HFSP: A HARDWARE-FRIENDLY SOFT PRUNING FRAMEWORK FOR VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recently, Vision Transformer (ViT) has continuously established new milestones in the computer vision field, while the high computation and memory cost makes its propagation in industrial production difficult. Pruning, a traditional model compression paradigm for hardware efficiency, has been widely applied in various DNN structures. Nevertheless, it stays ambiguous on how to perform exclusive pruning on the ViT structure. Considering three key points: the structural characteristics, the internal data pattern of ViT, and the related edge device deployment, we leverage the input token sparsity and propose a hardware-friendly soft pruning framework (HFSP), which can be set up on vanilla Transformers of both flatten and CNN-type structures, such as Pooling-based ViT (PiT). More concretely, we design a dynamic attention-based multi-head token selector, which is a lightweight module for adaptive instance-wise token selection. We further introduce a soft pruning technique to package the pruned tokens, which integrate the less informative tokens generated by the selector module into a package token, and participates in subsequent calculations rather than being discarded completely. From a hardware standpoint, our framework is bound to the tradeoff between accuracy and specific hardware constraints through our proposed hardware-oriented progressive training, and all the operators embedded in the framework have been well-supported. Experimental results demonstrate that the proposed framework significantly reduces the computational costs of ViTs while maintaining comparable performance on image classification. For example, our method reduces the FLOPs of DeiT-S by over 42.6% while only sacrificing 0.46% top-1 accuracy. Moreover, our framework can guarantee the identified model to meet resource specifications of mobile devices and FPGA, and even achieve the real-time execution of DeiT-T on mobile platforms. Code will be publicly released.

1 INTRODUCTION

Recently, a new trend of leveraging Transformer architecture (Vaswani et al., 2017) into the computer vision domain has emerged (Hudson & Zitnick, 2021; Chen et al., 2021g; Kim et al., 2021; Deng et al., 2021; Xue et al., 2021; Zhao et al., 2021; Guo et al., 2021; Srinivas et al., 2021). The Vision Transformer (ViT), which solely exploits the self-attention mechanism that inherits from the Transformer architecture, has set up many state-of-the-art (SOTA) records in image classifications (Dosovitskiy et al., 2020; Touvron et al., 2021; Chen et al., 2021b), object detection (Carion et al., 2020; Dai et al., 2021; Amini et al., 2021; Misra et al., 2021), tracking (Chen et al., 2021i; Yan et al., 2021; Meinhardt et al., 2021), semantic segmentation (Zheng et al., 2021; Cheng et al., 2021), depth estimation (Yang et al., 2021b; Li et al., 2020b), image retrieval (El-Nouby et al., 2021a), and image enhancement (Yang et al., 2020; Chen et al., 2021c; Lu et al., 2021). However, despite the impressive general results, ViTs have sacrificed lightweight model capacity, portability, and trainability in return for high accuracy. The mass number of computation layers (e.g., Conv, MatMul, Softmax, GeLU, Add) of existing models remains a setback for edge device deployment.

Pruning, as one of the most straightforward and effective methods to reduce network dimensions, is thoroughly explored in convolution-based neural networks (Han et al., 2015; Liu et al., 2017; Ren et al., 2018), yet its application in self-attention-based neural networks remain scarce (Guo et al., 2020; Sanh et al., 2020; Li et al., 2020a; Wang et al., 2021a). Currently, some pioneering works are exploring ViT pruning. However, there still exists a gap between the actual device deployment and

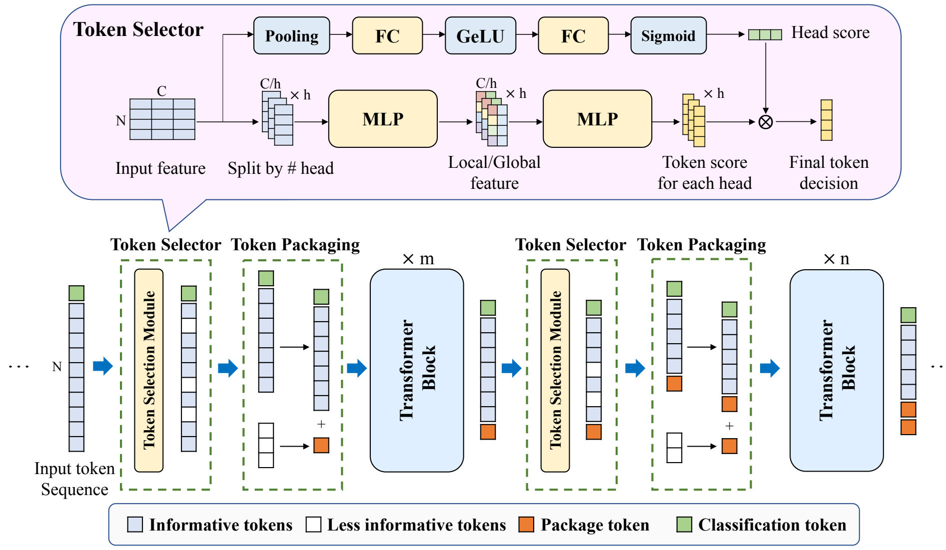

Figure 1: Overall workflow. Upper Figure: Our attention-based multi-head token selector to obtain token scores for keep/drop decisions. Lower Figure: Token selector is inserted multiple times throughout the model, along with the token packaging technique to generate a package token from the less informative tokens. The package token is concatenated with the informative tokens to be fed in the following transformer blocks.

