PartialNet: Compute Less, Perform Better

Haiduo Huang, Tian Xia, Wenzhe Zhao, Pengju Ren*

Xi'an Jiaotong University
huanghd@stu.xjtu.edu.cn, tian_xia@xjtu.edu.cn, wenzhe@xjtu.edu.cn, pengjuren@xjtu.edu.cn

Abstract

Achieving a balance between low parameter count, reduced FLOPs, and high accuracy and throughput remains a central challenge in neural network design. To address this, we propose the **p**artial **c**hannel **m**echanism (**PCM**), which leverages the inherent redundancy in feature map channels. PCM divides feature map channels into multiple groups, each processed by distinct operations such as convolution, attention, pooling, or identity mapping. Building on this, we introduce **p**artial **at**tention **conv**olution (**PATConv**), a novel module that efficiently fuses convolution and visual attention within a unified framework. Our results demonstrate that PATConv can fully replace both standard convolution and visual attention modules, leading to significant reductions in parameters and FLOPs. Furthermore, PATConv enables three efficient visual attention variants: Partial Channel Attention, Partial Spatial Attention, and Partial Self-Attention. To further optimize the allocation of channel splits, we propose **d**ynamic **p**artial **conv**olution (**DPConv**), which adaptively learns the optimal split ratio for each layer, achieving a better trade-off between speed and accuracy. By integrating PATConv and DPConv, we develop a new hybrid network family, **PartialNet**, which achieves superior top-1 accuracy and inference speed on ImageNet-1K, and demonstrates strong performance on COCO detection and segmentation tasks.

Code — https://github.com/haiduo/PartialNet

Introduction

The pursuit of efficient and effective neural network architectures remains a key focus in computer vision. Depthwise separable convolution (DWConv) (Howard et al. 2017) has been widely adopted to reduce computational cost and parameter count, as seen in various CNN-based (Sandler et al. 2018; Tan and Le 2019a) and hybrid (Yang et al. 2022; Hou et al. 2022; Rao et al. 2022) models. However, DWConv often suffers from frequent memory access and limited parallelism during inference (Ma et al. 2018; Ding et al. 2022; Chen et al. 2023), resulting in suboptimal throughput.

Given the significant redundancy present in feature maps (Han et al. 2020; Chen et al. 2023), we propose the Partial Channel Mechanism (PCM) to further reduce computational cost and parameters while enhancing inference speed and accuracy. PCM operates by splitting feature maps into several groups, each processed by a different operation, and then concatenating the results. This design allows us to strategically allocate lightweight, efficient operators to replace expensive, dense ones, thereby improving both speed and accuracy. Specifically, we replace computationally intensive convolutions with more efficient visual attention mechanisms, enhancing representational power and inference efficiency. This leads to the development of Partial Attention Convolution (PATConv), which unifies partial convolution and partial visual attention, as illustrated in Figure 1, and can serve as a drop-in replacement for both standard convolution and attention modules while reducing overall model complexity.

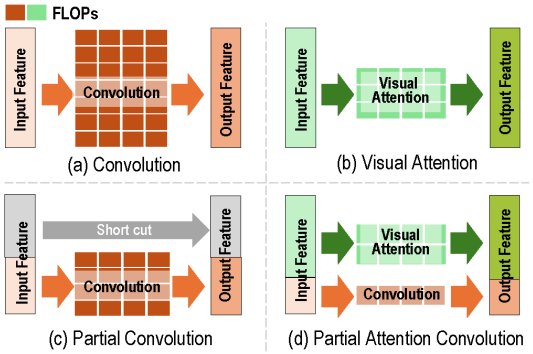

Figure 1: Comparison of different operation types.

While some prior works have explored feature map splitting, their approaches are often simplistic—processing only a subset of channels and leaving the remainder untouched. For example, ShuffleNet V2 (Ma et al. 2018) employs a "Channel Split" operation, where one group of channels is directly passed through, and the other is processed by convolutional layers. Similarly, FasterNet (Chen et al. 2023) introduces partial convolution (PConv), applying convolution to only a portion of the input channels. Although these methods improve inference speed, they overlook the potential of the untouched channels and focus solely on reducing the number of processed channels.

In contrast, we advocate for a holistic approach that consid-

*Corresponding author

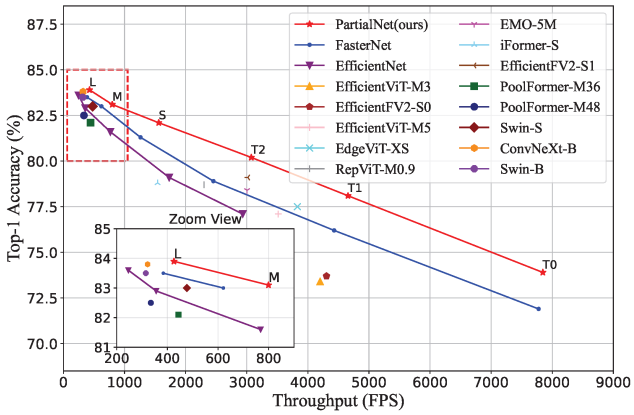

Figure 2: Our PartialNet achieves higher trade-off of accuracy and throughput on ImageNet-1K.

ers FLOPs, inference speed, and accuracy by fully utilizing all feature channels. We propose to combine convolution and attention, each applied to a subset of channels. Since effective partial channel attention mechanisms are lacking, we introduce three efficient partial visual attention variants: (1) Partial Channel Attention (PAT_{ch}), which incorporates an enhanced Gaussian channel attention (Hu, Shen, and Sun 2018) for richer inter-channel interactions; (2) Partial Spatial Attention (PAT_{sp}), which brings spatial attention into the MLP layer to further boost accuracy; and (3) Partial Self-Attention (PAT_{sf}), inspired by the MetaFormer paradigm (Yu et al. 2022), which integrates global self-attention in the final stage to expand the receptive field.

