that here for either of these two cases, only one type of the structured sparse pattern exist because of the *architectural homogeneity* of the CNN and RNN.

On the other hand, a ViT model exhibits inherent *architectural heterogeneity*. Within the same block, the front-end multi-head attention module and the back-end MLP module represent two types of design philosophy for information processing, and thereby leading to huge difference on both computing procedures and the available structured sparse patterns.

To be specific, when we consider performing structured pruning of ViT model, three types of structured sparse patterns can co-exist with different levels of granularity across different modules. For the multi-head attention module, because each attention head is processing the information individually in a parallel way, the pruning can be performed at the *head-level* to sparsify this component. In addition, consider the weights in the heads are represented in the matrix format; the *column-level* sparsity can also be introduced towards structured pruning. Meanwhile, because the MLP consists of multiple weight matrices as well, the column-level of granularity sparsity can be imposed on this back-end module at the same time. Consequently, a structured pruned ViT model can exhibit heterogeneous structured sparsity (see Fig. 1).

Problem Definition. Based on the above analysis, the structured pruning of a vision transformer model with loss function $\ell(\cdot)$ can be formulated as the following general optimization problem:

$$\begin{aligned} \min_{\boldsymbol{W}_{\mathrm{attn}}, \boldsymbol{W}_{\mathrm{mlp}}} \quad & \ell(\boldsymbol{W}_{\mathrm{attn}}, \boldsymbol{W}_{\mathrm{mlp}}), \\ \text{s.t.} \quad & \|\boldsymbol{W}_{\mathrm{attn}}\|_0^{\mathrm{h}} \leq \kappa_{\mathrm{attn}}^{\mathrm{h}}, \\ & \|\boldsymbol{W}_{\mathrm{attn}}\|_0^{\mathrm{c}} \leq \kappa_{\mathrm{attn}}^{\mathrm{c}}, \\ & \|\boldsymbol{W}_{\mathrm{mlp}}\|_0^{\mathrm{c}} \leq \kappa_{\mathrm{mlp}}^{\mathrm{c}}, \end{aligned} \tag{1}$$

where κ^{c} and κ^{h} are the desired number of columns and the desired number of heads after pruning, respectively. $\|\cdot\|_0^{\mathrm{c}}$ and $\|\cdot\|_0^{\mathrm{h}}$ are the column-based and head-based group L_0-norm, which denote the number of non-zero columns and the number of non-zero heads, respectively.

Questions to be Answered. Solving the above optimization problem is non-trivial since it contains the constraints involved with multi-granularity sparsity for different model components. More specifically, two important questions need to be answered. **Question #1:** *What is the suitable pruning criterion to obtain head-level sparsity?*

Analysis: From the perspective of information processing, multi-head attention shares some interesting similarity with convolutional layer. Both of them use multiple individual computing units, i.e., attention heads and convolutional filters, to perform parallel computations. Therefore, a naive way to perform head-level pruning is to leverage the existing criteria developed in the channel pruning of CNNs. However, such straightforward solution, in principle, may not be the best choice because of two reasons. First, the receptive fields and the focused locality of the attention head and filters are different, and hence simply using the criterion for pruning channels is not a suitable strategy. Second and more importantly, most of the existing channel pruning criterias are built on the information of each individual channel (the corresponding filter weight and/or its feature map). When adopting this philosophy in the head pruning, the insufficient utilization of inter-head information will probably cause non-negligible performance loss. Overall, the unique characteristics of multi-head attention mechanism calls for attention-specific pruning criterion.

Question #2: *How should we coordinate the pruning across different modules with different levels of granularity?*

Analysis: As indicated before, three types of structured sparse pattern can co-exist in the different modules of the pruned ViT models. A key component of the to-be-explored structured pruning strategy is to develop a good coordination scheme that can properly impose these different structured sparse patterns in a joint and global way. Consider the complicated interaction among different types of structured sparsity, the expected pruning strategy should be able to solve this problem in a systematic and global way.

Structured Pruning of ViTs: Method

Graph-based Head Ranking

To answer Question #1, we propose a graph-based approach to measure and determine the importance of different attention heads, which can be further used for the follow-up pruning. **Our key idea** is to model the inter-head correlation as a graph, and then leverage the *graph-based ranking*, a methodology that has been successfully used in many web search and NLP algorithms, such as PageRank (Page et al. 1999), TextRank (Mihalcea and Tarau 2004) and LexRank (Erkan and Radev 2004), to select important attention heads.