acceleration in their frameworks. For instance, attention head pruning (Chen et al., 2021f) performs weight pruning on the transformation matrix (W_Q, W_K, W_V) before the multi-head self-attention (MSA) operation. It is an inefficient way for computation reduction, because only part of the ViT computations (i.e., MSA) can be alleviated (see Section 3 for justification). In a lightweight model, head pruning cannot guarantee an ideal pruning rate without significant accuracy degradation. Static token pruning (Rao et al., 2021) reduces the number of input tokens by a fixed ratio for different images, which restricts the image pruning rate, ignoring the fact that the high-level information of each image varies both in the region size and location. It is also difficult for the deployment on edge devices since newly introduced operations (e.g., Argsort) are currently not well supported by many frameworks (Prillo & Eisenschlos, 2020). In contrast, dynamic token pruning (Pan et al., 2021) deletes redundant tokens based on the inherent image characteristics to achieve per-image adaptive pruning rate. However, this method implies a potentially huge search space, which will easily cause limited overall pruning rate or undermined accuracy if the token selection mechanism is not carefully designed.

In this paper, we manage to overcome the above limitations. Specifically, we propose HFSP as shown in Figure 1, a Hardware-Friendly Soft Pruning framework which simultaneously optimizes ViT accuracy and maximizes per-image dynamic pruning rate, while maintaining actual deployment efficiency on edge devices. In ViT, each head encodes the visual receptive field independently (Pan et al., 2021; Heo et al., 2021; Mao et al., 2021), which implies that each token has a different influence in different heads (Dosovitskiy et al., 2020; Zhai et al., 2021; Yu et al., 2021; Gao et al., 2021). We thus propose a token selection module to evaluate the importance score of each token based on its characteristic statistics in all heads. Then, through the attention-based branch (Hu et al., 2018) in the selection module, we sum up the final score of a token, which determines whether the token should be pruned. With the selection module, all tokens generated from the input images can be precisely ranked and pruned based on their importance scores and thus achieving a high overall pruning rate. However, the token representations (Wu et al., 2020; Xu et al., 2021a; Chen et al., 2021e; Chefer et al., 2021) in shallow or middle layers are insufficiently encoded as shown in Figure 6 (see Appendix), which makes token pruning quite difficult. And this technique soft the pruning process, because the pruned tokens have not been deleted totally. To mitigate the challenge, we introduce a package token technique, which compresses the less-informative tokens, picked out and to be pruned by the selection module, into a package token. Then, we concatenate the package token to other remaining tokens for subsequent blocks. On one hand, although informative tokens may be discarded due to the poor encoding ability in earlier blocks of ViT (Xu et al., 2021b), this error will be partly corrected by the residual information stored in the package token. On the other hand, background features can help emphasize foreground features (Yang et al., 2021a). Completely removing less informative (negative) tokens will weaken self-attention's ability to capture key information. Therefore, the package token can serve as a way to help preserve background features. By adding minimal computational costs, the token pruning rate will be increased significantly.

Taking the hardware efficiency into consideration, all the operators contained in our framework have been well-supported on edge devices. In addition, we elaborate a hardware-oriented progressive training, which consists of two parts: hardware-constraint loss function and layer-to-phase progressive training. The former bridges the token pruning rates with operating constraints of diverse edge devices. The latter indicates that we progressively insert one selector in each block, and train the new selector under the resource budget constraints of the target device. Next, we group adjacent blocks with similar pruning rates into a phase, keep the first selector in this phase and remove others. While maintaining high accuracy, it can search for the appropriate pruning rate for each block and the desirable insertion position of the selector. Our contributions are summarized as follows:

- We provide a detailed analysis on the computational complexity of ViT and different compression strategies. Based on our analysis, pruning tokens holds a greater computation reduction compared to compression of other dimensions.
- Considering the vision pattern inside ViT, we propose HFSP, which includes the attention-based multihead selection module and the token packaging technique to achieve per-image adaptive pruning. We also design a hardware-oriented progressive training, which efficiently explores the HFSP design space given the hardware resource budget, and maximize the per-image pruning rate without accuracy degradation.
- HFSP enables a higher pruning rate than other state-of-the-art with comparable accuracy. By applying HFSP to PiTs, more efficient and accurate models are generated compared with the embedding dimension scaling (Touvron et al., 2021) of original models.
- To the best of our knowledge, it is the first time that the ViT models perform inference on the edge devices, and even beyond real-time[1] for a DeiT-T on mobile phones and DeiT-S on a Xilinx FPGA (e.g., 32 ms on a Samsung Galaxy S20 and 13.2 ms on an Xilinx ZCU102 FPGA).

2 Related Work

Vision Transformers. ViT (Dosovitskiy et al., 2020) is a pioneering work that uses only Transformer to solve various vision tasks. Compared to traditional CNN structures, ViT allows all the positions in an image to interact through transformer blocks whereas CNNs operated on a fixed-sized window with restricted spatial interactions, which can have trouble capturing relations at the pixel level in both spatial and time domains (Raghu et al., 2021). Since then, many variants have been proposed (Graham et al., 2021; Liu et al., 2021b; Yuan et al., 2021a; Wang et al., 2021c; Han et al., 2021; Wu et al., 2021; Chen et al., 2021d; Steiner et al., 2021; El-Nouby et al., 2021b; Liu et al., 2021a; Wang et al., 2021b; Bao et al., 2021). For example, DeiT (Touvron et al., 2021),T2T-ViT (Yuan et al., 2021b) and Mixer (Chen et al., 2021h) tackle the data-inefficiency problem in ViT by training only with ImageNet. PiT (Heo et al., 2021) replaces the uniform structure of Transformer with depth-wise convolution pooling layer to reduce spacial dimension and increase channel dimension. LV-ViT (Jiang et al., 2021) introduces a token labeling approach to improve training. PS-ViT (Yue et al., 2021) abandons the fixed length tokens with progressive sampled tokens.

