Vision-MoR: Scaling Vision Transformer via Patch-Level Mixture-of-Recursions

Yunhong He[1,†,*], Zhengqing Yuan[2,†]
Weixiang Sun[2], Yiyang Li[2], Yixin Liu[3], Yanfang Ye[2], Lichao Sun[3]

[1]Independent Researcher
[2]University of Notre Dame
[3]Lehigh University

Abstract

Scaling Vision Transformers (ViTs) has yielded remarkable advancements in diverse vision tasks, albeit at the cost of escalating computational, memory, and parameter demands. Existing efficiency techniques typically address only one dimension, computation, memory, or parameters, lacking a cohesive approach. In this paper, we introduce **Vision-MoR**, a novel ViT architecture that unifies parameter sharing, spatially adaptive computation, and memory-efficient design into a single framework. Vision-MoR employs a spatial-aware router with shifted-window attention to dynamically assign per-patch recursion depths, coupled with a recursive Transformer loop enabling token-wise early exiting. This facilitates content-adaptive processing and recursive parameter reuse while preserving spatial locality. On ImageNet-1K, **Vision-MoR Small** attains 74.6% Top-1 accuracy with 140M FLOPs and 5.7M parameters, outperforming EfficientViT-M2 (70.8%) and SHViT-S1 (72.8%) at superior throughput. The **Vision-MoR X-Large** variant achieves 80.4% Top-1 and 95.2% Top-5 accuracy using 14.3M parameters and 2044M FLOPs, surpassing ResNet-50 and EfficientNet-B1. On COCO object detection, Vision-MoR X-Large yields 39.1 AP with the lowest latency among comparable models. These results underscore Vision-MoR's state-of-the-art accuracy-efficiency trade-offs, positioning it as a scalable, deployment-friendly backbone for real-time vision applications.

Introduction

Since Vision Transformer (ViT) models (Rao et al. 2021; Mehta and Rastegari 2022; Graham et al. 2021; Wei et al. 2025) first demonstrated that pure self-attention architectures could rival—and often surpass—convolutional networks on large-scale image recognition, ever-larger ViT variants have driven steady gains across classification, detection, segmentation, and even video understanding (Thisanke et al. 2023; Wang et al. 2021; Radford et al. 2021; Tschannen et al. 2025; He et al. 2025). These improvements, however, have come at a steep computational cost (Alabdulmohsin et al. 2023): scaling depth, width, and input resolution inflates quadratic attention complexity, ballooning both memory footprint and FLOPs and rendering training or real-time inference feasible only on specialized clusters. A rich line of work therefore seeks "efficient ViTs" (Yin et al. 2022; Liu et al. 2023; Wang et al. 2024), exploring patch pruning and token dropping (Bolya et al. 2022), low-rank or sparse attention (Liu, Wu, and Guo 2022), layer re-parameterization (Li et al. 2023b), and partial weight sharing (Zhang et al. 2022). Yet each technique typically targets one axis of efficiency in isolation. As a result, practitioners face a fragmented design space—pruning sacrifices accuracy under heavy compression, sparse attention lowers compute but not parameters, and tied-layer schemes decrease parameters but leave per-token compute unchanged. The absence of a unified strategy that can jointly trim parameters, adapt computation to spatial content, and reduce memory traffic limits the practicality of ViTs in resource-constrained settings.

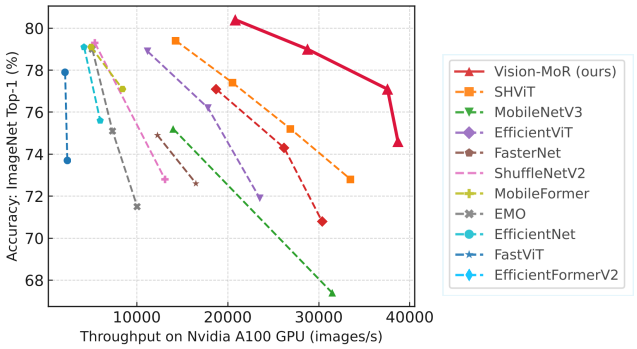

Figure 1: Comparison of throughput and accuracy between our Vision-MoR and other recent methods.

Despite steady progress, today's ViT-efficiency methods remain siloed (Xu et al. 2024; Yuan et al. 2024). Token/-patch pruning and early exiting lower compute but retain full model weights; weight-tying or low-rank factorization shrink parameters but still expend a uniform number of FLOPs on every patch; memory-centric tricks such as reversible layers or i-KV caching reduce activation costs but leave the core attention pattern unchanged (Yun and Ro 2024; Papa et al. 2024a). Consequently, no existing vision architecture simultaneously (i) reuses parameters, (ii) allocates computation adaptively per spatial region, and (iii)

*Yunhong He is an independent undergraduate student, remotely working with Lichao Sun.
†Equal Contribution

trims memory traffic end-to-end (Yuan et al. 2024). In the large-language-model (LLM) community, however, the newly proposed Mixture-of-Recursions (MoR) (Bae et al. 2025) has shown that these three axes can be unified: MoR trains lightweight routers that decide, token-by-token, how many times a shared block is reapplied, achieving parameter sharing, token-level adaptive depth, and recursion-wise KV caching in a single stack—all while improving perplexity–compute trade-offs. This holistic recipe suggests a promising blueprint for vision.

Directly extending the 1D language MoR framework from language to 2D introduces several unique challenges. 1) First, maintaining spatial coherence and locality becomes nontrivial. Vision tokens lie on a 2D grid, and re-entering a shared computation block with varying subsets of patches may disrupt local feature continuity, thereby impairing the model's ability to capture fine-grained structures. 2) Second, vision patches exhibit highly heterogeneous information density—unlike language tokens, which are relatively uniform. This demands a sophisticated routing mechanism capable of assigning per-patch recursion depths without incurring excessive routing overhead or disrupting the attention topology.