Graph Construction of Markov Chain. To be specific, we first construct a graph $G = (\boldsymbol{A}, \boldsymbol{E})$ to represent the attention heads and their similarities in the block of a ViT model. The set of nodes $\boldsymbol{A}$ denote all the attention heads $\{A_h\}_{h=1}^{H}$, and $\boldsymbol{E}$ is the set of connected edges. For edge $E(A_i, A_j)$, its weight is defined as the expected cosine similarity between A_i and A_j. According to (Mihalcea and Tarau 2004), the graph defined with such cosine similarity can be interpreted as a *Markov chain*, where each node is a state, and the transition probability $P(i, j)$ between two states is the edge weight. In such scenario, $P(i, j)$ can be calculated as:

$$P(i,j) = \mathbb{E}_{\boldsymbol{X}\sim\mathcal{D}}\left[\texttt{CosineSim}(A_i(\boldsymbol{X}), A_j(\boldsymbol{X}))\right], \tag{2}$$

where $A_i(\boldsymbol{X})$ is the output of i-th attention head with sampled input $\boldsymbol{X}$ and $\mathcal{D}$ is the data set. Built upon this calculation, the entire transition matrix $\boldsymbol{P}$ of a Markov chain. Notice that as indicated in (Erkan and Radev 2004), each column of $\boldsymbol{P}$ should be further normalized.

Batch estimation. Calculating the transition probability can be very costly since it needs to be performed across the entire training dataset $\mathcal{D}$ (see Eq. 2). To solve this problem, we adopt a batch-based estimation strategy to improve computation efficiency without sacrificing ranking performance. To be specific, as described in Eq. 3, only a batch of training data is sampled and used to to estimate the transition probability. As our ablation study in Section will show, using different batch sizes (B) bring very stable ranking results for the attention heads, thereby empirically verifying the effectiveness of this estimation strategy.

$$P(i,j) = \texttt{CosineSim}\left(\sum_{b=1}^{B} A_i(\boldsymbol{X}_b), \sum_{b=1}^{B} A_j(\boldsymbol{X}_b)\right). \tag{3}$$

Importance Ranking. Mathematically, an irreducible and aperiodic Markov chain is guaranteed to converge to a

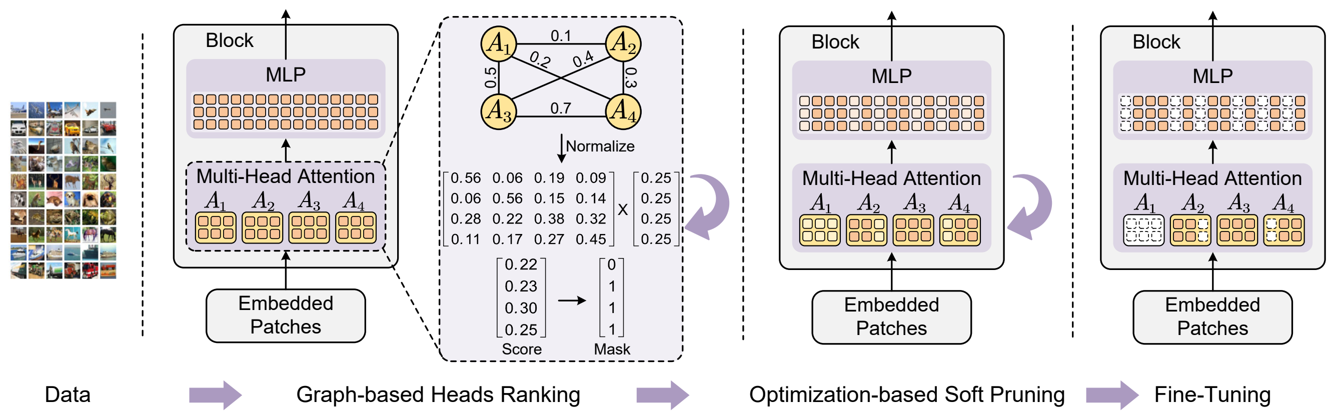

Figure 2: Procedure of the proposed multi-stage structured pruning approach.

stationary distribution (Seneta 2006). As indicated in (Erkan and Radev 2004), once converged, the probability of a random walker stays in one state can reflect the state importance. Motivated by this observation, we propose to quantify the importance of each attention head via calculating the stationary distribution in our constructed Markov chain. To that end, the iterative *power method* (Erkan and Radev 2004) can be used via setting a uniform distribution for the states as the initialization. Overall, the entire graph-based head ranking procedure is described in Algorithm 2.