Despite these advances, the optimal split ratio between partial convolution and partial visual attention remains crucial for balancing latency and parameter efficiency. To this end, we draw inspiration from dynamic group convolution (Zhang et al. 2019) and propose Dynamic Partial Convolution (DPConv), which adaptively learns the split ratio for each layer based on constraints such as parameters and latency, achieving an optimal trade-off between speed and accuracy.

In summary, our enhanced model, **PartialNet**, delivers consistent performance improvements on ImageNet-1K classification while maintaining competitive throughput, as shown in Figure 2. We present PartialNet, which integrates parallel partial attention and convolution with adaptive channel splitting to achieve efficient and accurate vision models.

Related Work

Recent progress in efficient neural network architectures has centered on balancing computational cost, inference speed, and accuracy. Depthwise separable convolution (DWConv) remains a foundational component for lightweight CNNs, as demonstrated in MobileNets (Sandler et al. 2018; Howard et al. 2019), EfficientNets (Tan and Le 2019a, 2021), MobileViT (Mehta and Rastegari 2021), and EdgeViT (Pan et al. 2022). However, DWConv's limited parallelism and hardware utilization have motivated the development of alternatives such as large-kernel DWConv (Ding et al. 2022), pooling-based spatial modules (Yu et al. 2022), and partial convolution (Chen et al. 2023). While these methods improve efficiency, they often involve trade-offs between speed and accuracy. Our PartialNet addresses these challenges by integrating visual attention into convolutional operations, achieving a better balance between efficiency and representational power. Attention mechanisms are pivotal in Vision Transformers (ViTs) (Raghu et al. 2021; Paul and Chen 2022), encompassing channel, spatial, and self-attention. Efficient self-attention variants, such as linear attention (Wang et al. 2020; Cai et al. 2023), have been proposed to reduce computational complexity. Channel and spatial attention modules, validated in works like SE-Net (Hu, Shen, and Sun 2018) and CBAM (Woo et al. 2018), remain effective for enhancing feature representation. Our approach leverages **parallel partial attention and convolution**, mitigating the inference overhead typically introduced by element-wise operations.

Hybrid designs that combine convolutional and transformer modules, such as EMOv2 (Zhang et al. 2024) and iFormer (Zheng 2025), further demonstrate the advantages of integrating local and global feature modeling. In addition, advanced training and optimization techniques—including Neural Architecture Search (NAS) (Tan et al. 2019), structured re-parameterization (Vasu et al. 2023; Wang et al. 2023b), knowledge distillation (Huang et al. 2022; Graham et al. 2021), and self-supervised pre-training (Woo et al. 2023)—have contributed to recent performance gains. In contrast, PartialNet achieves strong results using standard training protocols without reliance on such advanced tricks. More discussions are provided in Appendix A.

Methodology

Partial Channel Mechanism

Designing efficient neural networks requires a holistic optimization of computational cost (FLOPs), model size, memory access, and accuracy. While recent approaches such as MobileViTv2 (Mehta and Rastegari 2022) and EfficientViT (Liu et al. 2023) combine depthwise separable convolutions with self-attention to reduce parameters and latency, other architectures like ShuffleNetv2 (Ma et al. 2018) and FasterNet (Chen et al. 2023) focus on minimizing FLOPs by processing only a subset of feature map channels. However, these strategies often *fall short in terms of accuracy improvements* when compared to models with similar computational budgets. For example, FasterNet leverages partial convolution to achieve high inference speed across hardware platforms, but the limited channel-wise processing restricts feature interaction and global information exchange, which can ultimately hinder model performance.

To address these limitations, we propose a **partial channel mechanism** that fully utilizes all feature map channels by assigning distinct operations to different channel groups. Specifically, we introduce Partial Attention Convolution (PATConv), which replaces dense convolution with a combination of lightweight visual attention and convolution. Prior studies (Han et al. 2020; Chen et al. 2023) have shown that feature channels are often redundant, suggesting that applying attention to only a subset of channels can efficiently capture global context. PATConv leverages this redundancy to re-

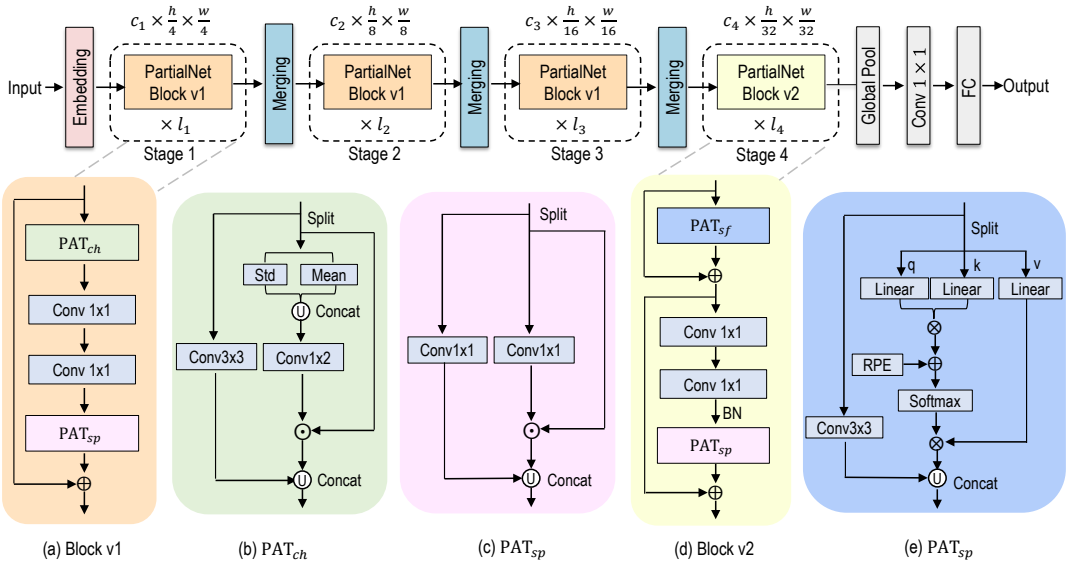

Figure 3: The overall architecture of PartialNet. The network is organized into four hierarchical stages, each comprising a stack of PartialNet blocks, followed by embedding and merging layers for spatial downsampling and channel expansion. The final three layers are dedicated to feature classification. Here, $\odot$ and $\otimes$ denote element-wise multiplication and matrix multiplication.

duce FLOPs while maintaining or even improving accuracy through targeted feature interaction.