To address these challenges and extend MoR's unified efficiency paradigm to the visual domain, we redesign the recursion mechanism around image patches. 1) Patch-Level Recursive Computation: We partition the image into spatially contiguous cells and use a lightweight router to assign individualized recursion depths. Background regions can exit early, while semantically rich foreground areas continue through shared blocks for deeper refinement. 2) Spatial-Aware Routing: The router operates on multi-scale pooled features, enforcing spatial consistency by ensuring neighboring patches differ by no more than one recursion step. This preserves locality and coherent receptive fields across layers.

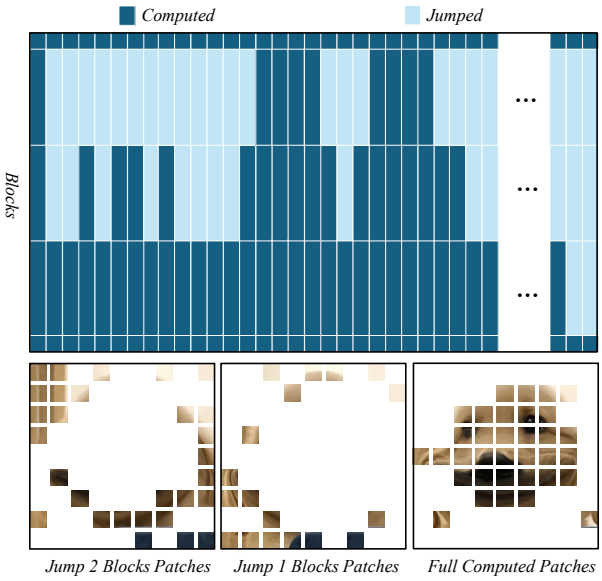

Figure 2: Visualization example of patch-wise adaptive computation in Vision-MoR Base.

Related Work

Single-Axis Efficiency Optimization

Recent advances in efficient ViTs have predominantly focused on optimizing a single resource dimension—compute, parameters, or memory—often in isolation (Zhang et al. 2025; Papa et al. 2024b). To reduce computational cost, token-level methods such as patch pruning and early exits (e.g., DynamicViT (Rao et al. 2021)) discard less informative tokens, while sparse attention techniques like Linformer (Wang et al. 2020) linearize the self-attention mechanism. Parameter efficiency is addressed through lightweight architectures and hybrid CNN-ViT designs, such as LeViT (Graham et al. 2021), which combine convolutional backbones with compact attention modules.

Hybrid Multi-Axis Efficiency Strategies and the Vision-MoR Paradigm

Building on the above single-axis survey, more recent work has begun to explore hybrid strategies that span multiple efficiency dimensions, yet still fall short of a fully unified solution. Recursive ViTs such as SReT(Shen, Liu, and Xing 2021) shares a single block across layers to slash parameter overhead but apply the same recursion depth to every patch; Dynamic-depth models like SkipViT (Ataiefard et al. 2024) and AdaViT (Meng et al. 2022) adaptively allocate computation at the patch level, while retaining the full parameter set. However, they do not prune or modify the underlying attention structure, and maintain uniform model capacity during inference. Other refinements (Tay et al. 2021; Wu et al. 2022; Mehta and Rastegari 2021) emphasize memory reduction via single-head attention and compact token embedding, without altering model size or compute throughput.

Methodology

The proposed Vision Mixture-of-Recursions (Vision-MoR) framework, illustrated in Figure 3, enables fine-grained, patch-wise dynamic depth allocation for visual inputs. It consists of two key components: a *Spatial Router* for token-specific routing decisions and a *Recursion Loop* that adaptively allocates compute by controlling recursion depth. This section details their design and interaction, along with the overall training strategy.

Spatial Router

To enable dynamic computation across different spatial regions of an image, we begin by tokenizing the input image $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$ into non-overlapping $P \times P$ patches. This yields $N = \left(\frac{H}{P}\right)\left(\frac{W}{P}\right)$ tokens $\{\mathbf{x}_i\}_{i=1}^{N}$, each embedded into $\mathbb{R}^d$. These tokens are then fed into a spatial router, which predicts a suitable recursion depth for each patch based on its content.

Unlike traditional MLP-based routers, our spatial router captures local context via a two-stage *shifted window self-attention* mechanism, inspired by the Swin Transformer (Liu et al. 2021). Specifically, the input patch grid $\mathbf{X} \in$