Efficient ViT. The huge memory usage and computation cost of the self-attention mechanism serve as the roadblock to the efficient deployment of the ViT model on edge devices. Many works aim at accelerating the inference speed of ViT (Chen et al., 2021a). For instance, S^{2}ViTE (Chen et al., 2021f) prunes token and attention head in a structured way via sparse training. VTP (Zhu et al., 2021) reduces the input feature dimension by learning their associated importance scores with L1 regularization. IA-RED^2 (Pan et al., 2021) drops redundant tokens with a multi-head interpreter. PS-ViT (T2T) (Tang et al., 2021) discards useless patches in a top-down paradigm. DynamicViT (Rao et al., 2021) removes redundant tokens by estimating their importance score with a MLP (Vaswani et al., 2017) based prediction module. Evo-ViT (Xu et al., 2021b) develops a slow-fast token evolution method to preserve more image information during pruning. Token-

[1]Real-time inference usually means 30 frames per second, which is approximately 33 ms / image.

Table 1: Computation complexity of ViT. The input $N \times D_{ch}$ goes through three linear transformation layers with $D_{ch} \times D_{attn}$ to generate Query (Q), Key (K), and Value (V) matrices of size $N \times D_{attn}$. N is transitive, while D_{ch} is not.

#	Method	Input Size	Operation	Layer Size	Output Size	Computation
①	MSA	$N \times D_{ch}$	Linear Transformation	$D_{ch} \times D_{attn}$	$N \times D_{attn}$	$ND_{ch}D_{attn} \times 3$
②		$N \times D_{attn}$	Q Multiplying K^T	-	$N \times N$	$N^2 D_{attn}$
③		$N \times N$	Multiplying V	-	$N \times D_{attn}$	$N^2 D_{attn}$
④		$N \times D_{attn}$	Projection	$D_{attn} \times D_{ch}$	$N \times D_{ch}$	$ND_{attn}D_{ch}$
⑤	FNN	$N \times D_{ch}$	FC Layer	$D_{ch} \times 4D_{fc}$	$N \times 4D_{fc}$	$4ND_{ch}D_{fc}$
⑥		$N \times 4D_{fc}$	FC Layer	$4D_{fc} \times D_{ch}$	$N \times D_{ch}$	$4ND_{fc}D_{ch}$
Total Computation Complexity						$4ND_{ch}D_{attn}+$ $2N^2 D_{attn} + 8ND_{ch}D_{fc}$

Learner (Ryoo et al., 2021) uses spatial attention to generate a small set of token vectors adaptive to the input. However, to the best of our knowledge, our idea of considering actual edge device deployment and acceleration has not been investigated by any existing ViT pruning approaches.

3 Computational Complexity Analysis

Given an input sequence $N \times D$, where N is the input sequence length or token number and D is the embedding dimension (Touvron et al., 2021) of each token, some works (Pan et al., 2021; Zhu et al., 2021) address the computational complexity of ViT as $(12ND^2 + 2N^2D)$. However, D represents different dimensions and should be written as $(4ND_{ch}D_{attn}+2N^2D_{attn}+8ND_{ch}D_{fc})$. Neglecting the difference may cause misleading conclusions, especially when analyzing the validity of pruning methods such as token pruning and dimension pruning.

Table 1 shows an analysis of each operation in a Transformer block. There are three main branches on ViT pruning. (i) Token channel pruning: The sequence tokens are pruned along D_{ch} dimension. D_{ch} is non-transmissible, which means reducing input dimension only affects the computation of the current matrix multiplication. To reduce computation for all layers, a mask layer is added to multiply with the input before going through the linear layer (Zhu et al., 2021). (ii) Token pruning: N is transitive, so directly pruning tokens will contribute to the linearly or even quadratically (N^2 in ② and ③) reduction of all operations. (iii) Attention head pruning (or attention channel pruning): The pruning operations are performed on weight tensors of each attention head in the MSA module. However, only the D_{attn} in the MSA module can be counted towards computation reduction, which usually contributes less than 40% of the total computation in most ViT architectures. Therefore, with the same pruning ratio, pruning tokens (reducing N) can reduce more overall computation than pruning channels (reducing D_{ch} or D_{attn}).

4 The Hardware-friendly Soft Pruning Framework

In this section, we first introduce our hardware-friendly soft token pruning (a.k.a HFSP) framework. Then, we show an elaborate design of the HFSP modules. Finally, we give a detailed discussion of our hardware-oriented progressive training strategy.

4.1 Framework Overview

Our soft pruning framework includes a token selection module and token packaging technique. We propose a hierarchical pruning scheme, where these two modules are inserted between multiple blocks throughout the model. As shown in Figure 1, the input token sequence first goes through a token selection module (selector), where each token is scored and defined as either informative or less informative. After that, less informative tokens are separated from the sequence and integrated into a package token. This package token then concatenates to the informative tokens to involve into subsequent calculations in the blocks. In the next phase, a newly generated package token will connect with the existing package token.