Soft-Pruning Mask. Once the importance score for each state is obtained via calculating the stationary distribution, the corresponding attention heads can be ranked. Here we use a binary mark matrix $\boldsymbol{M}_{\text{attn}} = \{\boldsymbol{M}_{\text{qkv}}, \boldsymbol{M}_{\text{proj}}\}$ to indicate the weight entries associated with the least important heads that should be removed. Notice that at this stage the head pruning is not performed yet. Instead such binary mask serves as the guideline for the next-stage optimization, and it is essentially integrated into Eq. 1 as follows:

$$\begin{aligned}\min_{\boldsymbol{W}_{\text{attn}},\boldsymbol{W}_{\text{mlp}}} \quad & \ell(\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}) \\ \text{s.t.} \quad & \|(\mathbf{1}-\boldsymbol{M}_{\text{attn}}) \odot \boldsymbol{W}_{\text{attn}}\|_0 = 0, \\ & \|\boldsymbol{W}_{\text{mlp}}\|_0 \le \kappa^{\text{c}}_{\text{mlp}}, \\ & \|\boldsymbol{M}_{\text{attn}} \odot \boldsymbol{W}_{\text{attn}}\|^{\text{c}}_0 \le \kappa^{\text{c}}_{\text{attn}},\end{aligned} \tag{4}$$

where $\odot$ is element-wise product. In general, because the overall optimization phase coordinates and adjusts the different types of structured sparse pattern from a global perspective, this ranking-only "soft" pruning strategy, instead of directly pruning the least important heads, can provide more flexibility and possibility for the next-stage optimization procedure to identify better structured sparse models.

Optimization-based Structured Pruning

As pointed out by Question #2, the co-existence of multi-granularity and multi-location of the sparsity of ViT models make the entire structured pruning procedure become very challenging. To solve this, we propose to use advanced optimization technique to perform systematic structured pruning. To be specific, considering the complicated interactions among different types of structured sparsity, we do not prune the heads or columns immediately, since any direct hard pruning at the early stage may cause severe accuracy loss. Instead, we adopt "soft-pruning" strategy via optimizing the entire ViT models towards the desired structured sparse formats. In other words, the three types of sparsity pattern are gradually imposed onto the attention heads and MLPs.

To that end, we first relax the constraints of Eq. 4 and rewrite it as follows:

$$\begin{aligned}\min_{\boldsymbol{W}_{\text{attn}},\boldsymbol{W}_{\text{mlp}}} \quad & \ell(\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}) + \frac{\lambda}{2}\|(\mathbf{1}-\boldsymbol{M}_{\text{attn}}) \odot \boldsymbol{W}_{\text{attn}}\|^2_F, \\ \text{s.t.} \quad & \|\boldsymbol{W}_{\text{mlp}}\|^{\text{c}}_0 \le \kappa^{\text{c}}_{\text{mlp}}, \\ & \|\boldsymbol{M}_{\text{attn}} \odot \boldsymbol{W}_{\text{attn}}\|^{\text{c}}_0 \le \kappa^{\text{c}}_{\text{attn}},\end{aligned} \tag{5}$$

where λ is the coefficient that controls the influence of quadratic term.

Optimization-based Soft Pruning. As indicated in (Boyd, Parikh, and Chu 2011), when the constraints of continuous non-convex problem are sparsity related (as Eq. 5 shows), Douglas—Rachford splitting method (Eckstein and Bertsekas 1992) can be a suitable optimization solution for such types of problem. Following this philosophy, we first introduce auxiliary variables $\boldsymbol{Z}_{\text{attn}}, \boldsymbol{Z}_{\text{mlp}}$ and indicator functions as:

$$g(\boldsymbol{Z}_{\text{attn}}) = \begin{cases} 0 & \|\boldsymbol{M}_{\text{attn}} \odot \boldsymbol{Z}_{\text{attn}}\|^{\text{c}}_0 \le \kappa^{\text{c}}_{\text{attn}}, \\ +\infty & \text{otherwise}, \end{cases} \tag{6}$$

$$h(\boldsymbol{Z}_{\text{mlp}}) = \begin{cases} 0 & \|\boldsymbol{Z}_{\text{mlp}}\|^{\text{c}}_0 \le \kappa^{\text{c}}_{\text{mlp}}, \\ +\infty & \text{otherwise}. \end{cases} \tag{7}$$

Then, we can rewrite Eq. 5 as the following equivalent form:

$$\begin{aligned}\min_{\boldsymbol{W},\boldsymbol{Z}} \quad & \ell(\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}) + g(\boldsymbol{Z}_{\text{attn}}) + h(\boldsymbol{Z}_{\text{mlp}}) + \\ & \frac{\lambda}{2}\|(\mathbf{1}-\boldsymbol{M}_{\text{attn}}) \odot \boldsymbol{W}_{\text{attn}}\|^2_F, \\ \text{s.t.} \quad & \boldsymbol{W}_{\text{mlp}} = \boldsymbol{Z}_{\text{mlp}}, \\ & \boldsymbol{W}_{\text{attn}} = \boldsymbol{Z}_{\text{attn}}.\end{aligned} \tag{8}$$