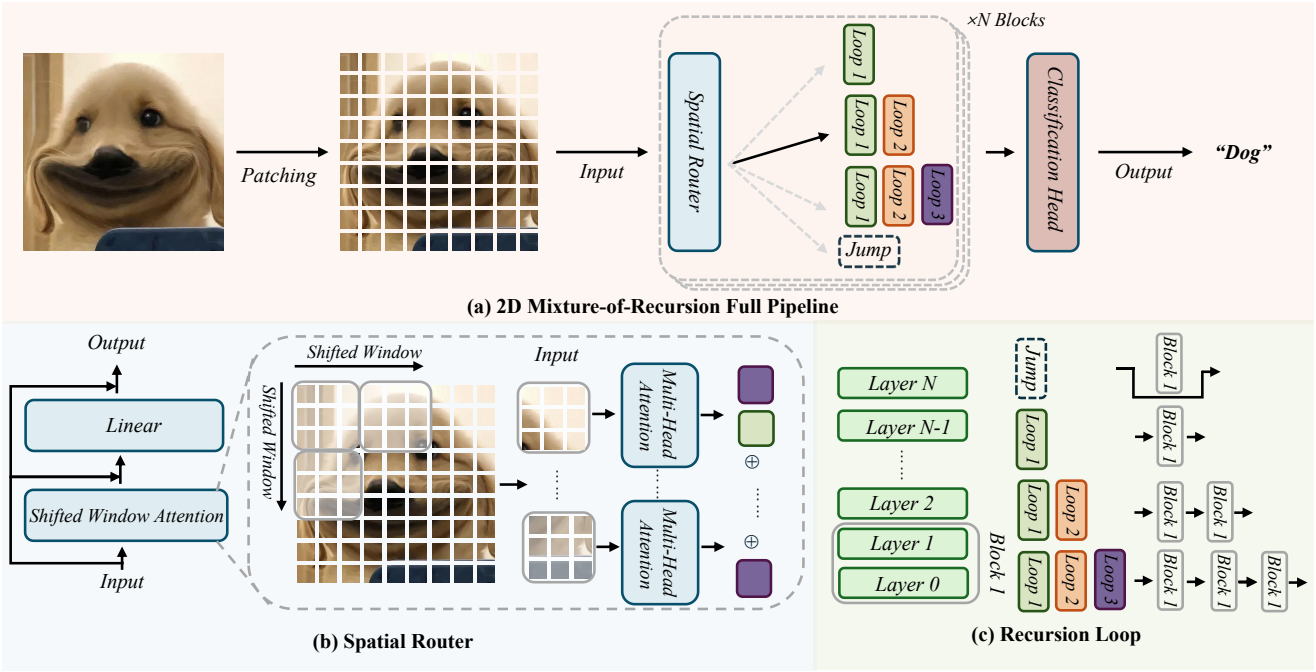

Figure 3: Overview of the Vision-MoR framework. (a) The full pipeline first tokenizes the image into patches and dynamically routes them to different recursion depths using a spatial router before classification. (b) The spatial router applies shifted window attention and Linear to determine expert paths for each patch token. (c) Based on routing decisions, tokens traverse different numbers of recursive computation blocks, enabling adaptive and efficient processing.

$\mathbb{R}^{B\times H'\times W'\times d}$ is divided into $M \times M$ windows. Within each window, standard attention is applied:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \tag{1}$$

$$\mathbf{K} = \mathbf{X}\mathbf{W}_K, \tag{2}$$

$$\mathbf{V} = \mathbf{X}\mathbf{W}_V, \tag{3}$$

$$\mathbf{A} = \mathrm{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}} + \mathbf{B}_{\text{rel}}\right). \tag{4}$$

where $\mathbf{B}_{\text{rel}}$ is a trainable relative positional bias. A second round of attention is then applied after shifting the window by $\lfloor M/2 \rfloor$ in both axes, enhancing cross-window interactions.

The attention-refined token embeddings $\tilde{\mathbf{x}}_i$ are passed through a layer norm and lightweight MLP to produce routing logits:

$$\mathbf{h}_i = \mathrm{LN}(\tilde{\mathbf{x}}_i), \quad \mathbf{z}i = \mathrm{MLP}(\mathbf{h}i) \in \mathbb{R}^{D\max}, \tag{5}$$

where $D\max$ denotes the maximum recursion depth. These logits determine the individual token's path through recursive transformer blocks, allowing either hard routing via $\arg\max$ or soft gating through a sigmoid/softmax transformation.

Recursive Computation with Dynamic Token Allocation

The recursion loop governs how each token is processed through shared transformer blocks, enabling both parameter reuse and adaptive depth control. Instead of allocating L distinct transformer layers, we group them into N_r parameter sets reused cyclically:

$$\text{Block}j = \Phi' j \bmod N_r : j = 0, \ldots, L-1, \tag{6}$$

where Φ'_k is the parameter set for the k-th recursion block.

At each recursion step r, only a subset of tokens $\mathcal{A}^{(r)}$ continues. For token t, the router predicts a gating value:

$$g_t^{(r)} = \mathcal{G}(\mathbf{w}_r^\top h_t^{(r)}), \tag{7}$$

where $h_t^{(r)}$ is the token's hidden state and $\mathcal{G}$ is typically a sigmoid. Based on the top-k gating values, selected tokens proceed to the next block:

$$\mathcal{A}^{(r+1)} = \text{Top-}k\left(g_t^{(r)} : t \in \mathcal{A}^{(r)}\right). \tag{8}$$

Others exit the computation loop and are passed directly to the output head.

The token state is recursively updated based on a gated residual connection:

$$h_t^{(r+1)} = f(h_t^{(r)}; \Phi') \cdot g_t^{(r)} + h_t^{(r)} \cdot (1 - g_t^{(r)}), \tag{9}$$

blending new features with previous representations to ensure smooth depth transitions. To preserve expressivity, the first and last layers are kept unique, while intermediate blocks are shared following a "middle-cycle" strategy (Bae et al. 2025):

$$\text{Layers} = [\Phi_0, \underbrace{\Phi', \ldots, \Phi'}\, N_r, \Phi L - 1]. \tag{10}$$

Here, the intermediate blocks cycle through (N_r) distinct parameter sets (Φ'_k) (for (k = 0, . . . , N_r-1)), repeated as necessary to fill the middle layers.

Full Pipeline and Integration

The full inference pipeline is depicted in Figure 3(a). After patch embedding, the spatial router (Figure 3(b)) assigns a recursion depth to each token based on its content and spatial context using Equations (3) and (5). These decisions govern how tokens traverse the recursion loop (Figure 3(c)), where they are dynamically processed according to Equations(7)–(9).

Upon completion of their recursion paths, all tokens are aggregated (via [CLS] token or pooling) and passed through a classification head. The overall design allows for a flexible allocation of compute: complex or semantically rich regions receive more processing, while redundant or background areas exit early, optimizing both accuracy and efficiency.

Training Objective

To jointly optimize the routing mechanism and final prediction, we adopt a multi-term training loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \lambda_{\text{aux}}\mathcal{L}_{\text{route}} + \lambda_z\mathcal{L}_z, \tag{11}$$

where $\mathcal{L}_{\text{CE}}$ is the standard cross-entropy loss for the final model predictions, λ_{aux} and λ_z are hyperparameters that weight the auxiliary losses **(typically set to small positive values like 0.1 to balance training)**, $\mathcal{L}_{\text{route}}$ is a binary cross-entropy loss between routing logits z_i **(the model's predicted scores for routing decisions on the i-th token)** and top-k supervision:

$$\mathcal{L}_{\text{route}} = \frac{1}{N}\sum_{i=1}^{N}\text{BCEWithLogits}(z_i, y_i^{\text{route}}), \tag{12}$$

with N being the total number of tokens and y_i^{route} denoting the ground-truth routing labels for the i-th token (derived from top-k selection based on content difficulty), and $\mathcal{L}_z$ is a Z-Loss regularizer to prevent collapse of routing distributions:

$$\mathcal{L}_z = \mathbb{E}_i\left[\left(\log\sum_j e^{z_{ij}}\right)^2\right]. \tag{13}$$

Here, $\mathbb{E}_i$ denotes the expectation (average) over all tokens i, and z_{ij} represents the logits for the i-th token across routing options j (e.g., different recursion depths). This regularizer penalizes overly confident or collapsed distributions by squaring the log-sum-exp of the logits.