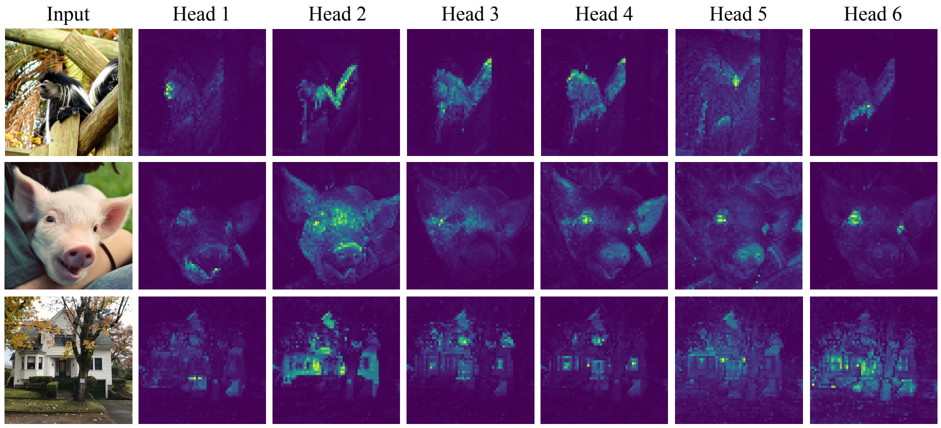

Figure 2: Heatmaps showing the informative region detected by each head in DeiT-S. Each attention head focuses on encoding different image features and visual receptive fields. Tool refers to (Caron et al., 2021).

For ViT training with HFSP framework, we devise a hardware-constraint sparsity loss for the hardware's maximum computation bandwidth. We perform a layer-to-phase progressive training schedule to compress the search space, where model accuracy optimization and hardware computation reduction can be simultaneously achieved. The overall framework is hardware friendly with no unsupported operations and miniature computation cost.

4.2 Attention-based Multi-head Token Selection Module

Multi-head Token Selector. We propose a fine-grained approach to evaluate token scores. As shown in Figure 2, in ViT's multi-head vision pattern, each head focus on encoding different features and respective fields of an image. This implies that the importance of each token towards each head is different. Our multi-head selector generates a list of token scores for each head. Let one head dimension be $S = C/H$, where C is the input dimension and H is the number of head. We split the input $X \in \mathbb{Z}^{N \times C}$ by attention head $[x_1, x_2, ..., x_i], 1 < i < H$, and obtain local and global features through an MLP layer separately:

$$f_i = [f_i^{local}, f_i^{global}] = \mathrm{MLP}(x_i) \quad \in \mathbb{Z}^{N \times S}, \quad 1 < i < H \tag{1}$$

The combined feature then passes through a MLP layer to produce token score maps with t_i indicating the token score from each attention head:

$$t_i = \mathrm{Softmax}(\mathrm{MLP}(f_i)) \quad \in \mathbb{Z}^{N \times 2}, \quad T = [t_1, t_2, ..., t_i], \quad 1 < i < H \tag{2}$$

Head Importance Score. We merge the score maps by the weights of each attention head. As shown in Figure 1, we add an attention-based branch along the selector backbone to synthesis the importance of each head:

$$\bar{X} = \mathrm{AvgPool}(X) = \frac{1}{S} \sum_{m=1}^{S} X(m) \quad \in \mathbb{Z}^{N \times H}, \tag{3}$$

$$A = \mathrm{Sigmoid}(\mathrm{FC}(\mathrm{GeLU}(\mathrm{FC}(\bar{X})))) \quad \in \mathbb{Z}^{N \times H}, \tag{4}$$

where $\bar{X}$ is a head-wise statistic generated by shrinking X through its channel dimension C with global average pooling. In Equation (4), the attention head score vector A is obtained by feeding $\bar{X}$ into the FC $\rightarrow$ GeLU $\rightarrow$ FC $\rightarrow$ Sigmoid pipeline to fully capture head-wise dependencies. The overall token score is calculated by adding all the individual scores from each head, multiplying by their attention head score $A = [a_1, a_2, ..., a_i], 1 < i < H$:

$$\tilde{T} = \frac{\sum_{i=1}^{H} t_i * a_i}{\sum_{i=1}^{H} a_i} \quad \in \mathbb{Z}^{N \times 2}, \tag{5}$$

where $\tilde{T}$ is the final token score and a_i is the individual head importance score. We apply the Gumbel-Softmax technique to generate the keep decision D for input tokens. Our module is hardware friendly with miniature computation cost (less than 1% of the total model FLOPs).

4.3 TOKEN PACKAGING TECHNIQUE

As discussed before, ViT is less accurate for evaluating token values in shallow blocks. Poor scoring may cause important tokens to be removed. Also, completely removing background (negative) tokens will weaken self-attention's ability to capture key information (Yang et al., 2021a). Instead of completely discarding tokens that are considered less informative, we apply a token packaging technique that integrates them into a package token. Assume there are L less informative tokens $\hat{X}$, along with their token scores $\hat{\hat{T}}$:

$$\hat{X} = [\hat{x}_1, \hat{x}_2, ..., \hat{x}_l] \quad \in \mathbb{Z}^{L \times C}, \quad \hat{\hat{T}} = [\hat{t}_1, \hat{t}_2, ..., \hat{t}_l] \quad \in \mathbb{Z}^{L \times 2}, \quad 1 < i < L \tag{6}$$

these tokens are combined to one token by:

$$P = \text{Package}(\hat{X}) = \frac{\sum_{i=1}^{L} \hat{x}_i * \hat{t}_i}{\sum_{i=1}^{L} t_i} \quad \in \mathbb{Z}^{1 \times C}, \tag{7}$$

where P is the package token; x_i is an individual token; t_i is its corresponding score. Token P will take part in subsequent calculations along with the informative tokens, giving the model an ability to correct scoring mistakes. All the operations in our framework (MLP, Softmax, Pooling, Sigmoid) are well supported on edge platforms.