In such scenario, the corresponding augmented Lagrangian function of the above optimization objective is:

$$\mathcal{L}_\rho(\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}, \boldsymbol{Z}_{\text{mlp}}) = \ell(\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}) + g(\boldsymbol{Z}_{\text{attn}}) + h(\boldsymbol{Z}_{\text{mlp}}) + \frac{\lambda}{2}\|(\mathbf{1} - \boldsymbol{M}_{\text{attn}}) \odot \boldsymbol{W}_{\text{attn}}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{W}_{\text{attn}} - \boldsymbol{Z}_{\text{attn}} + \boldsymbol{U}_{\text{attn}}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{U}_{\text{attn}}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{W}_{\text{mlp}} - \boldsymbol{Z}_{\text{mlp}} + \boldsymbol{U}_{\text{mlp}}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{U}_{\text{mlp}}\|_F^2, \quad (9)$$

where $\rho > 0$ is the penalty parameter, and $\boldsymbol{U}_{\text{attn}}, \boldsymbol{U}_{\text{mlp}}$ are the Lagrangian multipliers. Then the variables at step t can be iteratively updated as:

$$\boldsymbol{W}_{\text{attn}}^t = \boldsymbol{W}_{\text{attn}}^{t-1} - \eta\frac{\ell(\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}^{t-1})}{\boldsymbol{W}_{\text{attn}}} - \lambda\left[(\mathbf{1} - \boldsymbol{M}_{\text{attn}}) \odot \boldsymbol{W}_{\text{attn}}^{t-1}\right] - \rho(\boldsymbol{W}_{\text{attn}}^{t-1} - \boldsymbol{Z}_{\text{attn}}^{t-1} + \boldsymbol{U}_{\text{attn}}^{t-1}), \quad (10)$$

$$\boldsymbol{W}_{\text{mlp}}^t = \boldsymbol{W}_{\text{mlp}}^{t-1} - \eta\frac{\ell(\boldsymbol{W}_{\text{attn}}^t, \boldsymbol{W}_{\text{mlp}})}{\boldsymbol{W}_{\text{mlp}}} - \rho(\boldsymbol{W}_{\text{mlp}}^{t-1} - \boldsymbol{Z}_{\text{mlp}}^{t-1} + \boldsymbol{U}_{\text{mlp}}^{t-1}), \quad (11)$$

$$\boldsymbol{Z}_{\text{attn}}^t = \boldsymbol{\mathcal{P}}(\boldsymbol{W}_{\text{attn}}^t + \boldsymbol{U}_{\text{attn}}^{t-1}), \quad (12)$$

$$\boldsymbol{Z}_{\text{mlp}}^t = \boldsymbol{\mathcal{P}}(\boldsymbol{W}_{\text{mlp}}^t + \boldsymbol{U}_{\text{mlp}}^{t-1}), \quad (13)$$

$$\boldsymbol{U}_{\text{attn}}^t = \boldsymbol{U}_{\text{attn}}^{t-1} + \boldsymbol{W}_{\text{attn}}^t - \boldsymbol{Z}_{\text{attn}}^t, \quad (14)$$

$$\boldsymbol{U}_{\text{mlp}}^t = \boldsymbol{U}_{\text{mlp}}^{t-1} + \boldsymbol{W}_{\text{mlp}}^t - \boldsymbol{Z}_{\text{mlp}}^t. \quad (15)$$

Here η is the optimizer learning rate for training the ViT, and $\boldsymbol{\mathcal{P}}$ is the Euclidean projection for the sparse constraint.

Final Hard-Pruning and Fine-Tuning. After the above described optimization procedure, the structured sparse patterns have been gradually imposed onto the ViT model. In other words, the weight values of the masked attention heads, as well as some columns of MLPs and attention heads, become extremely small. At this stage, we can now prune those small weights and then perform a few rounds of fine-tuning to achieve higher performance.

Algorithm 1: Overall Procedure of GOHSP Framework

```
Input: Dense weight $\{\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{mlp}}\}$, desired model size $\{\kappa_{\text{attn}}, \kappa_{\text{mlp}}\}$, training data $\mathcal{D}$, number of epochs $E$;
Output: Structured sparse weight $\{\tilde{\boldsymbol{W}}_{\text{attn}}, \tilde{\boldsymbol{W}}_{\text{mlp}}\}$;
 1: Sample a batch of data $\{X_b\}_{b=1}^B$ from $\mathcal{D}$;
 2: Calculate importance score $\boldsymbol{s}$ via Alg. 2;
 3: Obtain structured mask $\boldsymbol{M}_{\text{attn}}$ according to $\boldsymbol{s}$;
 4: $\boldsymbol{Z}_{\text{attn}} := \boldsymbol{W}_{\text{attn}}$, $\boldsymbol{Z}_{\text{mlp}} := \boldsymbol{W}_{\text{mlp}}$; // Initialize auxiliary variables
 5: $\boldsymbol{U}_{\text{attn}} := \mathbf{0}$, $\boldsymbol{U}_{\text{mlp}} := \mathbf{0}$; // Initialize Lagrangian multipliers
 6: for $e = 1$ to $E$ do
 7:     Update $\boldsymbol{W}_{\text{attn}}, \boldsymbol{W}_{\text{attn}}$ via Eq. 10 and Eq. 11;
 8:     Update $\boldsymbol{Z}_{\text{attn}}, \boldsymbol{Z}_{\text{mlp}}$ via Eq. 12 and Eq. 13;
 9:     Update $\boldsymbol{U}_{\text{attn}}, \boldsymbol{U}_{\text{mlp}}$ via Eq. 14 and Eq. 15;
10: Fine-tune pruned weight $\{\tilde{\boldsymbol{W}}_{\text{attn}}, \tilde{\boldsymbol{W}}_{\text{mlp}}\}$.
```