To encourage gradual adoption of dynamic routing, we employ a warm-up strategy that linearly increases the routing capacity c_t **(the allowed ratio of tokens forwarded to deeper recursions at training step t)** as training progresses:

$$c_t = c_{\text{target}} + (1 - c_{\text{target}}) \cdot 0.5 \left[1 + \cos\Big(\pi \cdot \min\big(1, \tfrac{t}{T_{\text{warmup}}}\big)\Big)\right], \tag{14}$$

where c_{target} is the target capacity (e.g., a value like 0.5 for half the tokens) and T_{warmup} is the duration of the schedule (e.g., a number of training epochs or steps).

Experiment

Experiment Setting

We training our proposed Vision-MoR framework training on the ImageNet-1K dataset (Deng et al. 2009), which consists of 1.28 million training images and 50,000 validation images across 1,000 classes. All models are trained from scratch using the AdamW optimizer (Loshchilov and Hutter 2018) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of 0.01. The base learning rate is set to 5×10^{-5}, and a cosine learning rate schedule with a minimum learning rate ratio of 0.1 is applied, along with linear warmup for the first 1,000 steps. Training is conducted for 300 epochs with a global batch size of 2048 using gradient accumulation. All training experiments are conducted on 4*Nvidia GH200 GPUs using the PyTorch framework with HuggingFace Transformers. We enable `bfloat16` (bf16) mixed-precision training and gradient checkpointing for memory efficiency.

Model Configurations We construct four Vision-MoR variants by varying the number of layers, embedding dimension, attention heads, and feed-forward width to accommodate different model capacities. The detailed architectural configurations for Vision-MoR Small, Base, Large, and X-Large are summarized in Table 2.

Vision-MoR on ImageNet-1K Classification

We evaluate our proposed Vision-MoR models on the ImageNet-1K classification benchmark and compare them against a wide spectrum of state-of-the-art lightweight vision architectures. As shown in Table 1, Vision-MoR consistently achieves superior performance across all scales under comparable or lower computational budgets. In particular, **Vision-MoR Small** attains a Top-1 accuracy of 74.6% with only 140M FLOPs and 5.7M parameters—outperforming all models below 300M FLOPs, such as EfficientViT-M2 (70.8%) and SHViT-S1 (72.8%), while maintaining the highest GPU throughput (38.7k images/s). This demonstrates the effectiveness of our patch-wise dynamic depth allocation in capturing complex visual patterns with minimal compute.

As we scale up the model, **Vision-MoR Base** and **Large** variants continue to lead in both accuracy and efficiency. Vision-MoR Base achieves 77.1% Top-1 accuracy at 369M FLOPs, surpassing EfficientViT-M5 (77.1%, 522M FLOPs) and PoolFormer-S12 (77.2%, 1823M FLOPs) with significantly lower computational cost. Notably, our Vision-MoR Large reaches 79.0% accuracy with 892M FLOPs, outperforming FasterNet-T2 (78.9%, 1910M FLOPs), EMO-6M (79.0%, 961M FLOPs), and FastViT-T12 (79.1%, 1400M FLOPs), while also achieving markedly higher inference throughput on both GPU (38.7k images/s) and ONNX CPU (1382 images/s). These results validate that our recursive and router-guided computation strategy scales efficiently with model size.

At the upper bound, **Vision-MoR X-Large** achieves a new state-of-the-art accuracy of 80.4% Top-1 and 95.2% Top-5 with only 2044M FLOPs and 14.3M parameters. It outperforms all competing models in this regime, including ResNet-50 (78.8%, 4110M FLOPs), EfficientNet-