4.4 HARDWARE-ORIENTED PROGRESSIVE TRAINING

Our framework mainly trains two structures, i.e., one is the training on the token selection module, which promotes them to produce the token pruning rates that favor the target edge device; The other is fine-tuning the backbones, which enables them to accommodate the pruning rates and maintain the inherent accuracy. Our training objectives include the standard cross-entropy loss, soft distillation loss, and hardware-constraint sparsity loss. The former two are the same as the loss strategy used in DeiT (Touvron et al., 2021).

Hardware-constraint Sparsity Loss. In order to bridge the efficient inference of ViT model produced by HFSP framework to the actual hardware deployment, we introduce a novel hardware-constraint sparsity loss:

$$\text{Block}_* = 12NC^2 + 2N^2C, \qquad \text{Selector}_* = \frac{5}{8}NC^2 + \frac{1}{2}NC, \tag{8}$$

$$\sum_{l=1}^{L} (\text{Block}_l(\rho_l, N) + I_l \cdot \text{Selector}_l(\rho_l, N)) \leq \gamma \text{HardwareCost}, \tag{9}$$

where Equation (8) shows the amount of computation of a single ViT block and selector, in which N is the token number and C is the token dimension; Equation (9) constrains the degree of model computation reduction, where l is the block index, ρ_l is the pruning rate of token number in Block_l, I_l is a binary variable indicating whether a selector gets inserted in block l, and HardwareCost is the maximum computation limit of the target hardware, which can be obtained by measuring the real hardware performance. Through Equations (8) and (9), we derive the final hardware-constraint sparsity loss:

$$\pounds_{ratio} = \sum_{L}^{l=1} (1 - \rho_l - \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{N} D_i^{l,b})^2, \tag{10}$$

where B is the training batch size, and $D_i^{l,*}$ means token keep rate in the l-th block. In order to achieve per-image adaptive pruning rates, we set the average token pruning rate of all images in a batch as the convergence target, as shown in Equation (10). Meanwhile, manually adjustable parameters γ are set in Equation (9) to provide the loss slack for the images with the largest pruning rate. Experiments show that the pruning rate difference of images in the same block will not exceed 4.2%.

Layer-to-Phase Progressive Training Schedule. The search space of the optimal model accuracy and hardware efficiency for HFSP training is large. Therefore, we design a hardware-oriented progressive training strategy that leverages the ViT characteristics to efficiently find the optimal accuracy-pruning ratio trade-offs. In a ViT architecture, the encoding efficiency is higher in later blocks, hence, we adopt progressive training on the selector from later blocks to earlier blocks. The process can be divided into three steps:

Table 2: Results of different ViTs on ImageNet-1K. We compare the proposed HFSP with existing ViT pruning methods under comparable FLOPs and the number of parameters. Note that "*" refers to our reproduced results to obtain models with similar FLOPs for comparison. Baseline/160/192/288/384/512/768 indicates the embedding dimensions.

Model	Method	Params (M)	FLOPs (G)	FLOPs ↓ (%)	Top-1 Acc. (%)
DeiT-T	Baseline/192	5.60	1.30	0	72.20
	Baseline/160*	4.00	0.90	30.77	68.10
	S^2ViTE	4.20	0.95	26.92	70.12
	DynamicViT	5.90	0.91	30.00	71.85
	HFSP (Ours)	5.70	0.90	30.77	72.10
DeiT-S	Baseline/384	22.10	4.60	0	79.80
	Baseline/288*	12.60	2.65	42.39	78.53
	IA-RED^2	-	3.15	31.52	79.10
	S^2ViTE	14.60	3.14	31.63	79.22
	DynamicViT	22.80	2.91	36.74	79.30
	DynamicViT*	22.80	2.71	41.09	79.12
	HFSP (Ours)	22.20	2.64	42.61	79.34
DeiT-B	Baseline/768	86.50	17.60	0	81.80
	IA-RED^2	-	11.80	32.96	80.30
	S^2ViTE	56.80	11.77	33.13	82.22
	DynamicViT	89.50	11.02	37.39	80.70
	HFSP (Ours)	86.60	10.49	40.40	81.05
LV-ViT-S	Baseline/384	26.20	6.55	0	83.30
	DynamicViT	26.90	4.57	30.22	83.00
	HFSP (Ours)	26.20	4.28	34.65	83.10
LV-ViT-M	Baseline/512	55.80	12.67	0	84.00
	DynamicViT	57.10	8.45	33.31	83.80
	DynamicViT*	57.10	7.35	41.99	83.61
	HFSP (Ours)	55.90	7.32	42.23	83.71

- **Inserting Token Selection Modules**: Each time we insert a token selector, we train the current selector and finetune the other parts to maximize the pruning ratio without noticeable accuracy drop. We repeat the insertion until there is one selector for each block.
- **Phase Merging**: If the generated pruning rates of the adjacent selection modules is similar, we combine them as one selection phase, and only keep the first selection module of the phase. The reason is that the first selector of one phase has completed most of the token pruning of its phase, the operations of subsequent selectors become less necessary.
- **Accuracy Refinement**: If the final computations are lower than the target upper bound of the computation limits required by hardware resource budget, we reduce the pruning rate of the first phase accordingly to further improve the model accuracy.

Note that restoring the tokens of the first phase can enhance the feature expression in every subsequent block, so as to maximize the accuracy improvement with the same amount of computation.