Algorithm 2: Graph-based Attention Head Ranking

```
Input: Sampled batch $\{\boldsymbol{X}_b\}_{b=1}^B$, attention heads $\{A_h\}_{h=1}^H$;
Output: Importance score $\boldsymbol{s} = [s_1, \cdots, s_H]$.
 1: Initialize transition matrix: $\boldsymbol{P} := \texttt{zeros}(H, H)$;
 2: for $i = 1$ to $H$ do
 3:     for $j = 1$ to $H$ do
 4:         Calculate $P(i, j)$ via Eq. 3;
 5: Normalize each column of $\boldsymbol{P}$;
 6: Initialize $\boldsymbol{s} := \texttt{ones}(H)/H$;
 7: repeat
 8:     $\boldsymbol{s}' := \boldsymbol{s}$;
 9:     $\boldsymbol{s} := \boldsymbol{P}s$;
10:     $\delta := \|\boldsymbol{s} - \boldsymbol{s}'\|$;
11: until $\delta \leq \epsilon$
```

Method	**Sparsity**	**# Paramters**	**Top-1 (%)**
Baseline	-	48.0M	97.85
SOMP	40%	28.8M	96.07
SGMP	40%	28.8M	96.93
GOHSP (Ours)	40%	28.8M	**97.89**
GOHSP (Ours)	80%	9.6M	97.40

Table 1: Performance comparison between our GOHSP and structured one-shot/gradually magnitude-based pruning (SOMP/SGMP) of ViT-Small model on CIFAR-10 dataset.

Overall, by using graph-based head ranking and optimization-based structured pruning, the previously raised Question #1 and #2 can be properly addressed. The overall GOHSP framework is summarized in Fig. 2.

Experiments

Experimental Settings

Dataset and Baseline. We evaluate the performance of our proposed GOHSP approach on CIFAR-10 and ImageNet datasets (Deng et al. 2009). For experiments on the CIFAR-10 dataset, the original dense model is ViT-Small[1](Dosovitskiy et al. 2020) with 48M parameters. For experiments on the ImageNet dataset, the original dense models are DeiT-Tiny and DeiT-Small (Touvron et al. 2021) with 5.7M and 22.1M parameters, respectively.

Hyper-parameters and Sparsity Ratio. For our experiments on the CIFAR-10 dataset, the batch size, learning rate and ρ are set as 256, 0.1 and 0.001, respectively. For ImageNet dataset, the batch size, learning rate and ρ are set as 256, 0.01 and 0.001, respectively. For both of these two experiments, SGD is selected as the training optimizer without using weight decay, and we apply Erdős-Rényi (Mocanu et al. 2018) to determine the sparsity distribution of each layer given an overall sparsity ratio. In particular, soft-pruning maintains high accuracy at the high sparsity ratios.

[1]We take this model from open source library timm.

Model	Method	Sparsity	# Parameters	FLOPs ↓	Run-time ↓	Top-1 (%)
DeiT-Tiny	Baseline	-	5.7M	-	-	72.20
	OMP (Unstructured)	30%	4.02M	25.56%	-	68.35
	GMP (Unstructured)	30%	4.02M	25.56%	-	69.56
	TP (Unstructured)	30%	4.02M	25.56%	-	68.38
	SSP (Structured)	30%	4.2M	23.69%	-	68.59
	S^2ViTE (Structured)	30%	4.2M	23.69%	10.57 %	70.12
	GOHSP (Structured)	30%	**4.0M**	**30%**	**13.41%**	**70.24**
DeiT-Small	Baseline	-	22.1M	-	-	79.90
	SSP (Structured)	40%	14.6M	31.63%	-	77.74
	S^2ViTE (Structured)	40%	14.6M	31.63%	22.65%	79.22
	GOHSP (Structured)	40%	14.4M	35%	24.61%	**79.98**
	GOHSP (Structured)	50%	11.1M	39%	**26.57%**	79.86

Table 2: Comparison results of our method, GOHSP, with other structured and unstructured pruning methods on ImageNet.