Model	Reso.	Epochs	FLOPs (M)	Params (M)	Throughput (images/s)			Top-1 (%)	Top-5 (%)
					GPU	CPU	CPU_{ONNX}		
MobileNetV3-Small (Koonce 2021)	224	600	57	2.5	31477	**167**	1172	67.4	-
MobileViT-XXS (Mehta and Rastegari 2021)	256	300	410	1.3	7594	21	170	69.0	-
MobileViTV2 ×0.5 (Mehta and Rastegari 2022)	256	300	**466**	1.4	8616	17	157	70.2	-
EfficientViT-M2 (Pan et al. 2022)	224	300	201	4.2	30377	147	781	70.8	90.2
MobileOne-S0 (Vasu et al. 2023b)	224	300	275	2.1	19689	86	**1648**	71.4	-
EMO-1M (Xie et al. 2024)	224	300	261	1.3	10032	34	119	71.5	-
FasterNet-T0 (Chen et al. 2023)	224	300	340	3.9	23518	92	844	71.9	-
ShuffleNetV2 ×1.5 (Ma et al. 2018)	224	300	299	3.5	16495	62	799	72.6	-
MobileFormer-96M (Chen et al. 2022)	224	450	96	3.6	13106	91	235	72.8	-
SHViT-S1 (Yun and Ro 2024)	224	300	241	6.3	33489	143	1111	72.8	91.0
Vision-MoR Small	224	300	140	5.7	**38700**	165	1291	**74.6**	**92.2**
EfficientFormerV2-S0 (Li et al. 2023a)	224	300	400	3.5	2374	54	372	73.7	-
EfficientViT-M4 (Liu et al. 2023)	224	300	299	8.8	26201	113	616	74.3	91.8
EdgeViT-XXS (Pan et al. 2022)	224	300	600	4.1	6763	33	168	74.4	-
MobileViT-XS (Mehta and Rastegari 2021)	256	300	**986**	2.3	4408	8	96	74.8	-
ShuffleNetV2 ×2.0 (Ma et al. 2018)	224	300	591	7.4	12276	40	250	74.9	92.4
EMO-2M (Xie et al. 2024)	224	300	439	2.3	7333	25	78	75.1	-
MobileNetV3-Large (Koonce 2021)	224	600	217	5.4	13994	43	613	75.2	-
SHViT-S2 (Yun and Ro 2024)	224	300	366	11.4	26878	99	951	75.2	92.4
Vision-MoR Base	224	300	369	9.97	**37551**	**138**	**1326**	**77.1**	**92.8**
FastViT-T8 (Vasu et al. 2023a)	256	300	700	2.1	5978	23	140	75.6	-
GhostNet ×1.3 (Paoletti et al. 2021)	224	300	226	7.3	9433	39	109	75.7	92.7
FasterNet-T1 (Chen et al. 2023)	224	300	850	7.6	17827	41	552	76.2	-
EfficientNet-B0 (Tan and Le 2019)	224	350	390	5.3	8433	26	267	77.1	93.3
EfficientViT-M5 (Pan et al. 2022)	224	300	522	12.4	18722	64	456	77.1	93.4
PoolFormer-S12 (Yu et al. 2022)	224	300	**1823**	11.9	5432	13	120	77.2	-
MobileOne-S2 (Vasu et al. 2023b)	224	300	1299	7.8	9355	22	581	77.4	-
SHViT-S3 (Yun and Ro 2024)	224	300	601	14.2	20522	62	731	77.4	93.4
Vision-MoR Large	224	300	892	11.4	**28760**	**117**	**1382**	**79.0**	**94.3**
EdgeViT-XS (Pan et al. 2022)	224	300	1100	6.7	5520	21	120	77.5	-
EfficientFormerV2-S1 (Li et al. 2023a)	224	300	650	6.1	2112	37	325	77.9	-
MobileViTV2 ×1.0 (Mehta and Rastegari 2022)	256	300	1800	4.9	4345	7	63	78.1	-
ResNet50 (He et al. 2016)	224	300	4110	25.6	5281	8	271	78.8	-
FasterNet-T2 (Chen et al. 2023)	224	300	1910	15.0	11181	21	417	78.9	-
EMO-6M (Zhang et al. 2023)	224	300	961	6.1	5105	15	50	79.0	-
EfficientNet-B1 (Tan and Le 2019)	240	350	700	7.8	4982	11	156	79.1	94.4
FastViT-T12 (Vasu et al. 2023a)	256	300	1400	6.8	4197	14	92	79.1	-
MobileFormer-508M (Chen et al. 2022)	224	450	508	14.0	5390	23	91	79.3	-
MobileOne-S4 (Vasu et al. 2023b)	224	300	2978	14.8	5281	11	281	79.4	-
SHViT-S4 (Yun and Ro 2024)	256	300	986	16.5	14283	36	509	79.4	94.5
Vision-MoR X-Large	224	300	2044	14.3	**20821**	**53**	**747**	**80.4**	**95.2**

Table 1: Comprehensive comparison of representative lightweight vision models on ImageNet-1K on Nvidia A100 GPU for Inference. For each model, we report input resolution, training epochs, FLOPs, parameter count, throughput on GPU/CPU/ONNX, and classification accuracy (Top-1 and Top-5). The table covers a diverse set of architectures, including MobileNet, ShuffleNet, MobileViT, EfficientViT, FasterNet, EdgeViT, GhostNet, and others. Our proposed Vision-MoR variants are highlighted in gray, consistently achieving leading accuracy and throughput among models with similar or lower computational cost.

B1 (79.1%), and MobileOne-S4 (79.4%), while requiring substantially less computation. The combination of high accuracy, lightweight design, and exceptional hardware efficiency—particularly on CPU and ONNX runtimes—highlights the universal deployability of Vision-MoR across diverse platforms. These results affirm the strength of Vision-MoR as a unified, scalable, and efficient vision backbone.

Different Device Performance

We benchmark the inference efficiency of our proposed Vision-MoR against SHViT—a state-of-the-art memory-efficient Vision Transformer—across various model scales on the Nvidia GH200 GPU. As summarized in Table 3, we report peak throughput (images per second) and inference latency under batch sizes of 1, 4, and 8—key metrics for real-time applications. Across all configurations, Vision-MoR consistently outperforms SHViT in both throughput and latency, demonstrating the effectiveness of its recursive computation and dynamic token routing. At the smallest scale (5.71M parameters), Vision-MoR achieves 8600.15 images/s, more than twice that of SHViT (6.33M, 4113.34 images/s). In terms of latency, Vision-MoR reaches 3.63 ms at batch size 1, compared to 7.03 ms for SHViT—a 48.4% reduction. This low-latency advantage persists across larger batches, where Vision-MoR remains nearly constant (3.72 ms @ batch 4, 3.71 ms @ batch 8), indicating efficient parallelism and minimal overhead. At higher model capacities (e.g., $\tilde{2}$5M), Vision-MoR sustains 835.76 images/s and 9.57 ms latency, while SHViT falls to 448.91 images/s and 17.82 ms, reflecting a steeper decline in performance. Notably, SHViT's latency nearly doubles at this scale, whereas

Model	Layers	Embed Dim	Hidden Size	Heads	FFN Dim
Vision-MoR Small	12	384	384	6	1536
Vision-MoR Base	12	512	512	8	2048
Vision-MoR Large	14	576	576	9	2304
Vision-MoR X-Large	16	640	640	10	2560

Table 2: Model configurations for Vision-MoR. We report the number of layers, embedding dimension, hidden size, number of attention heads, and feed-forward network (FFN) dimension for each variant.