Unlike conventional visual attention modules that process all channels, PATConv achieves higher efficiency by applying the computationally less intensive attention operation to a larger portion of the channels, while the remaining channels are processed by convolution. This parallel design enables both branches (convolution and attention) to operate simultaneously, maximizing GPU utilization and throughput (Kirk and Wen-Mei 2016). Formally, let the input feature map be $\mathbf{F} \in \mathbb{R}^{h \times w \times c_{\text{in}}}$ and the output be $\mathbf{O} \in \mathbb{R}^{h \times w \times c_{\text{out}}}$, where h and w are spatial dimensions, and c_{in}, c_{out} are the input and output channel dimensions. The PATConv operation is defined as:

$$\mathbf{O} = \text{PATConv}(\mathbf{F}) = \mathcal{C}(\mathbf{F}[..., : r_p c_{\text{in}}]) \text{U} \mathcal{A}(\mathbf{F}[..., r_p c_{\text{in}} :]), \quad (1)$$

where $\mathcal{C}(\cdot)$ denotes the convolution operation, $\mathcal{A}(\cdot)$ denotes the attention operation, U denotes channel-wise concatenation, and $r_p \in (0, 1)$ is an adaptive channel splitting ratio. The output and input dimensions are kept consistent, i.e., $\dim(\mathbf{O}) = \dim(\mathbf{F})$. PATConv supports both channel-wise and spatial-wise mixing, and can incorporate self-attention to further expand the receptive field. We instantiate three efficient variants to demonstrate the flexibility and effectiveness of this mechanism:

PAT_{ch}: Channel Attention with Gaussian Statistics. We integrate 3×3 convolution with a channel attention module that captures global spatial interactions using enhanced Gaussian statistics. Unlike SENet (Hu, Shen, and Sun 2018), which only considers channel means, our approach also incorporates standard deviation, leveraging the observation that feature maps tend to follow a normal distribution during training (Ioffe and Szegedy 2015). This enables richer channel-wise representations, as illustrated in Figure 3(b) and detailed in Appendix Figure 7 (a). For each channel $c \in \{1, ..., (1 - r_p)c_{\text{in}}\}$, we compute:

$$\mu_c = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \mathbf{F}_{i,j,c} \quad (2)$$

$$\text{std}_c = \sqrt{\frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} (\mathbf{F}_{i,j,c} - \mu_c)^2 + \epsilon} \quad (3)$$

where ϵ is a small constant for numerical stability. The statistics are concatenated as $\mathbf{z} = [\mu, \text{std}] \in \mathbb{R}^{1 \times 2 \times (1 - r_p)c_{\text{in}}}$. The excitation operation applies a single fully connected layer:

$$\mathbf{s} = \text{BN}(\text{Sigmoid}(\mathbf{W} * \mathbf{z})), \ \tilde{\mathbf{F}} = \mathbf{F} \odot \mathbf{s} \quad (4)$$

where $\mathbf{W} \in \mathbb{R}^{(1 - r_p)c_{\text{in}} \times 2}$ are learnable weights, BN denotes batch normalization, and Sigmoid denotes the sigmoid activation function. The recalibrated features $\tilde{\mathbf{F}}$ are then concatenated with the output of the partial 3×3 convolution.

PAT_{sp}: Spatial Attention Integration. We further combine spatial attention with 1×1 convolution, as both perform channel-wise mixing. The spatial attention branch uses pointwise convolution to aggregate channel information into a single-channel spatial map, which is then activated by a Hard-Sigmoid function and used to reweight the features. This module is typically placed in the MLP layer, and its 1×1 convolution can be merged with the MLP's second 1×1 convolution during inference (see Figure 3(c) and Appendix Figure 7 (b,c)), minimizing the runtime overhead.

$$\mathbf{s} = \text{BN}(\text{Hard-Sigmoid}(\mathbf{W} * \mathbf{F})), \ \tilde{\mathbf{F}} = \mathbf{F} \odot \mathbf{s} \quad (5)$$

where $\mathbf{W} \in \mathbb{R}^{1 \times 1 \times (1 - r_p)c_{\text{in}}}$ are learnable weights, and $\odot$ denotes spatial broadcasting multiplication.

PAT_{sf}: Efficient Self-Attention. To further enhance spatial interactions and expand the receptive field, we introduce a self-attention variant, PAT_{sf}, which can substitute channel

attention. To maintain efficiency given the quadratic complexity of self-attention, we restrict PAT_{sf} to the final stage of the network. We also incorporate relative position encoding (RPE) (Wu et al. 2021) to improve accuracy. The module is detailed in Appendix Figure 7 (c). The self-attention process:

$$\mathbf{s} = \text{Softmax}\left(\frac{(\mathbf{W}_q\mathbf{F})(\mathbf{W}_k\mathbf{F})^\top}{\sqrt{d}} + \mathbf{R}\right), \ \tilde{\mathbf{F}} = (\mathbf{W}_v * \mathbf{F}) * \mathbf{s} \quad (6)$$

where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{d\times 1\times 1\times (1-r_p)c_{\text{in}}}$ are the query, key, and value projections, $\mathbf{R} \in \mathbb{R}^{HW\times HW}$ is the relative position encoding, and $d = HW$ is the scaling factor.