Performance Evaluation

CIFAR-10 Dataset. Table 1 shows performance comparison on CIFAR-10 dataset between our proposed GOHSP and other structured pruning method (structured one-shot magnitude pruning (SOMP) (Han, Mao, and Dally 2015) and structured gradually magnitude pruning (SGMP) (Zhu and Gupta 2017)) for ViT-Small model. It is seen that with the same sparsity ratio, our approach brings significant performance improvement. Compared to SGMP approach, GOHSP achieves 0.96% accuracy increase with the same pruned model size. Even compared with the baseline, the structured sparse model pruned by GOHSP can outperform the uncompressed model with 40% fewer parameters while 80% compressed model achieves only 0.45% worse than the full ViT-Small model.

ImageNet Dataset. Table 2 summarizes the performance on ImageNet dataset between GOHSP and other structured pruning approaches (SOMP, SGMP, Talyer pruning (TP), Salience-based Structured Pruning (SSP) and S^2ViTE(Chen et al. 2021)) for DeiT-Tiny and DeiT-Small models. It is seen that due to the limited redundancy in such small-size model, the existing pruning approaches suffer from more than 2.5% accuracy loss when compressing DeiT-Tiny. Instead, with the even fewer parameters and more FLOPs reduction, our GOHSP approach can achieve at least 0.68% accuracy increase over the unstructured pruning approaches. Compared to the structured pruning approach (SSP), our method enjoys 1.65% accuracy improvement with lower storage cost and computational cost. In addition, when compressing DeiT-Small model, with fewer parameters and more FLOPs reduction, our GOHSP approach can achieve 0.76% accuracy increase as compared to the state-of-the-art structured pruning method S^2ViTE (Chen et al. 2021) and can even outperform the original DeiT-Small. With 50% pruned DeiT-Small we achieve similar accuracy to the full DeiT-Small. Finally, we report 26.57% improvement in run-time efficiency with our 50% pruned DeiT-Small.

Ablation Study, Visualization and Discussion

To obtain the deep understanding of the effect of our proposed approach, we perform several ablation studies and a detailed analysis. Here the experiments conducted in the ablation study focus on compressing ViT-Small on CIFAR-10.

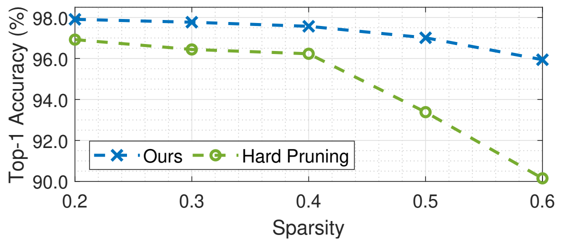

Figure 3: Results on the effect of soft-pruning (ours) and hard-pruning for ViT-Small model on CIFAR-10 dataset.

Soft Pruning vs Hard Pruning. As described in Optimization section, after ranking the attention heads, we use the ranking information as a soft-pruning mask to guide the next-phase optimization. The optimization itself is also

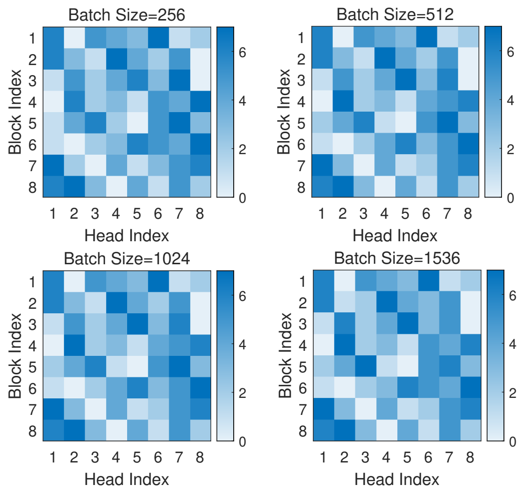

Figure 4: The effect of batch sizes for ranking results. Different colors represent different ranking scores. We can see that our head ranking algorithm is not sensitive to batch size.

a soft-pruning procedure that does not directly zero the weights but gradually impose the structured sparsity. To analyze the effect of this strategy, we conduct an ablation experiment via performing the direct hard pruning. In this ablation study, the least important attention heads are removed according to their ranks, and the columns of MLPs with least group L_1 norm are also pruned. Such hard pruned models are still trained with the same hyper-parameters settings that are used for soft pruning method. Fig. 3 shows the curves of top-1 test accuracy with different target sparsity settings. The soft-pruning strategy brings very significant accuracy improvement over the direct hard pruning with the same sparsity ratio.

Effect of Batch Size on Head Ranking. As shown in Eq. 3, the importance scores of attention head is calculated on a batch of data. To investigate the potential impact of batch sizes for the ranking results, we observe the change of ranking with different batch sizes. As shown in Fig. 4, the ranking results are very stable (almost the same) when the batch size varies. Therefore we can conclude that using batches of data can already achieve very good estimation of head ranking. In other words, our ranking approach has low sensitivity to the distribution of input data.