Models	Params (M)	Max Throughput (16 imgs/s)	Batch 1 Time (ms)	Batch 4 Time (ms)	Batch 8 Time (ms)
Vision-MoR	5.71	8600.15	3.63 ± 0.06	3.72 ± 0.08	3.71 ± 0.05
	9.97	6238.36	4.99 ± 0.10	5.19 ± 0.11	5.16 ± 0.08
	15.42	3286.04	4.82 ± 0.06	4.87 ± 0.06	4.90 ± 0.07
	17.88	3234.68	4.91 ± 0.11	4.89 ± 0.09	4.89 ± 0.07
	22.04	1738.72	9.02 ± 0.24	9.21 ± 0.14	9.28 ± 0.14
	25.58	835.76	9.37 ± 0.18	9.47 ± 0.15	9.57 ± 0.13
SHViT	6.33	4113.34	7.03 ± 0.11	7.50 ± 0.09	7.67 ± 0.10
	11.48	4553.22	6.39 ± 0.08	6.48 ± 0.09	6.64 ± 0.11
	14.25	2039.91	7.08 ± 0.11	7.45 ± 0.12	7.77 ± 0.10
	18.70	1950.13	7.91 ± 0.11	8.66 ± 0.12	9.12 ± 0.12
	20.46	951.90	16.49 ± 0.20	16.53 ± 0.18	17.25 ± 0.14
	25.31	448.91	17.84 ± 0.16	17.90 ± 0.19	17.82 ± 0.18

Table 3: Performance Comparison of Vision-MoR and SHViT on GH200 GPU

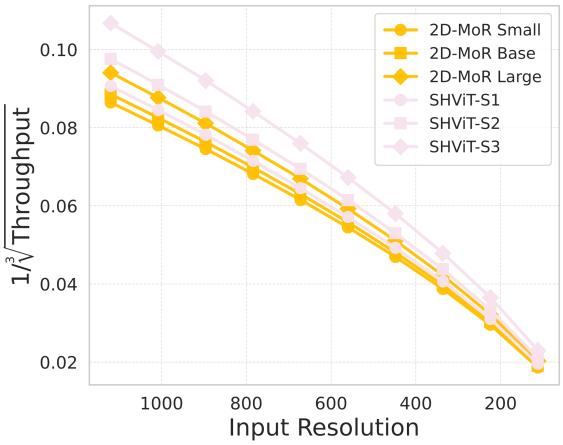

Figure 4: Inverse throughput ($1/\lambda$) versus input resolution for Vision-MoR and SHViT models. Lower values indicate higher efficiency. Across all resolutions, Vision-MoR variants consistently exhibit superior scalability and throughput compared to SHViT counterparts, particularly in the high-resolution regime. This demonstrates the robustness of Vision-MoR's spatially adaptive routing in maintaining computational efficiency as image resolution increases.

Vision-MoR remains stable, highlighting its robust scalability in both compute- and memory-bound regimes. These results affirm that Vision-MoR delivers superior hardware efficiency and practical deployment potential across a range of real-world settings.

Vision-MoR on downstream tasks

To assess the generalization capability of Vision-MoR beyond image classification, we evaluate its performance on the COCO 2017 object detection benchmark using RetinaNet as the detection head. All backbones are pretrained on ImageNet-1K following the settings in Table 4, and subsequently fine-tuned on COCO. As reported in Table 4, our Vision-MoR X-Large backbone achieves the highest AP score of 39.1, outperforming competitive lightweight models such as SHViT-S4 (38.8) and EdgeViT-XXS (38.7), while also delivering the lowest inference latency on both GPU (0.22 ms) and CPU (4.4 ms). Notably, Vision-MoR also leads in AP metrics across small, medium, and large objects, indicating its robust multi-scale representation capability. These results confirm that the adaptive and recursive design of Vision-MoR not only benefits classification but also translates effectively to dense prediction tasks, offering a compelling balance of accuracy and efficiency for downstream vision applications.

Vision-MoR on Different Input Resolutions

We further investigate the resolution scalability of Vision-MoR by evaluating its inference speed and accuracy across a broad range of input resolutions, from 112×112 to 1120×1120. As shown in Figure 4, Vision-MoR consistently outperforms SHViT variants in both throughput and accuracy at every resolution setting. Notably, **Vision-MoR Small** achieves 74.6% Top-1 accuracy at the standard 224×224 resolution, which improves to 79.6% at 1120×1120, while maintaining a reasonable throughput of 1548 images/s. Similarly, **Vision-MoR Large** scales from 79.0% to 83.9% accuracy over the same range, with throughput remaining consistently higher than SHViT-S3 under identical conditions. This trend underscores the efficiency of our adaptive recursion mechanism, which enables deeper feature refinement without introducing prohibitive computational cost. In contrast, SHViT models exhibit a steeper decline in throughput and slower accuracy gains as resolution increases, highlighting their limited scalability. Overall, these results demonstrate that Vision-MoR not only delivers strong baseline performance at standard resolutions but also adapts effectively to high-resolution inputs, making it well-suited for vision tasks requiring fine-grained detail under real-time.

RetinaNet Object Detection on COCO								
Backbone	**Latency (ms)** GPU	CPU	**AP**	**AP$_{50}$**	**AP$_{75}$**	**AP$_s$**	**AP$_m$**	**AP$_l$**
MobileNetV3 (Koonce 2021)	0.34	7.5	29.9	49.3	30.8	14.9	33.3	41.1
EfficientViT-M4 (Pan et al. 2022)	0.33	7.3	32.7	52.2	34.1	17.6	35.3	46.0
PVTv2-B0 (Wang et al. 2022)	0.73	115.4	37.2	57.2	39.5	**23.1**	40.4	49.7
MobileFormer-508M (Chen et al. 2022)	0.89	35.7	38.0	58.3	40.3	22.9	41.2	49.7
EdgeViT-XXS (Pan et al. 2022)	0.88	38.4	38.7	59.0	41.0	22.4	42.0	51.6
SHViT-S4 (Yun and Ro 2024)	0.28	5.0	38.8	59.8	**41.1**	22.0	**42.4**	52.7
Vision-MoR X-Large	**0.22**	**4.4**	**39.1**	**59.9**	41.0	**23.2**	**42.4**	**53.1**

Table 4: Comparison results on object detection and instance segmentation on COCO 2017 using RetinaNet.