5 Experiments

5.1 Datasets and Implementation Details

Our experiments are conducted on ImageNet-1K (Deng et al., 2009) with different backbones including DeiT-T, DeiT-S, DeiT-B (Touvron et al., 2021); LV-ViT-S, LV-ViT-M (Jiang et al., 2021); PiT-T, PiT-XS, PiT-S (Heo et al., 2021). The image resolution is 224×224. We follow most of the training settings as in DeiT and train all backbone models for 60 epochs. Our batch size is 256 for DeiT-T, DeiT-S, and LV-ViT-S; and 128 for DeiT-B, LV-ViT-M, PiT-T, PiT-XS, and PiT-S. We set an initial learning rate to be 5e-4 for the soft pruning module and 5e-6 for the backbone. The final model has three token selectors. All models are trained on 8 NVIDIA A100-SXM4-40GB GPUs.

5.2 Experimental Results

Main Results. In Table 2, we compare our method with several representative pruning methods including DynamicViT (Rao et al., 2021), IA-RED^2 (Pan et al., 2021), and S^{2}ViTE (Chen et al., 2021f). We report the top-1 accuracy and FLOPs for each model. Note that "*" refers to the results reproduced with similar FLOPs. Overall, Table 2 show that our HFSP reduces the computational costs by 31%∼43% for various backbones with negligible 0.1%∼0.5% accuracy degradation, which outperforms existing pruning methods on both accuracy and efficiency. On lightweight ViT, DeiT-T, the proposed HFSP still reduces FLOPS by 31% with a negligible accuracy drop (-0.1%). To explore model scaling on ViT, we train more DeiT models with the embedding dimension of 160 and 288 as our baselines. Under comparable computational complexity, our token sparsification method surpasses model scaling method by 4% on DeiT-T($\sim$ 0.9 GFLOPs) and 0.8% on DeiT-S($\sim$ 2.64 GFLOPs). Additionally, our HFSP reaches a throughput of 6354.3 img/s for DeiT-T and 2418.2 img/s for DeiT-S, outpacing DynamicViT by 9%. Note that although S^{2}ViTE can achieve better accuracy on DeiT-B by head pruning, the accuracy of a more edge device compatible model with fewer head has significant degradation and are not comparable with the ones in our method.

Integrated Extension of Pooling-based ViT (PiT Series) and HFSP. Thanks to the pooling-layer mechanism, PiT filters less informative tokens the same as the downsampling process in CNN, which improves the encoding efficiency of data features. Integrated with HFSP, PiT-S can further reduce the amount of computation by 11% without accuracy drop. Figure 3 shows that the attention matrix before and after HFSP retains great similarity, which enlightens that the encoding redundancy of the pooling-layer mechanism can be recognized precisely by HFSP. We also compared HFSP with the embedding dimension scaling on the PiT. When compressing PiT-S to the comparable size of PiT-XS, the accuracy of the produced model is 0.91% acc higher than the original PiT-XS; When the target size is PiT-T, the accuracy of the produced model is 0.9% higher than the original PiT-T. By applying HFSP, more efficient and accurate models can be generated.

Table 3: Detail analysis on Pooling-based ViT with HFSP.

Model	Method	Top1 Acc (%)	FLOPs (G)
PiT-S	Base Model	80.9	2.90
PiT-S	**HFSP (Ours)**	80.9	2.58
PiT-XS	Base Model	78.10	1.40
PiT-S	**HFSP (Ours)**	79.01	1.42
PiT-T	Base Model	73.00	0.71
PiT-XS	**HFSP (Ours)**	74.06	0.72

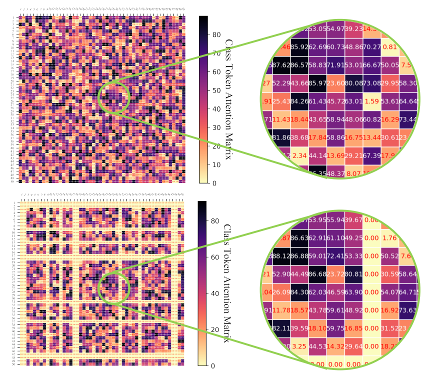

Figure 3: Illustration of the first attention matrix at the final block. Upper figure is the original PiT-S, lower one is with HFSP.

5.3 Deployment on Edge Devices

To evaluate the hardware performance, we implement a framework which runs the ViT model on the edge devices. The evaluation is conducted on a Samsung Galaxy S20 cell phone that has Snapdragon 865 processor, which consists of an Octa-core Kryo 585 CPU carrying high performance with good power efficiency. We use all eight cores on mobile CPUs. We report the average latency over 100 inferences. As shown in Table 4, HFSP achieves 32 ms per inference on mobile CPUs which meets the real-time requirement. As far as we know, this is the first demonstration of ViT inference over 30fps on edge devices.

Additionally, HFSP is evaluated on an embedded FPGA platform, namely, Xilinx ZCU102. To maintain the model accuracy on hardware, 16-bit fixed-point precision is adopted to represent all the model parameters and activation data. The comparison results with baseline models are shown in Table 5. In addition to the total latency, the average latency of the multi-head attention and MLP

Table 4: Evaluation results on Samsung Galaxy S20 with Snapdragon 865 processor.

Model	Method	FLOPs (G)	Latency (ms)
DeiT-T	Baseline	1.30	44
	HFSP (Ours)	0.90	**32**
DeiT-S	Baseline	4.60	122
	HFSP (Ours)	2.64	**84**

Table 5: Evaluation results on Xilinx ZCU102 FPGA board.