Sensitivity of Penalty Parameter ρ. We also explore the effect of hyperparameter ρ on the structured pruning procedure. Fig. 5 (a) shows the convergence of training process with respect to different ρ. It is seen that the convergence speed is always fast, and hence it demonstrates the promising convergence property of our approach in practice. Fig. 5 (b) illustrates the L_2-norm of the masked entries. It is seen that the larger ρ makes the model exhibit more sparsity at the earlier stage, thereby indicating that larger ρ can bring fewer epochs in the final fine-tuning stage. However, as shown in Fig. 5 (c), too large ρ brings accuracy degradation, so ρ can be considered as a parameter that controls the trade-off between the speed of imposing sparsity and task performance.

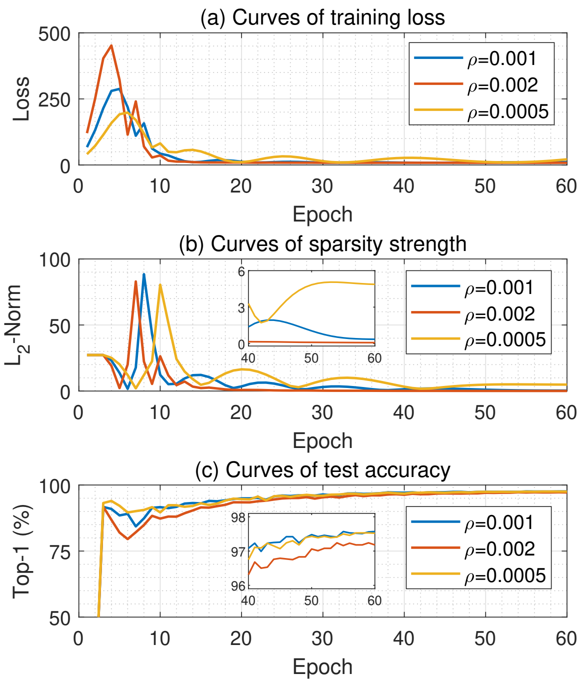

Figure 5: Effect of ρ on the structured pruning procedure. ρ controls the trade-off between the speed of imposing sparsity and task performance.

Visualization. Fig. 6 illustrates the sparsity patterns in the pruned ViT models after performing our GOHSP approach. It is seen that three types of structured sparsity patterns (head-level sparsity, column-level sparsity in the head and column-level sparsity in the MLP) are imposed on the pruned models. Such pruning can be more effective on hardware than the unstructured pruning methods.

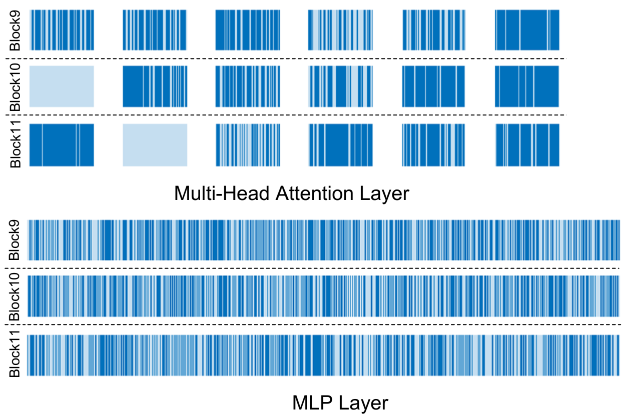

Figure 6: Visualization of the imposed structured sparsity on the DeiT-Small model. The columns and heads with lighter color are pruned. Our method can prune columns (Block9, Block10, and Block11), and heads (Block10, Block11) of the Multi-Head Attention layer. On the other hand, we can prune columns of MLP layers in all the blocks.

Why Douglas—Rachford splitting method? 1) Convergence: According to (Boyd, Parikh, and Chu 2011), within a few iterations it can provide satisfied solution for large-scale problems – particularly attractive for DNN applications. More specifically for this work, the fast convergence of Douglas—Rachford splitting method can avoid gradient explosion problem introduced by the additional sparsity loss in Eq. 9. 2) Flexibility: Douglas—Rachford splitting method, by its nature, divides the original difficult optimization problem into several less complicated sub-problems, each of which can be then addressed independently.

Conclusion

In this paper we propose GOHSP, a unified framework to perform graph and optimization-based heterogeneous structured pruning for vision transformers. By using graph-based ranking and leveraging the advanced optimization technique, our approach can efficiently impose different types of structured sparse patterns on the vision transformers with high compression rate and task performance.

References

Anwar, S.; Hwang, K.; and Sung, W. 2017. Structured pruning of deep convolutional neural networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 13(3): 1–18.