Architectural Advantages. Unlike traditional CNNs that apply attention and convolution sequentially, our parallel design processes both operations simultaneously on the same input features, leading to improved speed-accuracy trade-offs. Moreover, the modular nature of PATConv allows for flexible integration with a variety of visual attention mechanisms, not limited to the three variants described above.

Learnable Dynamic Partial Convolution

In PATConv, the split ratio $r_p \in (0, 1)$ is a crucial hyperparameter that directly influences both the model's parameter count and computational efficiency. If r_p is set too high, PATConv degenerates into a standard convolution, losing the benefits of global attention. Conversely, a too-small r_p results in insufficient local inductive bias. While prior works such as FasterNet adopt a fixed split ratio (e.g., $r_p = \frac{1}{4}$) across all layers, we introduce Dynamic Partial Convolution (DPConv), where r_p is learnable and adaptively optimized for each layer to achieve the best trade-off between efficiency and representational power.

To learn adaptive split ratios for each layer, DPConv introduces learnable gating variables that control channel allocation. Instead of directly learning a full binary connectivity matrix (which would require $O(c^2)$ memory), we employ a structured decomposition that reduces memory overhead to $O(\log_2 c)$. Specifically, for channels $c = 2^K$, we use K learnable gating parameters $\widetilde{g}_k \in [-1, 1]$ that are binarized via $g_k = \text{Sign}(\widetilde{g}_k) \in \{0, 1\}$ with straight-through estimation (Courbariaux et al. 2016). These gates control a hierarchical channel splitting pattern: larger g_k values activate more channels for convolution, while smaller values allocate more channels to attention. The split ratio r_p is computed as $r_p = 2^{\sum_{k=1}^{K}(1-g_k)}/c_{\text{in}}$, which determines how many channels are processed by convolution versus attention. This decomposition enables efficient learning with minimal memory overhead. The initialization $\widetilde{g}_k \sim \mathcal{U}(-1, 1)$ ensures balanced exploration. The DPConv operation processes the input feature map by applying convolution to the first $r_p c_{\text{in}}$ channels and attention to the remaining $(1 - r_p)c_{\text{in}}$ channels:

$$\mathbf{O} = \text{DPConv}(\mathbf{F}) = \text{Conv}(\mathbf{F}[..., : r_p c_{\text{in}}]) \ \text{U} \ \text{Attn}(\mathbf{F}[..., r_p c_{\text{in}} :]) \quad (7)$$

The convolution weights $\mathbf{W} \in \mathbb{R}^{k\times k\times r_p c_{\text{in}}\times r_p c_{\text{out}}}$ have dimensions $k^2 \times r_p c_{\text{in}} \times r_p c_{\text{out}}$ for kernel size k (e.g., $k = 3$ for 3×3 convolution), while the attention branch uses projection matrices $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{(1-r_p)c_{\text{in}}\times(1-r_p)c_{\text{out}}}$ that process only $(1 - r_p)c_{\text{in}}$ channels. The overall DPConv generation process is illustrated in Figure 4.

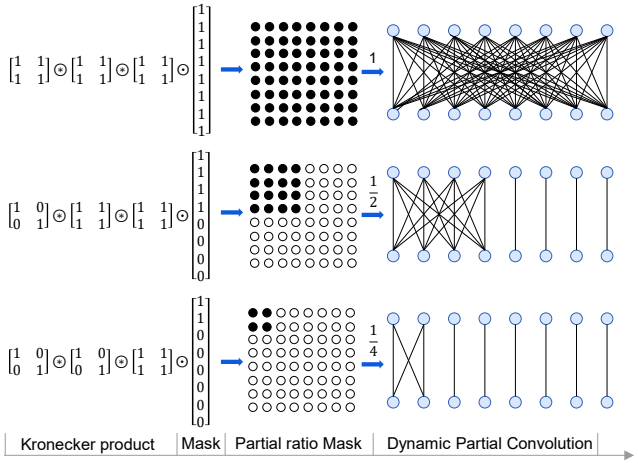

Figure 4: Architecture of Dynamic Partial Convolution. The Kronecker product ($\circledast$) constructs the binary matrix U, which is then elementwise multiplied ($\odot$) with the channel mask $\mathbf{m}$ and convolution weights $\mathbf{W}$. This decomposition enables hardware-friendly and efficient implementation.

Complexity-Aware Regularization To ensure hardware efficiency, we introduce a resource-aware regularization strategy. The model complexity ζ is controlled by the gating variables: $\zeta = \sum_{l=1}^{L} 2^{2\sum_{k=1}^{K_l} g_k^l}$, where larger g_k^l values increase the number of active parameters. Given a target complexity $\kappa = \sum_{l=1}^{L}(c_l/\theta)^2$, where $\theta \geq 1$ controls the compression ratio, we formulate the following constrained optimization. The parameter θ determines the desired model complexity: larger values encourage higher compression (fewer parameters), while smaller values allow more parameters. For example, $\theta = 4$ roughly corresponds to FasterNet-equivalent FLOPs. The relationship between θ, parameters, FLOPs, and accuracy is analyzed in Table 7 (Appendix), showing that $\theta \in [3, 5]$ typically provides good accuracy-efficiency trade-offs. For detailed theoretical analysis, see Appendix C. The constrained optimization is formulated as:

$$\min_{\mathbf{W},\widetilde{\mathbf{g}}} \mathcal{L}(\mathbf{W}, \widetilde{\mathbf{g}}) \cdot \left[\frac{\kappa}{\zeta}\right]^{\alpha} + \beta\psi, \quad \text{s.t.} \quad \alpha = \begin{cases} 0 & \zeta \leq \kappa \\ -0.01 & \text{otherwise} \end{cases} \quad (8)$$

where $\beta = 0.9$ controls the regularization strength, and $\psi = \sum_{l=1}^{L} \psi_l$ enforces monotonicity of the gates via $\psi_l = \sum_{i=1}^{K_l-1} \max(0, \widetilde{g}_{i+1}^l - \widetilde{g}_i^l)$. During training, we apply temperature annealing to the gates: $\widetilde{g}_k^l \leftarrow \widetilde{g}_k^l / \sqrt{t/T}$, where t is the current epoch and T is the total number of epochs. The complete training procedure is summarized in Algorithm 1.