Expert-choice Router				**Performers**		**Token-choice Router**				**Performers**	
Sampling	**Func**	**Arch**	**z-loss**	**Top-1**	**Top-5**	**Balancing**	**Func**	**Arch**	**z-loss**	**Top-1**	**Top-5**
Aux Rout	σ	Spatial Router	✓	55.4	61.5	Loss-free	soft	Spatial Router	✗	73.5	91.9
Aux Loss	σ	Spatial Router	✗	76.5	92.4	Loss (0.1)	soft	Spatial Router	✓	73.5	91.7
Aux Loss	tanh	Spatial Router	✓	76.4	92.4	Loss (0.1)	σ	Spatial Router	✗	64.2	79.2
Aux Loss	σ	Linear	✓	70.1	79.9	Loss (0.1)	soft	Linear	✗	65.6	79.4
Aux Loss	σ	MLP	✓	70.2	81.1	Loss (0.1)	soft	MLP	✗	69.8	80.9
Aux Loss	σ	Spatial Router	✓	77.1	**92.8**	Loss (0.1)	soft	Spatial Router	✗	74.2	**91.9**

Table 5: Ablation results for expert-choice (*Left*) and token-choice (*Right*) routers with various design choices.

Ablation Study

We conduct extensive ablation studies to evaluate various design choices for routing in our Vision-MoR framework, focusing on two configurations: expert-choice and token-choice routing. These studies examine the impact of different supervision signals (auxiliary router vs. auxiliary loss), normalization functions (sigmoid vs. tanh), router architectures (Linear, MLP, Spatial Router), and regularization methods (with/without Z-loss). As shown in Table 5, these choices substantially affect routing quality and final recognition accuracy.

In the expert-choice setting, where each token dynamically decides its own depth during recursive computation, we find that using an auxiliary loss significantly outperforms an auxiliary router, confirming the findings of prior MoR studies. In particular, applying a sigmoid activation within our spatial router module, coupled with Z-loss regularization, yields the highest performance: 77.1% Top-1 and 92.8% Top-5 accuracy. This validates our Methodology design, where we explicitly adopt this configuration. Notably, this design surpasses simpler MLP or Linear routers, suggesting that capturing spatial context via shifted attention is crucial for accurate token-level gating decisions.

In the token-choice setting, which selects a global recursion depth per token at initialization, we assess the trade-off between explicit balancing loss and bias-based loss-free routing. While both achieve similar results with softmax normalization, our method—using a balancing loss of 0.1, spatial router, and no Z-loss—achieves the best balance between accuracy and routing stability, with 74.2% Top-1 and 91.9% Top-5 accuracy. This again supports the methodology we proposed, where a light balancing signal combined with soft competition leads to robust load balancing without over-regularizing the routing distribution.

Conclusion

In this paper, we introduced Vision-MoR, a unified and efficient Vision Transformer framework that jointly optimizes parameter reuse, spatially adaptive computation, and memory efficiency. By integrating a spatial-aware router with recursive transformer blocks, Vision-MoR enables dynamic, patch-wise depth allocation and early exiting, allowing the model to allocate more compute to informative regions while skipping redundant ones. Our spatial router leverages shifted window attention for context-aware routing, and a middle-cycle recursion strategy for parameter sharing without loss of expressivity. On ImageNet-1K, Vision-MoR X-Large achieves 80.4% Top-1 with just 14.3M parameters—both outperforming existing models at similar or higher costs. These results validate that Vision-MoR offers a principled and scalable solution for efficient visual recognition across a wide range of tasks and deployment scenarios.

References

Alabdulmohsin, I. M.; Zhai, X.; Kolesnikov, A.; and Beyer, L. 2023. Getting vit in shape: Scaling laws for compute-optimal model design. *Advances in Neural Information Processing Systems*, 36: 16406–16425.

Ataiefard, F.; Ahmed, W.; Hajimolahoseini, H.; Asani, S.;

et al. 2024. SkipViT: Speeding Up Vision Transformers with a Token-Level Skip Connection. *CoRR*, abs/2401.15293. ArXiv preprint.

Bae, S.; Kim, Y.; Bayat, R.; Kim, S.; Ha, J.; Schuster, T.; Fisch, A.; Harutyunyan, H.; Ji, Z.; Courville, A.; et al. 2025. Mixture-of-Recursions: Learning Dynamic Recursive Depths for Adaptive Token-Level Computation. *arXiv preprint arXiv:2507.10524*.

Bolya, D.; Fu, C.-Y.; Dai, X.; Zhang, P.; Feichtenhofer, C.; and Hoffman, J. 2022. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*.

Chen, J.; Kao, S.-h.; He, H.; Zhuo, W.; Wen, S.; Lee, C.-H.; and Chan, S.-H. G. 2023. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. *arXiv preprint arXiv:2303.03667*.

Chen, Y.; Dai, X.; Chen, D.; Liu, M.; Dong, X.; Yuan, L.; and Liu, Z. 2022. Mobile-Former: Bridging MobileNet and Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.

Graham, B.; El-Nouby, A.; Touvron, H.; Stock, P.; Joulin, A.; Jégou, H.; and Douze, M. 2021. LeViT: A Vision Transformer in ConvNet's Clothing for Faster Inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 12259–12269.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

He, Y.; Yuan, Z.; Tu, Z.; Ye, Y.; and Sun, L. 2025. 3D4D: An Interactive, Editable, 4D World Model via 3D Video Generation. *arXiv preprint arXiv:2511.08536*.

Koonce, B. 2021. MobileNetV3. In *Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization*, 125–144. Springer.

Li, Y.; Hu, J.; Wen, Y.; Evangelidis, G.; Salahi, K.; Wang, Y.; Tulyakov, S.; and Ren, J. 2023a. Rethinking Vision Transformers for MobileNet Size and Speed. In *Proceedings of ICCV 2023*.

Li, Z.; Xiao, J.; Yang, L.; and Gu, Q. 2023b. Repq-vit: Scale reparameterization for post-training quantization of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17227–17236.

Liu, X.; Peng, H.; Zheng, N.; Yang, Y.; Hu, H.; and Yuan, Y. 2023. Efficientvit: Memory efficient vision transformer with cascaded group attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 14420–14430.

Liu, X.; Wu, T.; and Guo, G. 2022. Adaptive sparse vit: Towards learnable adaptive token pruning by fully exploiting self-attention. *arXiv preprint arXiv:2209.13802*.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10012–10022.

Loshchilov, I.; and Hutter, F. 2018. Decoupled Weight Decay Regularization. In *ICLR*.