Bakhtiarnia, A.; Zhang, Q.; and Iosifidis, A. 2021. Single-Layer Vision Transformers for More Accurate Early Exits with Less Overhead. *arXiv preprint arXiv:2105.09121*.

Boyd, S.; Parikh, N.; and Chu, E. 2011. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

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

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Eckstein, J.; and Bertsekas, D. P. 1992. On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1): 293–318.

Erkan, G.; and Radev, D. R. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22: 457–479.

Han, S.; Mao, H.; and Dally, W. J. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

He, Y.; Liu, P.; Wang, Z.; Hu, Z.; and Yang, Y. 2019. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4340–4349.

He, Y.; Zhang, X.; and Sun, J. 2017. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, 1389–1397.

Kim, Y.-D.; Park, E.; Yoo, S.; Choi, T.; Yang, L.; and Shin, D. 2016. Compression of deep convolutional neural networks for fast and low power mobile applications. In *International Conference on Learning Representations*.

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

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

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

Liu, Z.; Sun, M.; Zhou, T.; Huang, G.; and Darrell, T. 2018. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*.

Lou, Q.; Hsu, Y.-C.; Uzkent, B.; Hua, T.; Shen, Y.; and Jin, H. 2022. Lite-MDETR: A Lightweight Multi-Modal Detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12206–12215.

Meng, L.; Li, H.; Chen, B.-C.; Lan, S.; Wu, Z.; Jiang, Y.-G.; and Lim, S.-N. 2022. AdaViT: Adaptive Vision Transformers for Efficient Image Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12309–12318.

Mihalcea, R.; and Tarau, P. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 404–411.

Mocanu, D. C.; Mocanu, E.; Stone, P.; Nguyen, P. H.; Gibescu, M.; and Liotta, A. 2018. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, 9(1): 1–12.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Pan, Y.; Xu, J.; Wang, M.; Ye, J.; Wang, F.; Bai, K.; and Xu, Z. 2019. Compressing recurrent neural networks with tensor ring for action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4683–4690.

Rao, Y.; Zhao, W.; Liu, B.; Lu, J.; Zhou, J.; and Hsieh, C.-J. 2021. DynamicViT: Efficient Vision Transformers with Dynamic Token Sparsification. *arXiv preprint arXiv:2106.02034*.

Seneta, E. 2006. *Non-negative matrices and Markov chains*. Springer Science & Business Media.

Tiwari, R.; Bamba, U.; Chavan, A.; and Gupta, D. 2021. ChipNet: Budget-Aware Pruning with Heaviside Continuous Approximations. In *International Conference on Learning Representations*.

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

Uzkent, B.; and Ermon, S. 2020. Learning when and where to zoom with deep reinforcement learning. In *Proceedings* *of the IEEE/CVF conference on computer vision and pattern recognition*, 12345–12354.

Uzkent, B.; Yeh, C.; and Ermon, S. 2020. Efficient object detection in large images using deep reinforcement learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 1824–1833.

Wang, Y.; Huang, R.; Song, S.; Huang, Z.; and Huang, G. 2021. Not All Images are Worth 16x16 Words: Dynamic Vision Transformers with Adaptive Sequence Length. *arXiv preprint arXiv:2105.15075*.

Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2016. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29: 2074–2082.

Xu, Y.; Zhang, Z.; Zhang, M.; Sheng, K.; Li, K.; Dong, W.; Zhang, L.; Xu, C.; and Sun, X. 2022. Evo-vit: Slow-fast token evolution for dynamic vision transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 2964–2972.

Ye, J.; Lu, X.; Lin, Z.; and Wang, J. Z. 2018. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. *arXiv preprint arXiv:1802.00124*.

Yu, R.; Li, A.; Chen, C.-F.; Lai, J.-H.; Morariu, V. I.; Han, X.; Gao, M.; Lin, C.-Y.; and Davis, L. S. 2018. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9194–9203.

Yu, S.; Chen, T.; Shen, J.; Yuan, H.; Tan, J.; Yang, S.; Liu, J.; and Wang, Z. 2022. Unified Visual Transformer Compression. In *International Conference on Learning Representations*.

Yu, X.; Liu, T.; Wang, X.; and Tao, D. 2017. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7370–7379.

Zhai, X.; Kolesnikov, A.; Houlsby, N.; and Beyer, L. 2021. Scaling vision transformers. *arXiv preprint arXiv:2106.04560*.

Zhou, D.; Kang, B.; Jin, X.; Yang, L.; Lian, X.; Jiang, Z.; Hou, Q.; and Feng, J. 2021. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*.

Zhu, M.; and Gupta, S. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.

Zhuang, Z.; Tan, M.; Zhuang, B.; Liu, J.; Guo, Y.; Wu, Q.; Huang, J.; and Zhu, J. 2018. Discrimination-aware channel pruning for deep neural networks. In *Advances in Neural Information Processing Systems*, 875–886.