PartialNet Architecture

The overall PartialNet architecture, illustrated in Figure 3, is organized into four hierarchical stages. Each stage is preceded by either an embedding layer (a 4×4 convolution with stride 4) or a merging layer (a 2×2 convolution with stride 2), which together perform spatial downsampling and channel expansion. The network is constructed from two specialized building blocks:

- **PartialNet Block v1** (Stages 1-3): Combines PAT_{ch} and PAT_{sp} modules, as shown in Figure 3(a).
- **PartialNet Block v2** (Stage 4): Replaces PAT_{ch} with PAT_{sf} and modifies shortcut connections for improved training stability, as depicted in Figure 3(d).

Algorithm 1: Complexity-Constrained DPConv Training

Require: Training set $\mathcal{D}$, target complexity κ, initial weights $\mathbf{W} \sim \mathcal{N}(0, 0.02)$, and gates $\widetilde{\mathbf{g}} \sim \mathcal{U}(-1, 1)$

for epoch $t = 1$ to T **do**
  for each layer l **do**
    Compute binary gates: $g_k^l = \text{Sign}(\widetilde{g}_k^l)$
    Construct U^l
  end for
  Calculate current complexity: $\zeta_t = \sum_l 2^{2\sum_k g_k^l}$
  Set adaptive weight:

$$\alpha_t = \begin{cases} 0 & \zeta_t \le \kappa \\ -0.01 & \text{otherwise} \end{cases}$$

  Compute loss:

$$\mathcal{L}_t = \mathcal{L}_{\text{task}} \cdot (\kappa/\zeta_t)^{\alpha_t} + 0.9\psi_t$$

  Update $\mathbf{W}$ and $\widetilde{\mathbf{g}}$ via Adam optimizer
  for each layer l **do**
    for $k = 1$ to $K_l - 1$ **do**
      Enforce monotonicity: $\widetilde{g}_k^l \leftarrow \max(\widetilde{g}_k^l, \widetilde{g}_{k+1}^l + \epsilon)$
    end for
    Apply temperature annealing: $\widetilde{g}_k^l \leftarrow \widetilde{g}_k^l / \sqrt{t/T}$
  end for
end for

To maximize throughput and maintain feature diversity, normalization or activation layers are applied only after each intermediate Conv1×1. Batch normalization is fused into adjacent convolution layers to accelerate inference without compromising accuracy. For activation, smaller PartialNet variants use GELU, while larger variants employ ReLU. The final three layers consist of global average pooling, a Conv1×1, and a fully connected layer for feature transformation and classification. We provide tiny, small, medium, and large variants—PartialNet-T0, PartialNet-T1, PartialNet-T2, PartialNet-S, PartialNet-M, and PartialNet-L—which share the same architectural design but scale in depth and channel width. Detailed configurations are listed in Appendix Table 7.

Experiments

PartialNet on ImageNet-1K Classification

Setup. We conduct experiments on the widely used ImageNet-1K dataset, which contains 1,000 classes, approximately 1.3M training images, and 50K validation images. All PartialNet models are trained for 300 epochs using the AdamW optimizer, with a 20-epoch linear warm-up. We follow the same regularization, augmentation, and multi-scale training strategies as FasterNet for fair comparison. Detailed training configurations are provided in Table 6 in the appendix. For inference benchmarking, throughput is measured on Nvidia V100 with a batch size of 256, and latency is evaluated on a single core of an AMD EPYC™ 73F3 CPU.

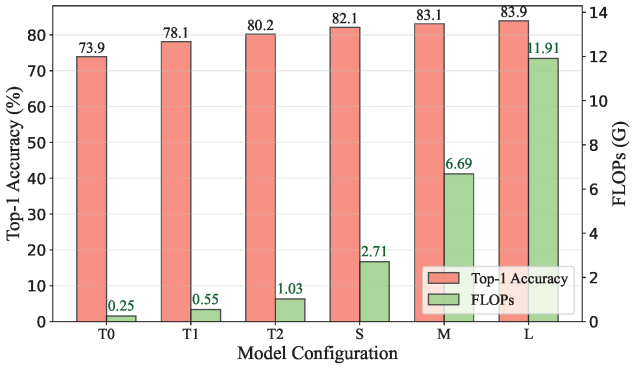

Figure 5: The relationship between FLOPs and Top-1 Accuracy in different PartialNet variants.

Results. Table 1 presents a comprehensive comparison between PartialNet (T0, T1, T2, S, M, L) and representative CNN and hybrid models. Across all model sizes, PartialNet consistently surpasses recent strong baselines, including FasterNet (Chen et al. 2023), in both accuracy and efficiency. For example, PartialNet-T2 achieves a 1.3% higher top-1 accuracy than FasterNet-T2, while also delivering 25.2% higher throughput and 24.1% lower latency. These results confirm that the integration of partial visual attention and partial convolution in PartialNet leads to superior accuracy while maintaining high computational efficiency.

Further, Figure 5 illustrates PartialNet's efficient scaling: as model size increases from T0 to L, the accuracy improvement (ΔTop-1) significantly outpaces the increase in FLOPs (ΔFLOPs), with the accuracy-to-computation ratio (ΔTop-1/ΔFLOPs) exceeding that of conventional models by 2.1× in comparable complexity ranges. This demonstrates that PartialNet effectively translates additional computation into tangible accuracy gains.

PartialNet on Downstream Tasks

Setup. To evaluate transferability, we employ PartialNet as the backbone in Mask-RCNN (He et al. 2017) for object detection and instance segmentation on the MS-COCO 2017 dataset (118K training and 5K validation images). The training strictly follows the FasterNet protocol: AdamW optimizer, 12 epochs, batch size 16, input resolution 1333×800, and standardized hyperparameters without further tuning.