Model	Method	FLOPs (G)	Latency (ms)		
			MSA	FFN	Total
DeiT-T	Baseline	1.30	5.04	3.36	8.81
	HFSP (Ours)	0.90	**3.12**	**2.16**	**5.60**
DeiT-S	Baseline	4.60	11.04	10.68	22.31
	HFSP (Ours)	2.64	**6.36**	**6.48**	**13.23**
DeiT-B	Baseline	17.60	33.24	39.36	73.61
	HFSP (Ours)	10.49	**19.44**	**23.04**	**43.24**

modules in each model is listed. Compared with the baseline, DeiT-T, DeiT-S, and DeiT-B could achieve 1.57×, 1.69×, and 1.70× acceleration in the total latency, respectively.

6 Ablation Analysis

Sub-method Effectiveness. To evaluate the effectiveness of each sub-method, we use a single head token selector as our pruning baseline for DeiT-S, then add up each sub-method step by step to compare their performance improvements. The sub-methods include:

- Apply the multi-head mechanism to get multiple token scores for each attention head individually, then use average pooling to get the final token score.
- Add the token package module to draw feature information from the less informative tokens into a package token.
- Add the attention-based branch to derive the head importance score so that the final token score gets obtained by weighted pooling.
- Add the hardware-oriented progressive training schedule, where each token selector is trained individually from the deep to shallow.

From Table 6, we can observe that multi-head token selector has a 0.12% accuracy improvement compared to single head. This demonstrates the importance of evaluating token scores based on the encoding information of each head. After adding the token packaging technique, the accuracy is further improved by 0.13%. This proves our argument that ViT's encoding limitations may cause informative tokens to get pruned, while adding the package means a remedy opportunity for the model. By using the attention-based branch, model is further advanced by 0.06%. This indicates that the selector's capability can improve by replacing the naive average pooling with weighted pooling. Progressive training's accuracy refinement process allows us to further maximize the performance while not exceeding the edge device computation cap.

Table 6: Sub-method effectiveness evaluation on DeiT-S.

Method	Params (M)	FLOPs (G)	Top-1 Acc. (%)	Throughput (img/s)
DeiT-S	22.10	4.60	79.80	1510.7
Pruning Baseline	22.20	2.63	79.03	$>$ **2424.1**
+ Multihead	22.20	2.63	79.15	$>$ **2424.1**
+ Package Token	22.20	2.64	79.28	$>$ 2419.8
+ Attention-based Branch	22.20	2.64	79.34	$>$ 2418.2
+ Progressive Training	22.20	2.65	**79.36**	$>$ 2415.3

7 Conclusion

In this paper, we propose a dynamic hardware-friendly soft pruning framework called HFSP for various ViT models. Our attention-based multi-head token selector and token packaging technique, along with the hardware-oriented progressive training can well balance the tradeoff between accuracy and specific hard-ware constraints without introducing operators that are not supported by the hardware. For instance, our method reduces the FLOPs of DeiT-S by over 42.6% while only sacrificing 0.46% top-1 accuracy. We further deploy our model on mobile device and FPGA, which both meets the real-time requirement.

REFERENCES

Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. T6d-direct: Transformers for multi-object 6d pose direct regression. *arXiv preprint arXiv:2109.10948*, 2021.

Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229. Springer, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 782–791, 2021.

Boyu Chen, Peixia Li, Baopu Li, Chuming Li, Lei Bai, Chen Lin, Ming Sun, Junjie Yan, and Wanli Ouyang. Psvit: Better vision transformer via token pooling and attention sharing. *arXiv preprint arXiv:2108.03428*, 2021a.

Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *arXiv preprint arXiv:2103.14899*, 2021b.

Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12299–12310, 2021c.

Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. *arXiv preprint arXiv:2107.00651*, 2021d.

Pengguang Chen, Yixin Chen, Shu Liu, Mingchang Yang, and Jiaya Jia. Exploring and improving mobile level vision transformers. *arXiv preprint arXiv:2108.13015*, 2021e.

Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *arXiv preprint arXiv:2106.04533*, 2021f.

Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021g.

Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pretraining or strong data augmentations. *arXiv preprint arXiv:2106.01548*, 2021h.

Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8126–8135, 2021i.

Bowen Cheng, Alexander G Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *arXiv preprint arXiv:2107.06278*, 2021.

Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1601–1610, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. Transvg: End-to-end visual grounding with transformers. *arXiv preprint arXiv:2104.08541*, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Alaaeldin El-Nouby, Natalia Neverova, Ivan Laptev, and Hervé Jégou. Training vision transformers for image retrieval. *arXiv preprint arXiv:2102.05644*, 2021a.

Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *arXiv preprint arXiv:2106.09681*, 2021b.

Peng Gao, Jiasen Lu, Hongsheng Li, Roozbeh Mottaghi, and Aniruddha Kembhavi. Container: Context aggregation network. *arXiv preprint arXiv:2106.01401*, 2021.

Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. *arXiv preprint arXiv:2104.01136*, 2021.

Cong Guo, Bo Yang Hsueh, Jingwen Leng, Yuxian Qiu, Yue Guan, Zehuan Wang, Xiaoying Jia, Xipeng Li, Minyi Guo, and Yuhao Zhu. Accelerating sparse dnn models without hardware-support via tile-wise sparsity. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15. IEEE, 2020.

Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R. Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199, Apr 2021. ISSN 2096-0662. doi: 10.1007/s41095-021-0229-5. URL http://dx.doi.org/10.1007/s41095-021-0229-5.

Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *arXiv preprint arXiv:2103.00112*, 2021.

Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pp. 1135–1143, Cambridge, MA, USA, 2015. MIT Press.

Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv preprint arXiv:2103.16302*, 2021.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

Drew A Hudson and C Lawrence Zitnick. Generative adversarial transformers. *arXiv preprint arXiv:2103.01209*, 2021.

Zihang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Yujun Shi, Xiaojie Jin, Anran Wang, and Jiashi Feng. All tokens matter: Token labeling for training better vision transformers. *arXiv preprint arXiv:2104.10858*, 2021.

Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, and Hyunwoo J Kim. Hotr: End-to-end human-object interaction detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 74–83, 2021.

Bingbing Li, Zhenglun Kong, Tianyun Zhang, Ji Li, Zhengang Li, Hang Liu, and Caiwen Ding. Efficient transformer-based large scale language representations using hardware-friendly block structured pruning. *arXiv preprint arXiv:2009.08065*, 2020a.

Zhaoshuo Li, Xingtong Liu, Nathan Drenkow, Andy Ding, Francis X Creighton, Russell H Taylor, and Mathias Unberath. Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2011.02910*, 2020b.

Yahui Liu, Enver Sangineto, Wei Bi, Nicu Sebe, Bruno Lepri, and Marco De Nadai. Efficient training of visual transformers with small-size datasets. *arXiv preprint arXiv:2106.03746*, 2021a.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021b.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pp. 2736–2744, 2017.

Zhisheng Lu, Hong Liu, Juncheng Li, and Linlin Zhang. Efficient transformer for single image super-resolution. *arXiv preprint arXiv:2108.11084*, 2021.

Mingyuan Mao, Renrui Zhang, Honghui Zheng, Peng Gao, Teli Ma, Yan Peng, Errui Ding, and Shumin Han. Dual-stream network for visual recognition. *arXiv preprint arXiv:2105.14734*, 2021.

Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021.

Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. *arXiv preprint arXiv:2109.08141*, 2021.

Bowen Pan, Yifan Jiang, Rameswar Panda, Zhangyang Wang, Rogerio Feris, and Aude Oliva. Ia-red^2: Interpretability-aware redundancy reduction for vision transformers, 2021.

Sebastian Prillo and Julian Eisenschlos. Softsort: A continuous relaxation for the argsort operator. In *International Conference on Machine Learning*, pp. 7793–7802. PMLR, 2020.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *arXiv preprint arXiv:2108.08810*, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *arXiv preprint arXiv:2106.02034*, 2021.

Ao Ren, Tianyun Zhang, Shaokai Ye, Jiayu Li, Wenyao Xu, Xuehai Qian, Xue Lin, and Yanzhi Wang. Admm-nn: An algorithm-hardware co-design framework of dnns using alternating direction method of multipliers, 2018.

Michael S. Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: What can 8 learned tokens do for images and videos?, 2021.

Victor Sanh, Thomas Wolf, and Alexander M. Rush. Movement pruning: Adaptive sparsity by fine-tuning, 2020.

Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16519–16529, 2021.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers, 2021.

Yehui Tang, Kai Han, Yunhe Wang, Chang Xu, Jianyuan Guo, Chao Xu, and Dacheng Tao. Patch slimming for efficient vision transformers, 2021.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herv'e J'egou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Hanrui Wang, Zhekai Zhang, and Song Han. Spatten: Efficient sparse attention architecture with cascade token and head pruning, 2021a.

Pichao Wang, Xue Wang, Fan Wang, Ming Lin, Shuning Chang, Wen Xie, Hao Li, and Rong Jin. Kvt: k-nn attention for boosting vision transformers, 2021b.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, 2021c.

Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020.

Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer, 2021.

Chenfeng Xu, Bohan Zhai, Bichen Wu, Tian Li, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. You only group once: Efficient point-cloud processing with token representation and relation inference module, 2021a.

Yifan Xu, Zhijie Zhang, Mengdan Zhang, Kekai Sheng, Ke Li, Weiming Dong, Liqing Zhang, Changsheng Xu, and Xing Sun. Evo-vit: Slow-fast token evolution for dynamic vision transformer, 2021b.

Fanglei Xue, Qiangchang Wang, and Guodong Guo. Transfer: Learning relation-aware facial expression representations with transformers, 2021.

Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking, 2021.

Ceyuan Yang, Zhirong Wu, Bolei Zhou, and Stephen Lin. Instance localization for self-supervised detection pretraining, 2021a.

Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5791–5800, 2020.

Guanglei Yang, Hao Tang, Mingli Ding, Nicu Sebe, and Elisa Ricci. Transformer-based attention networks for continuous pixel-wise prediction. In *ICCV*, 2021b.

Qihang Yu, Yingda Xia, Yutong Bai, Yongyi Lu, Alan Yuille, and Wei Shen. Glance-and-gaze vision transformer. *arXiv preprint arXiv:2106.02277*, 2021.

Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers, 2021a.

Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021b.

Xiaoyu Yue, Shuyang Sun, Zhanghui Kuang, Meng Wei, Philip Torr, Wayne Zhang, and Dahua Lin. Vision transformer with progressive sampling. *arXiv preprint arXiv:2108.01684*, 2021.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. *arXiv preprint arXiv:2106.04560*, 2021.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer, 2021.

Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6881–6890, 2021.

Mingjian Zhu, Kai Han, Yehui Tang, and Yunhe Wang. Visual transformer pruning. *arXiv preprint arXiv:2104.08500*, 2021.