Ma, N.; Zhang, X.; Zheng, H.-T.; and Sun, J. 2018. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, 116–131.

Mehta, S.; and Rastegari, M. 2021. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*.

Mehta, S.; and Rastegari, M. 2022. MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer. In *International Conference on Learning Representations (ICLR)*.

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

Pan, J.; Bulat, A.; Tan, F.; Zhu, X.; Dudziak, L.; Li, H.; Tzimiropoulos, G.; and Martinez, B. 2022. EdgeViTs: Competing Light-weight CNNs on Mobile Devices with Vision Transformers. In *Proceedings of ECCV*.

Paoletti, M. E.; Haut, J. M.; Pereira, N. S.; Plaza, J.; and Plaza, A. 2021. Ghostnet for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 59(12): 10378–10393.

Papa, L.; Russo, P.; Amerini, I.; and Zhou, L. 2024a. A survey on efficient vision transformers: algorithms, techniques, and performance benchmarking. *IEEE transactions on pattern analysis and machine intelligence*, 46(12): 7682–7700.

Papa, L.; Russo, P.; Amerini, I.; and Zhou, L. 2024b. A survey on efficient vision transformers: algorithms, techniques, and performance benchmarking. *IEEE transactions on pattern analysis and machine intelligence*, 46(12): 7682–7700.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763. PmLR.

Rao, Y.; Zhao, W.; Liu, B.; Lu, J.; Zhou, J.; and Hsieh, C.-J. 2021. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34: 13937–13949.

Shen, Z.; Liu, Z.; and Xing, E. 2021. Sliced Recursive Transformer. *arXiv preprint arXiv:2111.05297*. Also presented at conference.

Tan, M.; and Le, Q. V. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 6105–6114.

Tay, Y.; Bahri, D.; Metzler, D.; Juan, D.-C.; Zhao, Z.; and Zheng, C. 2021. Synthesizer: Rethinking self-attention for transformer models. In *International conference on machine learning*, 10183–10192. PMLR.

Thisanke, H.; Deshan, C.; Chamith, K.; Seneviratne, S.; Vidanaarachchi, R.; and Herath, D. 2023. Semantic segmentation using vision transformers: A survey. *Engineering Applications of Artificial Intelligence*, 126: 106669.

Tschannen, M.; Gritsenko, A.; Wang, X.; Naeem, M. F.; Alabdulmohsin, I.; Parthasarathy, N.; Evans, T.; Beyer, L.; Xia, Y.; Mustafa, B.; et al. 2025. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*.

Vasu, P. K. A.; Gabriel, J.; Zhu, J.; Tuzel, O.; and Ranjan, A. 2023a. FastViT: A Fast Hybrid Vision Transformer Using Structural Reparameterization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Vasu, P. K. A.; Gabriel, J.; Zhu, J.; Tuzel, O.; and Ranjan, A. 2023b. MobileOne: An Improved One millisecond Mobile Backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, S.; Li, B. Z.; Khabsa, M.; Fang, H.; and Ma, H. 2020. Linformer: Self-Attention with Linear Complexity. *arXiv preprint arXiv:2006.04768*.

Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2021. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, 568–578.

Wang, W.; Xie, E.; Li, X.; Fan, D.-P.; Song, K.; Liang, D.; Lu, T.; Luo, P.; and Shao, L. 2022. Pvt v2: Improved baselines with pyramid vision transformer. *Computational visual media*, 8(3): 415–424.

Wang, X.; Song, D.; Chen, S.; Zhang, C.; and Wang, B. 2024. Longllava: Scaling multi-modal llms to 1000 images efficiently via a hybrid architecture. *arXiv preprint arXiv:2409.02889*.

Wei, L.; Li, Y.; Wang, C.; Wang, Y.; Kong, L.; Huang, W.; and Sun, L. 2025. Unsupervised Post-Training for Multi-Modal LLM Reasoning via GRPO. *arXiv preprint arXiv:2505.22453*.

Wu, K.; Zhang, J.; Peng, H.; Liu, M.; Xiao, B.; Fu, J.; and Yuan, L. 2022. Tinyvit: Fast pretraining distillation for small vision transformers. In *European conference on computer vision*, 68–85. Springer.

Xie, H.; Peng, C.-J.; Tseng, Y.-W.; Chen, H.-J.; Hsu, C.-F.; Shuai, H.-H.; and Cheng, W.-H. 2024. Emovit: Revolutionizing emotion insights with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 26596–26605.

Xu, K.; Wang, Z.; Chen, C.; Geng, X.; Lin, J.; Yang, X.; Wu, M.; Li, X.; and Lin, W. 2024. Lpvit: Low-power semi-structured pruning for vision transformers. In *European Conference on Computer Vision*, 269–287. Springer.

Yin, H.; Vahdat, A.; Alvarez, J. M.; Mallya, A.; Kautz, J.; and Molchanov, P. 2022. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10809–10818.

Yu, W.; Luo, M.; Zhou, P.; Si, C.; Zhou, Y.; Wang, X.; Feng, J.; and Yan, S. 2022. MetaFormer Is Actually What You Need for Vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10819–10829.

Yuan, Z.; Zhou, R.; Wang, H.; He, L.; Ye, Y.; and Sun, L. 2024. Vit-1.58 b: Mobile vision transformers in the 1-bit era. *arXiv preprint arXiv:2406.18051*.

Yun, S.; and Ro, Y. 2024. SHViT: Single-Head Vision Transformer with Memory Efficient Macro Design. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5756–5767.

Zhang, D.; Yan, R.; Dong, P.; and Cheng, K.-T. 2025. Memory efficient transformer adapter for dense predictions. *arXiv preprint arXiv:2502.01962*.

Zhang, J.; Li, X.; Li, J.; Liu, L.; Xue, Z.; Zhang, B.; Jiang, Z.; Huang, T.; Wang, Y.; and Wang, C. 2023. Rethinking Mobile Block for Efficient Attention-based Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1389–1400.

Zhang, J.; Peng, H.; Wu, K.; Liu, M.; Xiao, B.; Fu, J.; and Yuan, L. 2022. Minivit: Compressing vision transformers with weight multiplexing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12145–12154.