Results. As shown in Table 2, PartialNet consistently outperforms mainstream backbones across all scales. For example, PartialNet-S achieves 42.7% AP^b and 39.3% AP^m, surpassing FasterNet-S by 7.0% and 6.5% respectively, while reducing FLOPs by 16.3% (216G vs. 258G) and maintaining comparable throughput (122 FPS vs. 121 FPS). At larger scales, PartialNet-L attains 44.7% AP^b (+1.6% over FasterNet-L) and 41.0% AP^m (+2.8%), with 18.0% fewer FLOPs (397G vs. 484G). These findings are consistent with our ImageNet results (Table 1), confirming that PartialNet achieves an effective balance between efficiency and representational power across diverse vision tasks.

Network	Type	Params (M)	FLOPs (G)	Throughput (FPS)	Latency CPU (ms)	Top-1 (%)
FasterNet-T0 (Chen et al. 2023)	CNN	3.9	0.34	7777	13.5	71.9
MobileNetV2 (Sandler et al. 2018)	CNN	3.5	0.31	3924	13.7	72.0
EfficientViT-M3 (Liu et al. 2023)	Hybrid	6.9	0.26	4200	22.3	73.4
EfficientFormerV2-S0 (Li et al. 2023)	Hybrid	3.5	0.40	4305	16.9	73.7
PartialNet-T0 (**ours**)	Hybrid	4.2	0.25	**7846**	**12.2**	**73.9**
FasterNet-T1 (Chen et al. 2023)	CNN	7.6	0.85	4430	22.2	76.2
EfficientNet-B0 (Tan and Le 2021)	CNN	5.3	0.39	2934	22.7	77.1
EfficientViT-M5 (Liu et al. 2023)	Hybrid	12.4	0.52	3516	35.9	77.1
EdgeViT-XS (Pan et al. 2022)	Hybrid	6.7	1.10	3828	30.2	77.5
PartialNet-T1 (**ours**)	Hybrid	7.8	0.55	**4657**	**21.5**	**78.1**
RepViT-M0.9 (Wang et al. 2023a)	CNN	5.1	0.80	2300	40.5	78.7
FasterNet-T2 (Chen et al. 2023)	CNN	15.0	1.91	2455	43.7	78.9
EfficientNet-B1 (Tan and Le 2021)	CNN	7.8	0.70	1730	35.5	79.1
EMO-5M (Zhang et al. 2024)	Hybrid	5.1	0.55	3005	37.9	78.4
iFormer-S (Zheng 2025)	Hybrid	6.8	1.10	1540	42.4	78.8
EfficientFormerV2-S1 (Li et al. 2023)	Hybrid	6.1	0.65	3020	36.3	79.1
PartialNet-T2 (**ours**)	Hybrid	12.6	1.03	**3074**	**35.2**	**80.2**
FasterNet-S (Chen et al. 2023)	CNN	31.1	4.56	1261	96.0	81.3
EfficientNet-B3 (Tan and Le 2021)	CNN	12.0	1.80	768	73.5	81.6
MobileViTv2-2.0 (Mehta and Rastegari 2021)	Hybrid	18.5	7.50	551	103.7	81.2
PartialNet-S (**ours**)	Hybrid	29.0	2.71	**1559**	**72.5**	**82.1**
EfficientNet-B4 (Tan and Le 2021)	CNN	19.0	4.20	356	156.9	82.9
FasterNet-M (Chen et al. 2023)	CNN	53.5	8.74	621	181.6	83.0
PoolFormer-M36 (Yu et al. 2022)	Hybrid	56.2	8.80	444	244.3	82.1
PoolFormer-M48 (Yu et al. 2022)	Hybrid	73.5	11.59	335	280.6	82.5
Swin-S (Liu et al. 2021)	Hybrid	49.6	8.77	477	199.1	83.0
PartialNet-M (**ours**)	Hybrid	58.3	6.69	**799**	**155.3**	**83.1**
FasterNet-L (Chen et al. 2023)	CNN	93.5	15.52	384	312.5	83.5
EfficientNet-B5 (Tan and Le 2021)	CNN	30.0	9.90	246	333.3	83.6
ConvNeXt-B (Woo et al. 2023)	CNN	88.6	15.38	322	317.1	83.8
Swin-B (Liu et al. 2021)	Hybrid	87.8	15.47	315	333.8	83.5
PartialNet-L (**ours**)	Hybrid	96.2	11.91	**426**	**272.5**	**83.9**

Table 1: ImageNet-1K comparison. Models are grouped by similar top-1 accuracy for fair and direct comparison.

Ablation Studies

Analysis of Partial Attention Mechanisms. We conduct comprehensive ablation studies on PartialNet-T2 to assess the effectiveness of our partial attention transformer convolution (PATConv), with results summarized in Table 3. By selectively replacing each PATConv component (channel-wise/ch, spatial-wise/sp, self-attention/sf) with its full attention counterpart while keeping other modules unchanged, we observe several key insights:

- **Channel Efficiency:** The F-P-P configuration increases parameters by 3.2% (13.0M vs. 12.6M) and FLOPs by 1.0% (1.04G vs. 1.03G) compared to P-P-P, but yields 0.1% lower accuracy (80.1% vs. 80.2%), indicating that partial channel attention achieves a better parameter-performance trade-off.
- **Spatial Computation Overhead:** P-F-P maintains similar parameters but increases FLOPs by 1.0% and reduces accuracy by 0.3%, highlighting the sensitivity of spatial attention to redundant computation.
- **Self-Attention Bottleneck:** The P-P-F variant incurs 9.7% higher FLOPs (1.12G vs. 1.03G) and 9.7% longer latency (38.6ms vs. 35.2ms) for the same accuracy, validating the computational efficiency of partial self-attention.

Further, Figure 6 visualizes attention maps using Grad-CAM (Selvaraju et al. 2017). PATConv produces more focused activations on discriminative object regions compared to full attention baselines, empirically supporting that partial channel processing preserves essential semantics while reducing redundant computation.

Component-wise Analysis of PATConv Modulations. Table 3 presents a stepwise evaluation of the impact of each attention modulation in PATConv. The baseline (w/o ch/sp/sf) achieves 76.0% Top-1 accuracy with 6405 FPS throughput, while the full PATConv (w./w./w.) improves accuracy by 4.2% to 80.2%. We observe that:

- **Channel Modulation Primacy:** Adding channel attention (ch) alone yields a 1.4% accuracy gain with minimal parameter increase, though at a 25.7% throughput cost.
- **Spatial Complementarity:** Incorporating spatial modulation (sp) further boosts accuracy by 1.5% (77.4%→78.9%) without increasing FLOPs, demonstrating efficient feature refinement.
- **Self-Attention Synergy:** Integrating self-attention (sf) provides an additional 1.3% accuracy (78.9%→80.2%) with a controlled 11.9% increase in FLOPs, achieving a strong accuracy-computation balance.

Backbone	Params (M)	FLOPs (G)	Throughput (FPS)	Object Detection			Instance Segmentation		
				AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet50 (He et al. 2015)	44.2	253	121	38.0	58.6	41.4	34.4	55.1	36.7
FasterNet-S (Chen et al. 2023)	49.0	258	121	39.9	61.2	43.6	36.9	58.1	39.7
PartialNet-S (**ours**)	46.9	216	**122**	**42.7**	**64.9**	**46.5**	**39.3**	**61.8**	**42.2**
ResNet101 (He et al. 2015)	63.2	329	62	40.4	61.1	44.2	36.4	57.7	38.8
FasterNet-M (Chen et al. 2023)	71.2	344	62	43.0	64.4	47.4	39.1	61.5	42.3
PartialNet-M (**ours**)	78.2	295	**65**	**44.3**	**65.8**	**48.5**	**40.6**	**63.3**	**43.7**
FasterNet-L (Chen et al. 2023)	110.9	484	35	44.0	65.6	48.2	39.9	62.3	43.0
PartialNet-L (**ours**)	122.0	397	**39**	**44.7**	**66.3**	**49.0**	**41.0**	**63.7**	**44.2**

Table 2: Results using PartialNet-S/M/L on object detection and instance segmentation benchmark in COCO dataset.

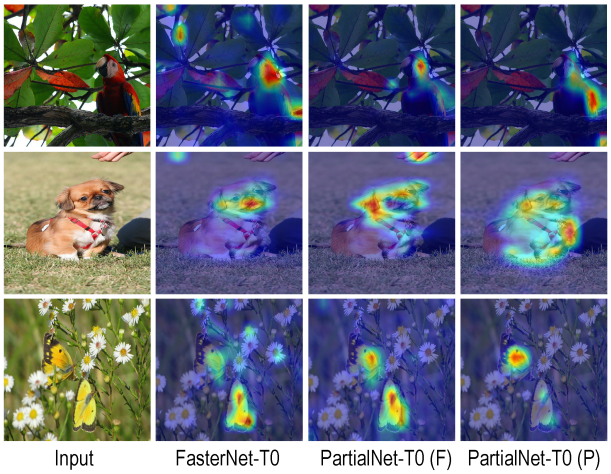

Figure 6: Visualization results show different categories of the ImageNet-1K validation set using Grad-CAM.

Cross-Architecture Generalization. Table 4 demonstrates that PATConv consistently boosts both throughput and accuracy across different architectures. For ResNet50, PATConv significantly increases throughput from 1258 to 2832 FPS, along with a 1.98% accuracy improvement. In MobileNetV2, it achieves 16.2% faster inference and a 3.81% accuracy gain. For ConvNext-tiny, throughput is enhanced by 24.5%, with a corresponding 3.38% increase in accuracy. Training curves in Figure 8 (Appendix) further illustrate that PATConv accelerates convergence, indicating improved optimization. These results underscore the versatility and effectiveness of our modulation design.

Efficiency Analysis of Partial Attention Convolution Table 5 compares PAT_{ch} with standard and depthwise convolutions. PAT_{ch} reduces parameters by 20% and slightly improves Top-1 accuracy over standard convolution. It also cuts FLOPs by 19.5% compared to depthwise convolution, maintaining similar latency and achieving higher throughput. These results highlight PAT_{ch}'s superior efficiency and accuracy in practical deployment.

Conclusion

This study presents a partial channel computation paradigm for efficient visual recognition. Key findings include: (1) Heterogeneous channel splitting improves the accuracy-throughput trade-off; (2) PATConv replaces convolution and attention, reducing FLOPs and speeding up processing; (3) DPConv enables adaptive layer-wise sparsity, enhancing compression efficiency; (4) PartialNet outperforms CNNs and Transformers on ImageNet-1K and COCO with similar costs. These results are achieved through parallel partial processing, multi-granular attention, and hardware-aware scaling, offering a solution for accuracy-efficiency trade-offs in deep vision models.

ch-sp-sf	Params (M)	FLOPs (G)	Throughput (FPS)	Latency (ms)	Top-1 (%)
P-P-P	12.6	1.03	**3074**	**35.2**	**80.2**
F-P-P	13.0	1.04	2975	36.5	80.1
P-F-P	12.6	1.04	3001	35.6	79.9
P-P-F	14.5	1.12	2913	38.6	**80.2**
✗-✗-✗	11.1	0.92	**4718**	**25.7**	76.0
✓-✗-✗	11.1	0.92	3753	30.9	77.4
✓-✓-✗	11.5	0.92	3470	31.7	78.9
✓-✓-✓	12.6	1.03	3074	35.2	**80.2**

Table 3: Ablation studies on PartialNet-T2 with different configurations of partial attention (P) and full attention (F) on ImageNet-1K dataset. The "ch", "sp", and "sf" denote channel-wise attention, spatial-wise attention, and self-attention respectively. The symbols ✓ and ✗ indicate the presence or absence of each attention type.

Model	Throughput(FPS)		Top-1(%)	
	original	PATConv	original	PATConv
ResNet50	1258	**2832** +12.30%	76.13	**77.64** +1.98%
MobileNetV2	3924	**4560** +16.21%	71.14	**73.85** +3.81%
ConvNext-tiny	902	**1123** +24.50%	76.00	**78.57** +3.38%

Table 4: The results of applying PATConv to other models.

Conv3×3	Params (M)	FLOPs (G)	Throughput (FPS)	Latency (ms)	Top-1 (%)
PAT_{ch}	**12.6**	**1.03**	**3074**	**35.2**	**80.2**
Conv	15.8	2.12	2535	49.9	79.9
DWConv	15.8	1.28	3001	35.4	79.6

Table 5: Ablation study on PartialNet-T2 with various convolution types on the ImageNet-1K dataset.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China No. 62088102, No.62302381 and No.52441602. The authors are with the National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center of Visual Information and Applications, and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an, Shaanxi, China.

References

Cai, H.; Li, J.; Hu, M.; Gan, C.; and Han, S. 2023. EfficientViT: Lightweight Multi-Scale Attention for High-Resolution Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17302–17313.

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12021–12031.

Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*.

Ding, X.; Zhang, X.; Han, J.; and Ding, G. 2022. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11963–11975.

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*, 12259–12269.

Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; and Xu, C. 2020. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1580–1589.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. *CoRR*, abs/1512.03385.

Hou, Q.; Lu, C.-Z.; Cheng, M.-M.; and Feng, J. 2022. Conv2former: A simple transformer-style convnet for visual recognition. *arXiv preprint arXiv:2211.11943*.

Howard, A.; Sandler, M.; Chu, G.; Chen, L.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; Le, Q. V.; and Adam, H. 2019. Searching for MobileNetV3. *Proceedings of the IEEE/CVF international conference on computer vision*, abs/1905.02244.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7132–7141.

Huang, T.; You, S.; Wang, F.; Qian, C.; and Xu, C. 2022. Knowledge distillation from a stronger teacher. *Advances in Neural Information Processing Systems*, 35: 33716–33727.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456.

Kirk, D. B.; and Wen-Mei, W. H. 2016. *Programming massively parallel processors: a hands-on approach*.

Li, Y.; Hu, J.; Wen, Y.; Evangelidis, G.; Salahi, K.; Wang, Y.; Tulyakov, S.; and Ren, J. 2023. Rethinking vision transformers for mobilenet size and speed. In *Proceedings of the IEEE/CVF international conference on computer vision*, 16889–16900.

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, 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*, 10012–10022.

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. Separable self-attention for mobile vision transformers. *arXiv preprint arXiv:2206.02680*.

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 *European Conference on Computer Vision*, 294–311.

Paul, S.; and Chen, P.-Y. 2022. Vision transformers are robust learners. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 36, 2071–2081.

Raghu, M.; Unterthiner, T.; Kornblith, S.; Zhang, C.; and Dosovitskiy, A. 2021. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34: 12116–12128.

Rao, Y.; Zhao, W.; Tang, Y.; Zhou, J.; Lim, S. N.; and Lu, J. 2022. Hornet: Efficient high-order spatial interactions with recursive gated convolutions. *Advances in Neural Information Processing Systems*, 35: 10353–10366.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–4520.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization.

In *Proceedings of the IEEE international conference on computer vision*, 618–626.

Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.; Howard, A.; and Le, Q. V. 2019. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2820–2828.

Tan, M.; and Le, Q. 2019a. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, 6105–6114.

Tan, M.; and Le, Q. 2021. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, 10096–10106.

Vasu, P. K. A.; Gabriel, J.; Zhu, J.; Tuzel, O.; and Ranjan, A. 2023. MobileOne: An Improved One Millisecond Mobile Backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7907–7917.

Wang, A.; Chen, H.; Lin, Z.; Pu, H.; and Ding, G. 2023a. RepViT: Revisiting Mobile CNN From ViT Perspective. *arXiv preprint arXiv:2307.09283*.

Wang, J.; Zhang, S.; Liu, Y.; Wu, T.; Yang, Y.; Liu, X.; Chen, K.; Luo, P.; and Lin, D. 2023b. RIFormer: Keep Your Vision Backbone Effective But Removing Token Mixer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14443–14452.

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

Woo, S.; Debnath, S.; Hu, R.; Chen, X.; Liu, Z.; Kweon, I. S.; and Xie, S. 2023. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16133–16142.

Woo, S.; Park, J.; Lee, J.-Y.; and Kweon, I. S. 2018. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, 3–19.

Wu, K.; Peng, H.; Chen, M.; Fu, J.; and Chao, H. 2021. Rethinking and improving relative position encoding for vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10033–10041.

Yang, J.; Li, C.; Dai, X.; and Gao, J. 2022. Focal modulation networks. *Advances in Neural Information Processing Systems*, 35: 4203–4217.

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*, 10819–10829.

Zhang, J.; Hu, T.; He, H.; Xue, Z.; Wang, Y.; Wang, C.; Liu, Y.; Li, X.; and Tao, D. 2024. EMOv2: Pushing 5M Vision Model Frontier. *arXiv preprint arXiv:2412.06674*.

Zhang, Z.; Li, J.; Shao, W.; Peng, Z.; Zhang, R.; Wang, X.; and Luo, P. 2019. Differentiable learning-to-group channels via groupable convolutional neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3542–3551.

Zheng, C. 2025. Iformer: Integrating ConvNet and transformer for mobile application. *arXiv preprint arXiv:2501.15369*.